



HAL
open science

Automated Collation and Digital Editions: from Theory to Practice

Elisa Nury

► **To cite this version:**

Elisa Nury. Automated Collation and Digital Editions: from Theory to Practice. Classical studies. King's College London, 2018. English. NNT: . tel-02493805

HAL Id: tel-02493805

<https://hal.science/tel-02493805v1>

Submitted on 28 Feb 2020

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

AUTOMATED COLLATION
and
DIGITAL EDITIONS

From Theory to Practice

By **Elisa Nury**

*Submitted in fulfillment of the requirements
for the degree of Doctor of Philosophy*

in the

*Department of Digital Humanities
School of Arts and Humanities
King's College London, University of London*

December 2018

Supervisors:

Prof. Elena Pierazzo

Department of Digital Humanities, King's College London
Laboratoire Universitaire Histoire Cultures Italie Europe, Université
Grenoble-Alpes

Dr Victoria Moul

Department of Classics, King's College London

Acknowledgements

FIRSTLY, my deepest gratitude goes to my supervisors, Prof. Elena Pierazzo and Dr Victoria Moul. I could never thank enough Prof. Pierazzo for her guidance through the ups and down of the past four years, her encouragements and her always constructive advice. I am also grateful to Dr Victoria Moul, my second supervisor, for the interesting discussions with a fresh point of view. This work has much benefited from her careful proofreading. Thanks are also due to Prof. Tara Andrews and Dr Hugh Houghton, who kindly accepted to be members of my thesis jury.

A special word of thanks to Prof. Danielle Van Mal-Maeder, who introduced me to the world of Roman Declamations, Dr Aris Xanthos, and Lavinia Galli-Milić. Without them, I would not have been able to start this PhD in the first place.

I was incredibly lucky to meet some amazing friends and colleagues at King's College: Gabby, Ginestra, Debby, Tuomo, Silke, Valeria and Simona. You have made my time in London such an enjoyable experience! I miss the fun lunchtimes, the dinosaurs, and our 'coffee and cake' breaks.

Many other people from different institutions have provided their advice and generous support. I am grateful to the team at the *Zentrum für Informationsmodellierung* in Karl-Franzen Universität, Graz: to Dr Walter Scholger who organised my stay in the centre in 2016, and also to Dr Roman Bleier and to Prof. Georg Vögeler for their insights and comments. Chapter 4 is greatly indebted to them.

Thanks to Prof. Michael Winterbottom, who discussed the tradition of Calpurnius Flaccus with me and helped me refine the collation visualisation of Chapter 8. I'm indebted to Prof. Andrea Balbo, who kindly tested the various interfaces of PyCoviz and gave me valuable feedback. Thanks to Prof. Hayim Lapin, who shared with me his work on the Digital Mishnah and collation visualisation. I am also grateful to all who took the time to help me with several automated collation tools, in particular

Ronald Dekker and Gregor Middell, Ilse de Vos and Anna Jordanous, Stefan Hagel, Marcus Pöckelmann. To everyone who patiently responded to my emails with endless questions, thank you!

Finally, I would like to thank my family, and especially Anurag, my \LaTeX wizard, who never stopped believing in me.



This research was made possible by the funding received from the Fondation Butticaz in Lausanne, and from the Swiss National Foundation for Science (SNSF) under project P1SKP1 155121.

Contents

Contents	4
List of Figures	7
List of Tables	11
General Introduction	12

THEORY

1 Collation	20
1.1 Definition	20
1.2 Historical Context	21
1.3 Purpose of Collation	23
1.4 Issues of Collation	28
2 The Theory of Automated Collation	44
2.1 Automatisations of Collation	44
2.2 Collation algorithms	49
2.3 The Evolution of Automated Collation	60
2.4 Methodology of Automated Collation	67
2.5 Advantages of Computers and Black Box Issue	87
2.6 Comparing collation tools	91
2.7 Conclusion	109
3 Transcription: a Prerequisite for Automated Collation	111
3.1 Theory of Transcription	111
3.2 Transcription and Collation: a Common Model?	117
3.3 Transcription Issues Related to Automated Collation	127
3.4 Conclusion	135

4 Readings and Variants	137
4.1 Readings in Context	140
4.2 Modelling a Reading	144
4.3 Digital Representation: from Reading to Token	150
4.4 Comparing Tokens in Different Contexts	154
4.5 Conclusion	157
PRACTICE	
<hr/>	
5 Test Case: the <i>Declamations</i> of Calpurnius Flaccus	163
5.1 The Tradition of Calpurnius Flaccus	163
5.2 Method: Automated Collation Applied to a Classical Text	178
6 XML Transcriptions	182
6.1 Transcription platforms	182
6.2 Description of the TEI Encoding	187
7 Automated Collation in Practice	205
7.1 CollateX	206
7.2 Juxta	221
7.3 Classical Text Editor	231
7.4 Conclusion	238
8 Collation Visualisation	240
8.1 Assessing Scholarly Needs	240
8.2 Table format: fixed visualisation	246
8.3 Interactive interfaces	254
8.4 PyCoviz: A Python Interactive Interface	263
8.5 PyCoviz Applied to Calpurnius Flaccus	283
8.6 Discussion	290
8.7 Conclusion	295
General Conclusion	300

APPENDICES

A Theory	305
A.1 List of tools — Automated Collation	305
A.2 Collation Model	317
B Practice	319
B.1 XML Transcriptions	319
B.2 XSLT Transformation	320
B.3 JSON Collation Results	321
B.4 HTML Tables	322
B.5 PyCoviz Jupyter Notebook	323
Bibliography	327

List of Figures

1.1	Lendle's notation method of collation.	36
1.2	Manual collation in Excel 1.	37
1.3	Manual collation in Excel 2.	38
2.1	Optical text collation with the Oxford Traherne.	46
2.2	A collation algorithm in natural language.	47
2.3	A collation algorithm in flowchart.	47
2.4	Dekker algorithm, alignment matrix.	56
2.5	Examples of alignments from iAligner.	58
2.6	Diff example.	59
2.7	Collate — base text collation vs. parallel segmentation.	72
2.8	Parallel segmentation of a text.	73
2.9	Variant graph: Colwell and Tune (1964).	74
2.10	Variant graph: Schmidt and Colomb (2009).	75
2.11	Variant graph: CollateX.	76
2.12	Tokenisation with markup context.	79
2.13	Examples of incorrect alignment.	80
2.14	Normalised and non-normalised texts.	81
2.15	Comparing exact matches first.	85
2.16	Homonyms and transpositions in CollateX.	86
2.17	Example of a punched card.	97
2.18	OCR with ABBYY FineReader.	99
2.19	Example of early collation output.	105
2.20	A collation visualisation in histogram.	106
2.21	Different highlightings in Juxta's Heat Map.	107
2.22	Variant graph: Stemmaweb.	108
2.23	Variant graph: TRAViz.	108
3.1	Summary of the transcription model.	116
3.2	Transcription model and individual readings.	122

3.3	Collation model: without variant.	124
3.4	Collation model: with variant.	124
3.5	Deletion and addition in Juxta Commons.	131
4.1	Readings.	138
4.2	Readings, differences and variants.	141
4.3	Sahle's wheel of text model.	145
4.4	Model for readings.	149
4.5	Every difference is a variant.	155
4.6	Non-orthographic differences are variants.	156
4.7	Orthographic differences are variants.	156
5.1	The <i>incipit</i> of manuscript A.	168
5.2	Manuscript C, folio 84r, line 38.	169
5.3	Additions in the margin of manuscript B.	170
5.4	Additions in manuscript B.	170
5.5	Manuscript M, folio 18v, lines 20-25.	171
5.6	Manuscript N, folio 246r, line 13.	172
5.7	Manuscript N, folio 259v, line 24.	172
5.8	Manuscript N, folio 254r, lines 19-22.	172
5.9	Comparison of corrections in manuscripts B and N.	173
5.10	Manuscript M, folio 5v, line 18	174
5.11	Calpurnius Flaccus stemma.	177
6.1	Transcription platform: TextGridLab.	183
6.2	Transcription platform: the Transcription Editor.	184
6.3	Transcription platform: T-Pen.	185
6.4	Transcription platform: HumEdit.	186
6.5	Transcription platform: Transkribus.	187
6.6	Unknown abbreviation in manuscripts B and C.	194
6.7	Pithoeus apparatus.	201
7.1	CollateX command line usage.	207
7.2	CollateX JSON input format.	212
7.3	Sample 1 from the XSLT transformation.	216
7.4	Sample 2 from the XSLT transformation.	218
7.5	Sample 3 from the XSLT transformation.	218
7.6	CollateX JSON input and XML TEI output.	221
7.7	Juxta Desktop interface.	222
7.8	Juxta XML parsing template.	223

7.9 Juxta Commons interface.	225
7.10 Juxta Commons, Versioning Machine visualisation.	226
7.11 Example of erroneous alignment in Juxta Commons.	227
7.12 The interface of Juxta Editions.	228
7.13 Classical Text Editor interface.	232
7.14 The Collation dialog box (CTE).	234
7.15 Issues of alignment with CTE algorithm.	235
7.16 Updates to the algorithm of CTE provided by Hagel.	236
8.1 An example of a CollateX Collation table.	246
8.2 A collation table form the Digital Mishnah project.	247
8.3 A collation table from the Beckett Archive Project.	248
8.4 An example of a Compare collation table.	248
8.5 A collation table from iAligner.	249
8.6 Example of a collation table in Stemmaweb's Stexaminer.	250
8.7 A collation table for Calpurnius Flaccus.	251
8.8 Javascript for expanding or hiding paragraphs.	254
8.9 Stemmaweb interface.	256
8.10 Stemmaweb graph example.	256
8.11 Modifying collation results with the Collation Editor.	258
8.12 Example of a TRAViz graph interface.	258
8.13 Example of interface with widgets.	260
8.14 Structure of CollateX JSON results.	264
8.15 Widgets for the selection of a table extract.	267
8.16 Summary of modifications available in PyCoviz.	269
8.17 Move one token down.	270
8.18 An alignment that needs to be updated.	271
8.19 Adding a note to a token.	271
8.20 The Agreements widget.	275
8.21 PyCoviz [4] - compare multiple cells.	277
8.22 PyCoviz [6] - compare witnesses.	277
8.23 The comparison of normalised forms ignores word separation.	278
8.24 M compared to BCN and P1594.	279
8.25 The Search widget.	280
8.26 The Clarify widget.	281
8.27 PyCoviz [11] - print info.	281
8.28 PyCoviz [5] - Find agreements.	282
8.29 PyCoviz [7] - View variants.	282
8.30 PyCoviz [10] - A short passage in text format.	283

List of Figures

8.31	Example of corrected alignment.	284
8.32	Pithoeus and B against CMN.	286
8.33	Pithoeus and C against BMN.	287
8.34	Pithoeus and N against BCM.	287
8.35	Four readings are maybe exchanged between B and N.	289
8.36	The reading <i>liniamentis</i> in the collation results.	291
8.37	Joint vs. separated tokens outputs.	292
8.38	Example of a collation table from a Digital Mishnah sample.	293
B.2	Example from the XSLT transformation files.	321
B.3	Example of CollateX JSON output.	325
B.4	HTML collation table visualised in the browser.	325
B.5	Sample of javascript code for the HTML collation table.	325
B.6	Agreements widget.	326
B.7	Code that select variant readings for the Agreement widget.	326

List of Tables

2.1	Incorrect progressive alignment.	55
2.2	Correct non-progressive alignments.	55
2.3	Preference for substitution over addition and omission.	84
2.4	The issue of aligning homonyms.	85
8.1	Relations of B2 with M, N, and P1594.	289

General Introduction

THE purpose of the dissertation is to investigate from a theoretical and methodological perspective the different tools that allow automated collation, and study the application of such tools to the creation of a digital critical edition in the context of Classical literature. By doing so, the dissertation examines many foundational but often neglected components of the philological method, such as the definition and wider implication of transcription, reading, and variant.

The goal is to provide a reflection on automated collation and the theoretical as well as practical challenges it poses: what is automated collation? How is it performed, and what are the main differences with manual collation? What are the benefits of automated collation? Why has it not been widely adopted yet, despite the fact that it was developed to help scholars? How to process the results of collation programmes? As a case study, a Classical Latin text has been used to test automated collation and to compare the various existing tools.

The method I follow in this dissertation is to apply automated collation to a selected text, the *Declamations* of Calpurnius Flaccus. To this purpose, the manuscript tradition, as well as the *editio princeps*, have been entirely transcribed. Afterwards, the transcriptions have been collated with different collation programmes. The results of the collation with different programmes have been examined and compared, as well as the possibilities offered to scholarly editors for visualising and further processing those results.

The content of the thesis is divided into two distinct parts, from theory to practice: a first part discusses the relevance of collation in the broader context of critical editing, introduces automated collation and its various issues, as well as the implications of automated collation for key concepts such as ‘witnesses’, ‘readings’ and ‘variants’; the second part of the thesis will describe the practical work on the text of Calpurnius Flaccus with automated collation programmes and the visualisation tool that was created to examine the collation results. Each part comprises a short

introduction and conclusion which summarises its content and its outcome.



In the process of editing a text, the collation of manuscripts and previous scholarly editions is a necessary and fundamental step. The work leading to the production of a critical edition is divided into two phases: the recension of witnesses, followed by the constitution of the text (Chiesa 2002). The editor, therefore, starts with the recension by gathering all the witnesses, manuscripts or editions, bearing a version of the edited text. Collation is the next step: comparing those witnesses to find the differences (or variants) between the versions. Finally, the editor analyses the variants in order to determine the genealogical relationships of the witnesses, if possible, and present those relationships in the form of a *stemma codicum*. The stemma's purpose is to help the editor produce a text 'as close as possible to the original' (Maas 1958, 1), and decide which variants are accepted as authorial and which variants are rejected as errors that got included in the tradition by the copyists of the manuscripts. After the recension, the editor prepares a critical text, selecting variant readings and making emendations when necessary.

Collation is important because it is one of the first stages in the editing process and the data gathered during collation forms the basis upon which the editor will later make critical decisions (Whittaker 1991). Collation is performed because witnesses of a text always contain variant readings, and the editor needs to be in possession of all the alternatives in order to establish the text. As complete collations are not usually published, this represents a regrettable loss of information, especially given the amount of time and effort invested in collation by the editors (West 1973, 63).

While collation is an essential part of textual criticism, collation is also a long, tedious and error-prone activity and needs to be checked more than once. For this reason, a new method was created: automated collation, which takes advantage of computers in order to compare texts and find variant readings. Scholars have been developing collation tools since the 1960s, but with limited success at first: what was considered a fairly mechanical process turned out to be more sophisticated than expected (Hockey 2000, 125). Since the pioneering work of Dearing (1962) and Froger (1968), automated collation has been studied for decades and has been constantly improved. In the past 50 years, close to thirty tools have been devised in order to obtain a collation with the support of increasingly complex algorithms.

How does automated collation work? To simplify, the tools for automated collation take, as an input, a transcription of each witness that needs to be compared. The transcriptions are then aligned with each other through an alignment algorithm.

The task of collation seems to highly benefit from the application of computing methods. The advantages offered by computing methods are various: consistency in the comparison, possibility to reuse the material in order to add new manuscripts to the collation, a common format to share collation data with other scholars. The results of automated collation can also be formatted for further processing, such as building a stemma with a different program or creating a digital scholarly edition.

However, automated collation is not completely accepted by the community of scholars, nor is its method fully understood. In 1973, at the beginning of automated collation, scepticism was understandable because of the many restrictions of early tools such as the small number of witnesses collated, or the limitation to comparing lines of poetry. Faced with those limitations, West (1973, 71) stated that ‘the time has not yet come when manuscripts can be collated automatically’. Forty years later, in spite of huge technical improvements, opinions have not really changed, to the point that Reeve declared that he was not convinced by computer methods, especially for large manuscript traditions (Reeve 2011, 393). But the computer is indeed supposed to be better at handling large amounts of data, and with more consistency than a human being. In fact, when dealing with large traditions, it is not always possible to sort out the relationships between manuscripts and draw a stemma by hand, yet editors are not keen on turning to electronic methods.

Some of the obstacles to the wide adoption of automated collation seem connected to a general misunderstanding. There is a fear that somehow the computer will eventually be replacing the editor, and that editors will lose their right to apply their individual judgement to the texts. Greetham (2007, 23) regrets, for instance, that the role of individual, subjective evaluation ‘has not always been recognised, especially by those wishing to emphasise the “scientific” aspects of the field’. The importance of individual judgement, and the role of the editor compared to the role of the computer, can be closely related to the black box issue (Sculley and Pasanek 2008): if scholars do not understand what a piece of software does, how can they trust that the programme did not deprive them of making certain choices or applying their own judgement?

In the course of my PhD I had several exchanges with colleagues in the field of Classics which have highlighted several underlying misunderstandings on either side.

For instance I was once told in an email that ‘automated collation is impossible, because computers cannot read manuscripts’. This statement does not recognise, for instance, the fact that computers are collating from transcriptions made by scholars. From this point of view, transcription and collation are strictly connected activities. This statement, however, illustrates perfectly the main tension between traditional, manual collation and automated collation: the difference of methodology. The confusion arises here because the full transcription of manuscripts is not a part of the traditional heuristics in textual criticism.

A generalised lack of tools and guidelines for the production of digital scholarly editions may also explain why automated collation is not the solution of choice. As yet, there is still no clear consensus regarding digital critical editions and their definition (‘what they should be’), which results in a lack of widely applicable tools (Andrews 2012). The scarcity of user-friendly tools for Humanities scholars is also noted by Robinson (2005, §13) and Monella (2014b), whereas Cayless (2015) mentions the absence of precise guidelines and orientation among the many options available. In fact, there is no lack of collation tools, but still little literature offers criticism to help scholars evaluate their growing number. Furthermore, collation is not the final goal but only one step in the course of editing. The results of automated collation are not easily manageable by scholars: they need to be analysed and manipulated to be fully understood. However there are not many options to do so with the current collation tools.



In light of this discussion, it appears that many aspects of collation and automated collation need to be discussed. It is the purpose of this dissertation to address concerns expressed by Classicists while clarifying the misunderstandings which can hamper the adoption of automated collation. Ultimately, the aim of the dissertation is to show how the flexible results of automated collation can support scholars in their work. The disciplinary background of the thesis is thus primarily in Classics, but it will also explore practices in the wider domain of textual criticism.

Chapter 1 presents traditional collation and its method, which is necessary in order to understand what changes with the adoption of a new automated method. Chapter 2 is dedicated to the history and methodology of automated collation. Since an accurate transcription is the crucial first step to achieve in order to undertake an au-

tomated collation, the dissertation devotes two chapters to transcription: Chapter 3 for theoretical aspects, and Chapter 6 to describe the transcription of Calpurnius' Declamations. Transcription also represents a major change in methodology, and the implications of the differences in heuristics will be explored further on in the first part of the dissertation, especially in Chapter 2 and Chapter 3.

The content of collation is determined by what editors consider a variant worthy of being recorded: therefore the definition of a variant, especially a significant variant, is crucial (Orlandi 2010, 115). Every editor has a different sensitivity about what constitutes a significant variant. Two editors will not produce the same edition of the same text. Moreover, what is significant for a historical linguist is not the same as what is relevant to the medievalist or the stemmatology specialist. This dissertation will therefore provide a reflection on what is a reading, a variant, a significant variant, and how these definitions may change according to the editor's field of study and perspective on the text (Chapter 4).

Since many collation tools are available and new ones are regularly created, there is a growing need for scholars to compare different tools and assess which one will be best suited to their needs. Chapter 2 provides a theoretical framework of tool criticism for automated collation with several criteria, and Chapter 7 applies those criteria to the comparison of three tools in practice. As a result of this comparison, it was possible to identify the need for a tool that would help scholars to manipulate collation results in a way that supports traditional textual criticism, which led to the development of a tool for this purpose.

In Chapter 8, I will describe the tool that I have created to visualise collation results and show how its application can support editors in applying their own judgement to the text, and also make their conclusions reproducible by others. While the collation tool does not make any judgement regarding the correctness of the text, scholars should nevertheless be aware of how the collation results format can affect the visualisation of variant readings and ultimately of the critical apparatus.

The shift from the traditional manual comparison of manuscripts to a transcription followed by an automated collation represents an important change in heuristics. What does this change implies for the understanding of what scholars do when they collate? This dissertation ultimately argues that the transition to computing methodology for textual scholarship has a profound impact on the understanding of the edited text, its variation and its meaning.

PART I

THEORY

Introduction

THE first part of this dissertation is focused on the theoretical aspects of collation. The purpose of this section is to give a theoretical framework of collation that will bring together the various issues of collation, whether manual or automated collation, and examine how they have been addressed.

In this part of the thesis, I will start in Chapter 1 by introducing the concept of collation and the purpose as well as the historical context for this critical activity which has been performed by scholars over the centuries. The different steps of collation and their issues will be reviewed, as well as the content of collation. The next chapters will then focus on automated collation and the theoretical issues surrounding this new digital method:

- **Automated Collation:** Chapter 2 presents a comprehensive discussion of the theory of automated collation. The chapter reviews not only the evolution of the name, definition and methodology of automated collation, but it also highlights the major issues related to each step of the process. As new tools continue to emerge, it seems necessary to establish criteria in order to compare collation tools and to help scholars select the most useful one depending on their needs.
- **Transcription:** in Chapter 3, I will compare transcription and collation. Thanks to the model of transcription by Huitfeldt, Marcoux, and Sperberg-McQueen (2008), I will highlight how the two activities are not so different from one another. However, I will also examine how the change from a manual collation to a digital transcription followed by automated collation forces scholars to think more deeply about the notion of witnesses, and the division of the text into units of comparison (i.e. readings, or tokens).
- **Reading and Variants:** finally, in Chapter 4, I will consider more closely these

units of comparison, and propose a model for representing readings into digital tokens, inspired by one of CollateX's formats.

This first part of the dissertation aims not only at providing the context necessary to the second part, but also at examining what the change of method implies for our scholarly understanding of collation. In this context, it is especially important to discuss definitions, as well as the purpose and methodology of collation. Although automated collation is a relatively recent approach that started only a few decades ago, it has evolved since its beginning in the 1960s: its name and definition, its purpose and methodology have changed not only in parallel to technological improvements, but also according to scholarly practices and their understanding of automated collation. It is likely still changing, with a trend towards 'alignment' instead of 'collation'.

Theoretical issues are not a strong focus in the literature of automated collation. For instance the Gothenburg model, which formalises the process of automated collation, is not discussed in detail in a scholarly publication. The discussion is often oriented towards practical aspects: how is automated collation performed? What can be achieved thanks to automated collation? How many texts can be collated? What is the percentage of errors in the results? However, automated collation also needs to be discussed with regard to the implications for editors: how is the digital method influencing collation? What decisions must be made during the transcription? What transformations are texts going through as they are first transcribed, then collated and analysed, and what are the consequences?

Collation

1

1.1 Definition

THE term collation etymologically means to ‘collect and combine (text, information or data)’ (Oxford Dictionaries), from the Latin *conferre*, ‘to gather, compare’. In philology, collation is commonly described as the operation of comparing each witness of a text with one another, so as to reveal their differences and with the aim of creating a critical edition (Chiesa 2002, 48-49)¹. The term ‘collation’ may refer both to the action of comparing texts and to the result of this comparison. The following examples illustrate how the term ‘collation’ can be used to refer to the result of a comparison: ‘collations may sometimes be used by other people’ (Macé et al. 2015, 331); ‘the digital edition gives many more collations’ (Robinson 2007a, 4); ‘The most obvious procedure would be to collate against [the manuscript] M [...] taking Vitelli’s collation as the standard against which all other manuscripts were to be judged’ (Dawe 1964, 15). In those examples, collation does not refer to the action of comparing texts, but carries the concrete meaning of something that can be used by, or given to, interested scholars. Nevertheless definitions do not usually distinguish between collation as an act and collation as a result, but rather focus on the process or action of comparing texts, word by word, character by character². Non-textual elements, such as decorations, calendars or diagrams, and the issues of their comparison are not discussed in relation to philology or textual criticism. However, historians of Greek, Arabic or Egyptian mathematics have acknowledged the need to collate and critically edit mathematical diagrams instead of simply providing corrected figures to fit modern standards³.

¹‘La seconda operazione che porta all’edizione critica, successiva alla ricognizione dei testimoni, è la loro collazione. Essa consiste nel confronto (così etimologicamente: collatio, dal latino confero) di ciascun testimone con gli altri, al fine di rilevarne le differenze’ (Chiesa 2002, 48-49). For definitions specific to book-related disciplines, such as codicology, bookbinding or analytical bibliography, see for instance ‘Collation’ in the Online Dictionary for Library and Information Science (Reitz 2013).

²For more definitions, see the Lexicon of Scholarly Editing, s.v. collation: <http://uahost.uantwerpen.be/lse/index.php/lexicon/collation/> (Accessed July 15, 2015).

³For more information, see De Young (2009).

In this chapter I have mainly surveyed sources which are discussing ancient texts, mostly Greek and Latin Classics. However, other disciplines with different practices are also relevant to the thesis, in particular New Testament studies which have worked extensively with automated collation.

A general definition such as the one provided by Chiesa is hardly satisfying. What do we do, precisely? How do we compare texts? From which type of support: manuscripts, printed photographs, microfilms, digital images? How does the use of manuscript or facsimiles of varying format and quality influence the results? There is an infinity of differences between two documents, how do we determine what is worth recording? According to Reeve ([1989] 2011) physical features (which could be considered codicological, such as damage, misbinding, and ‘quirks of layout’)⁴, as well as textual evidence, must be considered, but how? Is establishing a critical edition the only purpose of collation? Has it always been the case?

1.2 Historical Context

Collation, that is comparing texts with each other, is an activity as old as writing, for ‘at the very moment in each culture that documents begin to preserve the records of that culture, the issues familiar to textual scholars will appear’ including textual variation (Greetham 2013, 17). Among the first known textual critics were Zenodotus, Aristophanes and Aristarchus, working in the Alexandrian library where they gathered multiple copies of the same work. According to Zetzel (1981, 16) however, the critical marks used by Aristarchus pertain more to a commentary than a critical apparatus or collation⁵. Such marks did not always refer to actual textual variation in the manuscripts but rather to what he considered to be not authentic material and should be rejected. Montanari (2002, 127 ff.), on the other hand, showed that when working on an *ekdosis* (edition), Alexandrian scholars performed a combination of both methods of textual criticism performed later by Humanist scholars: providing personal conjectures and collating documentary sources, the equivalents for *emendatio ope ingenii* and *emendatio ope codicum* (see p. 22 below). The introduction of the *obelos* by Zenodotus to mark suspicious, non-Homeric readings represents a ‘crucial intellectual step’, the shift from ‘correcting a

⁴Physical evidence is any peculiarity of a witness other than its readings that accounts for an innovation in another witness’ (Reeve [1989] 2011, 152). Reeve builds upon Timpanaro’s discussion on the “prove ‘materiali’” (Timpanaro 1985, 170 ff.). Timpanaro himself was discussing Maas (1958, 4), paragraph 8a, regarding the *eliminatio codicum descriptorum* based on ‘the external state of the text’. See also West (1973, 33, note 4).

⁵Critical signs (*σημεία*) were invented by Alexandrian scholars. The signs included obelos, a short line placed in the margin of the text, asteriskos, sigma, diples etc. For more information regarding the shape and meaning of those signs, see for instance Schironi (2012).

single copy' to 'editing a text' (Montanari 2015, 40). In addition, Montanari raises the question of the number of manuscripts involved in the comparison: can it be called a collation when only a very small number of manuscripts are compared? 'Or was it sufficient to compare a few, to detect variants when the tradition was not univocal, and then address the problem of which text was correct and which ones were wrong?' (Montanari 2015, 43).

The interests of early Latin critics were also clearly in identifying and compiling lists of manuscripts in an effort to establish chronology and authenticity. Even later, when their aim was to 'recover or explain the text, not to suggest what it should have been' (Zetzel 1981, 74), they 'paid little attention either to diplomatic evidence or to the stylistic character of the authors, two criteria which are basic to any modern textual argument' (Zetzel 1981, 206). The subscriptions written in various manuscripts do not suggest that collation was performed on a wide number of copies. On the contrary, it seems that 'one copy was corrected, and it was corrected against only one other copy, which may well have been the exemplar from which it was itself transcribed' (Zetzel 1981, 228). The variants recorded reflected personal interests, not the will to 'improve the text for posterity' (Zetzel 1981, 230). Educated readers, not professional critics, were responsible for most of the textual scholarship from Antiquity that survives in our manuscripts, and it could be misleading to assume that their methods and purposes were those of a modern scholar.

A major change occurred during the fifteenth century, when Politian laid the foundations of modern textual criticism. Until then editors used to follow a vulgate text, later transmitted from the *editio princeps* which was based on recent manuscripts. The vulgate was corrected either by conjecture (*emendatio ope ingenii*) or by collation of manuscripts (*emendatio ope codicum*), but only when it was not satisfactory. Politian was the first to understand that the collation had to remain separate from the emendation of the text (Rizzo 1973, 178), and that 'the manuscripts [...] had to be collated not occasionally but systematically, registering all the readings that diverged from the vulgate text' (Timpanaro 2005, 48) because he was aware that ancient manuscripts could provide variants closer to the 'truth' (see Rizzo 1973, 245)⁶. While humanists were noting down variants only when they seemed correct or at least worthy of consideration, modern editors would subsequently follow Politian's method and 'signal every difference in the collation exemplar (whether or

⁶The notion of a fixed, true original text, representing without doubt the author's intentions, has been challenged in the last decades (Chiesa 2002, 120).

not they will be printed in the apparatus)' (Pasquali 1952, 62)⁷. The aim for completeness makes the difference between humanists and modern editors: 'among modern editors, several are collators as inexperienced or careless as the humanists, although unlike them, they usually seek to be as complete as possible' (Pasquali 1952, 73)⁸.

1.3 Purpose of Collation

Collation is therefore prompted by a strong 'mistrust of texts', as Vinaver (1939, 352) famously formulated it, a mistrust coming both from the editors and their audience. The editors do not blindly rely on any witnesses but critically compare them in order to gather as much evidence as possible in their quest to recover what the author wrote⁹. On the other hand, the audience needs an apparatus of variants so as to question in turn the text (and possibly to contest the editor's decisions).

This 'mistrust' is well exemplified by West when he explains why it is imperative to collate at least the most important manuscripts. The main reason for collation is because the data needed to edit the text and recover as much as possible the original, if it exists, is not trustworthy. Even when a previous collation is available, it will contain mistakes: 'no one ever checks anybody else's collations (or his own, for that matter) without finding mistakes in them' (West 1973, 63). However, collations are not usually published, except in the incomplete form of a critical apparatus. The apparatus is only a selection from the variants used by the editor to establish the text, which are themselves only a selection among all the differences recorded during the collation. As West (1973, 63) noted, 'it is very likely that no complete collations have been published, only selected variants, and [the editor] will want to make his own selection from the complete evidence'. Previous collations are thus unreliable because they are erroneous and incomplete. West was concerned with editions of Greek and Latin texts. There is however some diversity in practice among the different domains of textual criticism: collations of one or more manuscripts were published for the New Testament, for instance by Scrivener in the nineteenth

⁷'[I]l metodo che seguiamo per lo più noi collazionatori moderni, di segnare nell'esemplare di collazione ogni divergenza (tranne poi a introdurla o no nell'apparato a stampa), ma che è rarissimo nei collazionatori del Rinascimento, anche tardo, che segnavano soltanto le varianti che parevano a loro giuste o almeno degne di considerazione' (Pasquali 1952, 62).

⁸'[...] tra gli editori moderni parecchi sono collazionatori altrettanto poco pratici o poco diligenti quanto gli umanisti, se pure, differientemente da questi, essi sogliono ricercare la maggiore completezza possibile' (Pasquali 1952, 73).

⁹It must be noted that the origin of a text is not always a written one, but an oral source which may not have been produced by a single author (for instance Homer's poems or the Sanskrit epic Mahabharata).

century (Parker 2008, 196-197).

The apparatus is recognised to give critical editions their scientific value (Boschetti 2007, 2). Its absence is often regretted in print: Kaster (1996, 85) for instance criticises Sussman (1994) for the absence of apparatus in his edition of Calpurnius. Parker (2012, 105-106) describes the components of a critical edition, among which the critical apparatus and critical text are universally agreed. However, editions of the New Testament without an apparatus, such as Westcott and Hort, may still be considered critical editions.

The absence of apparatus from otherwise good electronic editions is considered a major shortcoming for academic and pedagogical usage (Tissoni 2004, 74; 2008, 141-142). Electronic editions run the risk of being trusted more than they deserve because of the lack of editorial control and responsibility because anyone can upload anything online (Reeve 2000, 200), by giving the false impression that they represent the one and only true version of the text (Pierazzo 2015, 151-152), or by obliterating linguistic or stylistic phenomena (Boschetti 2007, 1).

Though provided so that readers do not depend on a potentially fallible editor (West 1973, 9), the apparatus is, nevertheless, subject to criticism. One of the reasons why the apparatus is criticised is its subjectivity by Flores (1998) and Cozzo (2006), two Italian scholars of Greek and Latin philology. Cozzo (2006) considers that the apparatus is subjective, and does not allow readers to make decisions independently of the editor. According to Cozzo (2006, 255), the presence of the apparatus gives the reader the illusion of being in possession of all the material in order to judge the editor's choices. Cozzo (2006) states that if the apparatus is based on a restricted selection of variants, it is impossible to recreate each witness, and the precise context for each variant is lost¹⁰.

For Flores (1998), this absence of precise context may be quite misleading, for instance when a manuscript with usually poor readings is quoted only in the very few cases where it yields plausible alternatives; the manuscript thus would convey a positive impression, albeit being altogether of an inferior quality (Flores 1998, 45-46). Flores states that the variants are not selected randomly but according to the *stemma codicum*, which makes it virtually impossible to produce a stemma different from the published one: the variants selected to appear in the apparatus

¹⁰Other scholars have underlined the issue of incomplete apparatuses, highlighting the imprecisions, errors and ambiguities contained in an apparatus (Dahlström 2000; Wiering 2010; Andrews and Macé 2013; Monella 2014b).

will reflect precisely the *stemma*, and according to him the variants in the apparatus only allow reconstitution of this particular *stemma* (Flores 1998, 43).

In addition, bias towards the edited text may be unavoidable, since it is much easier to accept the editor's reading text rather than attempting to reconstruct a text from variants scattered in an apparatus (Sperberg-McQueen 2009, 37, note 4). Those variants that have been rejected by the editor and placed in the apparatus may be perceived negatively because they represent the 'wrong thing' (Shillingsburg 1996, 118). This negative impression is emphasised by the apparatus position at the bottom of the page, smaller and less visible than the reading text (Cozzo 2006, 255; Epp 2007, 275). The variants are even less accessible when the apparatus is separated from the reading text and placed at the end of the edition (Sperberg-McQueen 2009, 33).

Machan (2002) raises another problem regarding the apparatus in an edition of Chaucer: it obscures the fact that 'Chaucer's putatively original reading (represented by the lemma) is a matter of speculation — informed speculation, to be sure, but speculation none the less'. That is, the apparatus does not always make it explicit when a variant has been preferred over others on the basis of speculation, without offering conclusive arguments, because it is simply impossible to ascertain which reading was the original one. When a doubt arises between different readings that the stemma cannot dissipate, Stussi (1994, 133) recommends first to apply the criterion of *usus scribendi*, to choose the reading which corresponds best to the style and language of the author (see also West 1973, 48). Then, the editor should prefer the *lectio difficilior*, or conjecture a *lectio difficilior* which accounts for the readings present in the tradition. The quality of a manuscript should serve as a criterion only when no other criteria can help judge between plausible variants (West 1973, 50). Stussi (1994, 152) also insists that when the editor does not know which reading to choose, they must signal it.

In his edition of Calpurnius Flaccus, Håkanson (1978) describes in the preface the variants he used to establish the stemma. Next to a few variants he expresses doubt with interrogation marks (*in eo est* (?) page VI, *gemitum miseri* (?) page VII, etc.), but it is difficult to extrapolate why these particular readings are dubious or what influenced his choice. Moreover, these interrogations are not reported inside the critical text. This leads to the question of whether Håkanson had doubts regarding variants not listed in the preface. On the other hand, another editor of Calpurnius, Lehnert (1903), did not show any hesitation for the same readings. Does that mean he was certain of his decisions, and if so, what were his motivations?

1.3. Purpose of Collation

Commentaries or ancillary articles can give a better appreciation of the editor's reasoning, personal interpretation, or hesitations only hinted at in the apparatus through expressions such as *fort. recte, dubitanter*, etc. Cozzo (2006, 256) states that '[i]n no part of the critical edition can the reader find reasons that led the editor to adopt one reading over another'¹¹. Cozzo's statement may be too radical, since editions can contain a philological commentary or a detailed introduction to justify the editor's choices, but it is not a standard practice to explain each decision regarding the choice among variants in the major published series of classical literature such as the *Bibliotheca Teubneriana*, the Collection Budé, or the Oxford Classical Texts. Parker (2012, 106) believes that a true critical edition should also offer a justification for each and every editorial decision; although such an edition does not exist yet for the New Testament, Parker (2012) thinks that it is now achievable.

1.3.1 Collation and Critical Apparatus: Summary

Vanhoutte (2011) summarises the objective of a critical apparatus, or a 'record of variants', in four points:

First, it is a documentation of the variation between all of the extant versions of a text which allows for the reconstruction of these versions. In pre-digital times this was the only affordable way to represent the genetic and transmissional history of texts. Second, it provides the account of the emendation of the base text and the constitution of the reading text. Third, it provides the user with control data which allows for the repeatability of the criticism performed on the text. Four, it functions as a research data base. It is in the record of variants that scholarly editors expose themselves and are explicit about their choices.

It is interesting to consider those four points and see if indeed they apply to critical apparatuses. First, an apparatus is not always complete enough to reconstruct the different versions of the text (see Section 1.3 above). The third point is debatable: as we have seen above, the apparatus will generally only allow confirmation of the editor's criticism and as a result fails in its primary purpose, which is to avoid dependence on the editor's judgement (West 1973, 9). The last sentence of Vanhoutte's statement as well as the second point could be challenged: apparatuses do

¹¹'In nessuna parte dell'edizione critica il lettore trova spiegazione delle ragioni che hanno indotto il filologo a dare come testo una lezione piuttosto che un'altra' (Cozzo 2006, 256).

1.3. Purpose of Collation

not necessarily provide a complete account of the ‘constitution of the reading text’, and are not always explicit about the editor’s choices. Even when the editor inserts comments, the cramped format of the apparatus does not allow for detailed explanations. In addition, those explanations are sometimes ambiguous and susceptible to being misunderstood (see for instance Winterbottom 1995, 41). Although not all fulfilled, the four objectives listed by Vanhoutte represent what scholars desire from a record of variants. These objectives are also useful to determine the purpose of collation, what scholars want to achieve with collation: a research database documenting versions of a text.

Collation provides data which are used to understand the articulation of the manuscript tradition (Stussi 1994, 124) and determine the witnesses’ genealogical relationships, i.e., the *stemma codicum* (Driscoll 2000, 86). In fact, the simple act of collating, reading the manuscripts one by one, may be helpful to build a mental stemma and develop a sense of the manuscript tradition. West (1973, 65) recommends not to ‘put off the question of the interrelationships of the manuscripts till [the investigator] has finished collating them, forming and modifying hypotheses all the time’. De Strycker (1975, 359) reveals that after a certain number of collated manuscripts, he was often able to situate a new manuscript precisely in the stemma within a few minutes¹². Collation and stemmatology are so closely associated that the difference is not always clearly expressed. Kline (1998, 270), for instance, defines collation as the ‘process by which editors isolate patterns of error that indicate transcriptional descent, to determine whether one or more texts is a copy made from an earlier and thus more reliable copy’, thereby blending the function of the collation and of the stemma constitution. The stemma then leads the editor to the *eliminatio codicum descriptorum*, the decision to disregard manuscripts which are not considered useful for the constitution of the text, because they derive from preserved witnesses. Flores (1998, 43-44) underlines the influence of collation in this regard; a good collation is necessary to avoid eliminating an important witness.

Collation also provides the list of readings to be analysed during the constitution of the text. For this reason, Lendle (1968, 4) advocates two collations: first a collation of short samples (or *loci critici*) to determine the stemma and choose the useful manuscripts, and a second complete collation of the chosen witnesses. Recording differences — or variants — should not only serve to determine as closely as possible the original state of the text, along with its history and transmission, but also

¹²‘Assez rapidement on acquerra en ce genre une sagacité basée davantage sur l’expérience et sur la connaissance des variantes que sur des règles théoriques. Arrivé au quarantième manuscrit du Protévangile, je pouvais presque toujours décider en moins d’un quart d’heure à quelle famille, groupe et sous-groupe un nouveau témoin appartenait’ (De Strycker 1975, 359).

ideally provide a justification, in the form of an apparatus, for the editor's choices to be evaluated by future scholars and readers.

In summary, the main goals of collation are to understand the manuscript tradition and visualise it in a stemma, establish a critical text and a critical apparatus (Macé et al. 2015, 332). These three elements, stemma, text and apparatus, are all part of the critical edition which would be the final purpose of the editor. However, scholars may decide to undertake a collation for other purposes, for example to study and reconstruct the manuscript tradition. Or again, a collation may also be taken on in order to check another scholar's conclusions, the validity of their procedures and the reliability of their work (Trovato 2014, 278; Dawe 1964, 16-17, note). In this latter case, Trovato and Dawe both advise to collate in addition a random selection of manuscripts which were not collated by the scholar whose work is under scrutiny, in order to ensure that they have correctly analysed the result of their collation and that they have not discarded good manuscripts¹³.

1.4 Issues of Collation

The issues of collation are admittedly neglected in modern classical scholarship, whether because they are considered too obvious to be discussed (Whittaker 1991, 121; Robinson 2004), or because collation is unjustly not regarded as a genuinely scholarly activity but an inevitable and tedious preliminary to textual criticism which 'truly begins only once all variants have been gathered' (Shillingsburg 1996, 140). Froger (1968, 230) also compares collation to accounting ('comptabilité'), easily left to a machine acting like a secretary; Lendle (1968, 53) refers to a 'technical, philologically sterile' work; Shillingsburg (1996, 134) again describes collation as 'idiot work' because it has 'a tendency to render workers raving madmen'. However, it must be strongly stressed that collation is already a critical activity, as it involves subjective choices from the editor. Decisions whether or not to collate a manuscript, whether or not to record a difference, will affect the establishment of the text and are an integral part of textual criticism. Since collation takes place at the very beginning of critical editing, its importance must not be underestimated. An incomplete or unclear collation will affect the quality of the final edition: 'the application of stemmatic theory, the elimination of *codices descripti*, the classification of variants,

¹³'[I]f, for example, one wished to to follow up Turyn's researches on the manuscript tradition of Sophocles or Euripides, it would not be enough to collate the manuscripts which he thinks important, and then see if the results support his stemmatic conclusions. One would have in addition to collate a random selection of manuscripts which he did not think important, and then see if their exclusion was justified. If one wishes to check the validity of a scholar's conclusions in a particular field of research, it is not enough to go only to that particular corner of the field where the scholar tells us all the rich ground lies' (Dawe 1964, 16-17, note).

the whole panoply of textual criticism, stand or fall according to the quality of the collations upon which they build' (Whittaker 1991, 121).

For these reasons, some scholars have given advice on how to accomplish an effective collation. The guiding instructions can be divided into two categories: first, the issue of how to collate, how to increase accuracy and how to represent the variants in a useful format; second, what is to be included in the collation, that is, what is a variant. Unfortunately, practical recommendations are mostly driven by the medium, the layout and formatting, instead of focusing on a scholarly method. The techniques described are heavily dependant on practical exigencies inherent to a hand-made collation on paper¹⁴. As a consequence, such instructions are becoming less and less relevant with the adoption of computers in the scholarly editing workflow. However, Parker (2008) and Macé et al. (2015) for instance also include instructions for collating in electronic files. In fact for New Testament studies, paper collation is not the preferred method anymore, it has now been superseded by an electronic transcription which can be collated automatically Parker (2008, 95). A survey of the scholarly literature on manual collations is still relevant to this dissertation, because it provides the necessary context to understand how collation has changed with the advent of automated collation tools. In addition, manual collation practices reveal methodological concerns which still need to be addressed in computational methods, such as the content or visualisation of the collation results.

Paper used to be omnipresent during collation, whether it is the paper of the base text, which should be clearly printed and wide-spaced for legibility (Whittaker 1991, 122), or the paper where the collation is written down. If the collation is to be handwritten within an existing printed edition, such edition should have spacious margins and ink-proof paper for Stählin (1914, 31), who describes a procedure to prevent paper from blotting by dipping it into a mixture of glue and water¹⁵. If the collation is to be written on paper sheets, the recommendations of De Strycker will help scholars organise their collation to the millimeter: 'In-quarto squared paper will be used (27 x 21 cm) with squares of 5 x 5 mm. [...] The variants are left aligned one square away from the line number' (De Strycker 1975, 351)¹⁶.

¹⁴'La technique que je vais décrire trouve son origine, tout comme celle de M. Lendle, dans des exigences pratiques' (De Strycker 1975, 346).

¹⁵'Aber nicht alle Drucke eignen sich für das Eintragen von Varianten. Ältere Ausgaben sind oft auf Löschpapier gedruckt, auf dem man nicht mit Tinte schreiben kann. Ein derartiges Papier kann aber der Buchbinden durch ein einfaches Verfahren zum Schreiben brauchbar machen; er taucht es in Leimwasser und glättet es nachher; dann wird es tintenfest' (Stählin 1914, 31).

¹⁶'On utilisera du papier quadrillé de format in-quarto (27 x 21 cm) avec des carreaux de 5 x 5 mm.

Today, few scholars are likely to worry about rain washing away their collation, which seems otherwise a very compelling worry for West (1973, 66) who cautions that ‘collation should always be in ink. If washable ink is used, beware of rain’, or coffee ... Nowadays they worry about keeping backups on different devices and synchronising them via cloud storage solutions, for instance especially in the context of collaborative work (see Pierazzo 2008, §41; Chaudhuri 2015, 60).

The second aspect concerns more theoretical issues: what to collate? How to choose the witnesses of a text, how to decide what is a variant worth being recorded? Here also, the influence of print and a paper-based workflow affects the editor: ‘a complete critical edition [...] should keep from the manuscript tradition everything that can be printed and noted with marks and characters’ (Dain 1964, 175)¹⁷. Despite being more relevant to develop a scholarly collation method, the choice of what to include in the collation is often left to the appreciation of the editor, since it ‘depends on the importance of each manuscript and the purpose of the collation’ (Stählin 1914, 33) and the purpose of the edition prepared¹⁸. As a result, a variant is rarely defined by the scholars who describe a collation technique, leaving disconnected how and what to collate. The following sections contain an overview of the scholarly recommendations and opinions about the practical aspects of collation.

Most sources quoted in the sections below are manuals with advice on how to edit Greek or Latin texts from classical antiquity (Stählin 1914; Willis 1972; West 1973; Chiesa 2002). Stussi (1994) presents an introduction to Italian philology which covers not only antiquity but also later periods. Whittaker (1991) gives general advice for manual collation based on his experience as an editor of classical texts. Other editors explain the method they have followed for a particular edition: Lendle (1968) provides an account of editing Gregorius Nyssenus’ *Encomium in Sanctum Stephanum Protomartyrem*¹⁹. De Strycker (1975), inspired by Lendle, shares the method he devised for collating a Greek hagiography with rich manuscript tradition. Although Froger (1968) introduces automated collation, the first part of

[...] Les variantes s’alignent sur la gauche à un carré de distance du chiffre désignant la ligne’ (De Strycker 1975, 351).

¹⁷‘En réalité, la proportion de détails qu’on doit relever dans une collation de manuscrits varie avec le type d’édition auquel on la destine. Une édition critique complète, et voulant être définitive, devrait garder de la tradition manuscrite tout ce qui peut s’imprimer et être noté par des signes et caractères’ (Dain 1964, 175).

¹⁸‘Eine allgemeine Regel lässt sich aber auch hier nicht geben; wieviel in der Kollation berücksichtigt werden muss, hängt von der Bedeutung der Handschrift und dem Zweck der Kollation ab’ (Stählin 1914, 33).

¹⁹Gregorius Nyssenus (Gregory, bishop of Nyssa c. 335 – c. 395) was a theologian of the fourth century AD and one of the Fathers of the Eastern Church.

his monograph is an initiation to textual criticism, aimed especially at scholars from a scientific background who are not familiar with the discipline. In addition, other domains of textual criticism are represented, such as New Testament studies (Parker 2008), oriental manuscript studies (Macé et al. 2015) or medieval literature (Bourgain and Vielliard 2002).

1.4.1 How to Collate?

1.4.1.1 Choice of a Base Text

The first step is to choose a single base text (also called reference text or collation exemplar) against which every other witnesses will be collated. The base text must remain the same throughout the entire collation, otherwise the comparison of every witnesses with each other is in practice impossible. In principle, any manuscript or edition arbitrarily chosen could serve as a base text (Froger 1968, 81)²⁰. Although Chiesa (2002, 49) does not pronounce himself on a general rule, he recognises that convenience is most important²¹. Hence a good base text will maximise the collator's physical comfort and minimise the amount of variants to record, in order to prevent errors during the collation.

To limit the strain on the eyes already solicited by a very meticulous attention to detail, a base text should be above all clearly printed and easy to read. If necessary, it should even be typed again in 'the largest and clearest script available' (Whittaker 1991, 122). This advice applies only for a paper-based workflow, but is unlikely to be relevant to present concerns when changing font-size or layout in electronic texts takes only a matter of seconds.

In addition, Pasquali (1952, 62), West (1973, 66), Whittaker (1991, 122) and Bourgain and Vielliard (2002, 49) all underline the necessity for the base text to differ as little as possible from the other witnesses. Indeed, experience teaches us that the higher the number of variants to be recorded, the more likely some of them will escape the collator's attention (Pasquali 1952, 62)²². In this condition, a critical edition is not always the most practical choice for a base text since it will contain emendations and conjectures, despite Stählin's advice to select 'the best text available' (Stählin

²⁰'Bien que dans un problème réel on choisisse comme exemplaire de référence un manuscrit commode, bien lisible, sans lacunes, etc., on peut en principe prendre n'importe lequel, arbitrairement' (Froger 1968, 81).

²¹'Una regola generale non si può dare, se non quella della comodità: l'esemplare di collazione dev'essere quello che a priori sembra permettere all'editore di lavorare meglio' (Chiesa 2002, 49).

²²'Quante meno differenze presenta il codice collazionato rispetto all'esemplare di collazione, tanto più probabile è che la collazione sia giusta e completa, come al contrario quante più discrepanze si devono segnare, tanto più facile è che qualcuna ne sfugga' (Pasquali 1952, 62).

1914, 33). On the other hand, a critical edition would likely have the advantage to be available in digital format or to be easily scannable (Macé et al. 2015, 332). The lack of space on the margins is an argument against collating directly inside a printed edition (West 1973, 66). The scholar who would nevertheless collate in an edition may have then to worry about the quality of the edition's paper and size of the margins (Stählin 1914, 31). Chiesa (2002, 49-50) underlines the psychological influence of the base text: if the base text is a manuscript or unedited text, the editor may tend to overvalue its readings; on the other hand, the editor may underestimate an edition, even unconsciously, in order to distance themselves from the work of another scholar²³. In the end, it seems, the choice of the base text will have a fairly limited impact on the result of the collation (Macé et al. 2015, 332).

1.4.1.2 Material Support of the Sources

Before the advent of photographic or microfilm reproductions, scholars were forced to travel across Europe in order to access and read manuscripts held in libraries. Due to the costs in time and funding of such travels, it was difficult to consult more than a handful manuscripts when preparing an edition, with the temptation to choose manuscripts based on their accessibility rather than their value (Stählin 1914, 29). Thanks to photographs and microfilms, much more material became available to the editor, for a more reasonable price. Although a facsimile cannot replace the original, it offers advantages such as independence from libraries' schedules, easy access conditions, as well as the opportunity to check the photograph or microfilm whenever a doubt arises. Parker (2008, 89) notes that 'it is reasonable to use a surrogate as much as possible'.

To consult microfilms a specialised equipment is needed. Working with microfilm-readers is not convenient, but tiring and uncomfortable, as amply discussed. Bülow-Jacobsen (1979) provides a review of the shortcomings that affect the quality of medieval manuscript microfilms; Macé et al. (2015, 330) also lists the disadvantages of other reproductions made from those microfilms. Microfilms are more difficult to compare to each other or annotate. 'Occasionally, things become clearer with photographs' (West 1973, 65) since they can be held against the light to examine the places where ink has faded away in order to accentuate the contrast between

²³Il vero problema nell'uso dell'esemplare di collazione è di ordine psicologico; se l'edizione cui si sta attendendo riguarda un testo fino a quel momento inedito, lo studioso sarà poi facilmente portato a pensare che il suo testo-base (che sarà inevitabilmente la trascrizione di un codice) sia per qualche ragione più importante, perché è la norma che viene continuamente confrontata; se vice-versa si tratta di un testo già pubblicato, e il testo-base è una precedente edizione, lo studioso può essere portato a pensare che essa sbagli più di quanto effettivamente non faccia, perché vuole, in qualche misura, prenderne le distanze' (Chiesa 2002, 49-50).

ink and paper (De Strycker 1975, 348)²⁴. Although black and white reproductions will not show differences in ink as well as the original, other features such as handwriting, erasures or corrections are easy to discern (Harkins 1958, 163). Similarly to microfilms, photographic reproductions can also suffer from poor quality. Lendle (1968, 99) for example admits that for some manuscripts, photographs were incomplete. After collating with photographic copies, it is advised to carry out an examination of the manuscripts in the libraries, the ‘autopsy’, to clarify passages of uncertainty. Those passages may contain in particular damaged folia, a text with many corrections, or involving more than one copyist. Other codicological features of the manuscript are absent from reproductions, such as the gatherings organisation, bindings, etc. It is thus methodologically unsound to rely solely on facsimiles, regardless of their quality.

Despite Dain’s claim that the technical material had reached perfection with photographs, and that everything possible had been implemented (Dain 1964, 178)²⁵, digital images constitute a significant progress. The price of production is considerably lower, although storage costs to maintain digital facsimiles accessible online are superior to microfilms storage costs. Digital copies provide colour images which can be zoomed in or out at will, and be manipulated on the computer in order to emphasise some aspects or render erased letters legible again. For instance, palimpsests such as the Archimedes Palimpsest would not be readable without digital image processing²⁶. The processing techniques of manuscript restoration can depend heavily on editorial decisions and should be documented to avoid misleading results (Craig-McFeely 2008)²⁷. Flüeler and Porter (2015) argues that digital manuscripts are more than a simple reproduction: digital manuscripts should be conceived as critical editions that can be enriched with high-quality images, metadata, text transcriptions, and links to other manuscripts for a broader context.

²⁴‘Si l’on se sert de photographies, on constatera qu’il y a avantage à les regarder par transparence aux endroits où l’encre est assez effacée’ (De Strycker 1975, 348).

²⁵‘[L]a technique de l’édition, en dépit des erreurs encore commises de nos jours, est arrivée à son point de perfection. Je ne parle ici que de ce qui touche l’usage des manuscrits. Et d’abord, notons le degré de perfection du matériel technique : reproductions photographiques sur films ou rotographs, usage de la photocopie, lecture aux rayons ultra-violets, tout a été mis en oeuvre’ (Dain 1964, 178).

²⁶‘What you see when you open the Archimedes palimpsest therefore, is not a mathematical text, nor even a piece of Greek oratory, but a prayer book. Only occasionally can one just discern, at right angles to the prayer book text, the erased writings that the current project is attempting to recover.’ <http://archimedespalimpsest.org/about/> (accessed August 5, 2015).

²⁷For considerations on quality issues with digital facsimiles, see Craig-McFeely (2008).

1.4.1.3 Comparison Process

Collation demands a sharp focus on details, a focus which can easily be lost in the constant gaze switching between the base text and the collated exemplar. As Pasquali (1952) points out, when reading first the base text edition and then switching to the manuscripts, there is a risk to replace unconsciously the manuscript reading with the one of the base text and thus miss a variant:

We too modern philologists, when we collate a ms. with a printed edition, we usually read every sentence first in the edition and then in the ms. We do not do the opposite, because the edition proves to be easier, and yet we should realise that this is the source of an infinite number of errors: having the printed text in mind, or so to speak in the eyes, we often unconsciously substitute it to the text of the manuscript (Pasquali 1952, 63)²⁸.

Willis (1972, 32) recommends to proceed phrase by phrase or sentence by sentence, ideally with an assistant reading the reference text aloud for an increased accuracy. While the collator may not have an assistant at their disposal, reading aloud is an effective way of improving concentration and therefore reducing errors. The 'bisensory method', combining sight and hearing, was already described by Harkins (1958) who worked with classical and patristic authors. Harkins (1958) recorded himself reading the base text on a tape, to avoid the issues of shifting gaze from one text to the other: loss of time, loss of accuracy and loss of his place in the text. The technique might involve the risk that, while listening to the tape, the collator passes through difficult passages hastily and misses important details (Lendle 1968, 50). Furthermore, Harkins' method does not take in account spelling variation which cannot be distinguished by pronunciation. When a kind of spelling error has been safely identified as a typical spelling for a group of manuscripts, Harkins (1958, 165) suggests that this can be safely ignored. In this case, orthographic spelling would still need to be collated for a portion of text, until the editor can decide that a spelling error is consistent. During collation, it may be indeed helpful to read aloud the manuscript while silently surveying the print edition. The production effort of reading aloud facilitates the memorisation of words compared to silent reading. It

²⁸'Anche noi filologi moderni, quando collazionamo un ms. con una stampa, sogliamo leggere ogni proposizione prima nella stampa e poi nel ms., e non viceversa, perchè la stampa riesce pur sempre più agevole: eppure dovremmo accorgerci che questa è la fonte di un'infinità di errori, perchè noi, avendo il testo della stampa in mente e si potrebbe dire nelli occhi, lo sostituiamo spesso inconsciamente a quello del ms' (Pasquali 1952, 63).

seems that reading aloud has two positive effects: we read better (making fewer mistakes), and memorise better²⁹. Keeping an index finger on both texts is another way to track the position and avoid missing a word or a line while shifting the gaze back and forth between the manuscript and the base text: this method is referred to as the ‘Wimbledon method’ by a few scholars (see Smith 2000, 131). While this method may be applicable to modern materials, it may not be appropriate in all circumstances and would not be allowed in manuscripts reading rooms.

The order in which witnesses are collated is in theory immaterial. However, starting with the oldest witnesses would be helpful to sort out a rich manuscript tradition and facilitate the *eliminatio codicum descriptorum*, for every later witness is usually related to a more ancient manuscript. It may also be worthwhile to take into account the resemblance between witnesses and collate successively the witnesses which are most similar (De Strycker 1975, 349-350). It is worth noting here that, while maybe trivial for a handmade collation, the order of witnesses can significantly influence the results of a computer-supported collation, as it will be seen in Section 2.2.

1.4.1.4 Recording and Visualising Variation

Recording and representing the collation results in a convenient format is a challenging task. Great care must be taken to avoid ambiguity in the collation. The base text must be clearly indicated and lines numbered; it is important to keep track of the manuscripts foliation in order to recover easily a reading and check the collation accuracy. Stählin (1914, 30) suggests recording the collation in the base text edition with the use of different coloured inks to distinguish different manuscripts and avoid confusion. He is nevertheless aware of the method’s shortcomings, i.e. that it is efficient only up to six manuscripts collated³⁰. Collating directly in the edition is not a good solution due to the lack of space (West 1973, 66), and an interleaved copy annotated with coloured inks will be illegible in the end, having ‘transmogrified itself into a baffling multicoloured jungle’ (Whittaker 1991, 124).

The alternative is to write down the collation on separate sheets of paper or notebooks. Lendle (1968, 53) applies a system of table where the rows record variant readings, and each column represent the base text and the manuscripts. At each point of variation in the text, the concurrent variant readings are recorded on the

²⁹For more information on the benefit of reading aloud, see for instance Forrin, Jonker, and MacLeod (2014).

³⁰For more than 6 witnesses, it is necessary to collate in several copies of the edition, and to gather the variant readings after collation is finished.

	v	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
6 ἐνδυσόμενος		×	×	×		×	×	×	×	×							×
ἐνδυσάμενος	×				×						×	×	×	×	×		×
7/8 τοῦ βίου τὸ σπῆλαιον		×					×					×	×				
τὸ τοῦ βίου σπῆλαιον	×		×	×	×	×		×	×	×	×			×	×	×	×

Figure 1.1: Lendle's notation method of collation with crosses (Lendle 1968, 53).

left with an identification number, and Lendle adds a cross in a manuscript column where the corresponding variant appears (see figure 1.1). The purpose of Lendle's method is to visualise easily the witness relationships. The combinations of crosses may show groups of witnesses which have variant readings in common against other witnesses, which is important for editors following Lachmann's method (see Chapter 8). However, the cross system is prone to confusion and space-consuming: one variant appearing in only one manuscript occupies an entire line. Moreover, a cross is not enough: the collation often requires a careful description and clarification of the manuscript reading (Whittaker 1991).

Very similar to Lendle's notation system, Trovato (2014) describes a collation of Dante recorded in Microsoft Excel, where crosses are replaced by zeros and ones. In figure 1.2, variant locations are given a reference number in the first column, and the variants are recorded in column C: the first readings highlighted in a darker background are the readings from the base text. The next columns represent the witnesses, which receive a number '1' if the witness has the same reading as the base text. If the witness' reading is different from the base text, a zero is noted for the base text. As for Lendle's method, Trovato's visualisation aims at helping the editor to find the genealogical relationships of witnesses: the same sequence of 0 and 1 in different columns indicate that the witnesses of these columns are related (Trovato 2014, 244).

De Strycker (1975, 351) devised a different notation method, in which he gives a number to each witness (preferably the number representing their order of collation). He indicates for each variant the numbers of the witnesses presenting the variant, and distinguished different notation systems, positive or negative, according to the kind of variants (De Strycker 1975, 352-357)³¹. De Strycker (1975, 358) also suggests to establish concordances of agreements and disagreements between presumably related manuscripts, in parallel to the collation.

³¹'La pratique de la collation fait voir assez rapidement que la méthode positive de notation des sigles n'est pas indiquée au même titre pour toutes les variantes' (De Strycker 1975, 355).

1.4. Issues of Collation

A	B	C	D	E	F	G	H
			U (366)	E (= Est. 474)	F	Ub (365)	Pad. 67
1.21.113	WP	mille dugento con sessanta sei	1	1	1	1	1
		m. d. un con sessanta sei					
		m. d. con settanta et sei					
		m. d. con cinquanta sei					
1.22.6	WP	fedir torneamenti e correr giostra	1	1	1	1	§
		fedir tornegiamenti e correr giostra					
		ferir di tormamente accorrer g.					
		far di torneamenti					
		f.t. e muover giostra					
1.22.58	WP	Tra male gatte era venuto 'l sorco	0	1	0	0	§
		Tra male gatte era arrivato 'l sorco					
		tra male gatte era abattuto il s.					
		tra male gente					
		tra male branche	1		1	1	

Figure 1.2: Manual collation recorded in a Microsoft Excel spreadsheet (Trovato 2014, 245).

Froger (1968, 82) adopts a collation in form of a table: in the first two columns are recorded the base text reading and the list of witnesses which present the reading. The next pair of columns display respectively a variant reading and the witnesses attesting it. Stussi (1994, 123) prefers, on the contrary, to write on the first line of a sheet a segment of the base text, and record variant readings of the other witnesses each on a different line below the base text. When the correspondence between the base text and a witness is perfect, the line is left blank. However, this method has the disadvantage that it would be difficult to know in advance if enough space is left for additions in the witnesses, which are longer than the base text. Bourgain and Vielliard (2002, 49) note that this method is only convenient for up to six or seven witnesses. Bourgain and Vielliard (2002) also propose to record variants as footnotes in a digital document, to imitate a traditional critical apparatus. The delimitation of a variant location is the main issue of this method: it is difficult to know where a variant will end, before all the witnesses are collated. In addition, the apparatus format is not the most practical to study the textual tradition. For that purpose, Bourgain and Vielliard (2002) prefer a list of variants, such as the example above from Trovato (figure 1.2). For the Greek New Testament, on the other hand, Parker (2008, 96) describes a collation in apparatus in this form: ‘reading of base text| reading of witness’. Parker (2008) advises to keep the variation units as short as possible in the collation.

The structure recommended by Macé et al. (2015, 332-333) is very similar to Stussi’s,

1.4. Issues of Collation

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
70	68 ruses														
71	69 de		des	des	des	des	ok	des	des	des	des	ok	ok	des	
72	70 Sioux														
73	71 du														
74	72 rationaliste		rationaliste				rationalis	rationalis	rationalis	ok	ok	ok	rationalis	ok	
75	73 ou														
76	74 la														
77	75 candeur														
78	76 ardente														
79	77 de														
80	78 l'athée.														
81	79 Je														
82	80 n'ose														
83	81 donc														
84	82 jeter														
85	83 la														
86	84 pierre														
87	85 ni														
88	86 à														
89	87 celle			celui	clie*	celui				celui	celui			celui	
90	88 qui														
91	89 croit									croient		p.corr.; crois a.corr., alia manu			
92	90 en		om.	om.	om.	om.	om.	om.	om.	om.	om.	ok	om.	om.	
93	91 des														
94	92 choses														
95	93 qui														
96	94 ne														
97	95 m'inspirent						m'inspire						p.corr.; m'inspire a.corr.; alia manu		
98	96 que														
99	97 le														
100	98 doute,		ok					doute	doute					doute	
101	99 ni														
102	100 à														
103	101 celui														
104	102 celui														

Figure 1.3: Manual collation recorded in a Microsoft Excel spreadsheet (Macé et al. 2015, 333).

even if updated by the use of the computer: the base text is added word-by-word in the first column of a MS Excel spreadsheet, and each other column contains the variants of one witness with a blank space where the witness is the same as the base text (see figure 1.3).

Willis (1972, 33-34) deals with the space issue with record cards. Each card records all the variants from the base text at one location in the text. The location would be indicated either with page and line numbers, or verse number. In the top-right corner the witnesses which have the same text as the collation exemplar. The result is a positive account of all variants, which eliminates hesitations arising from the absence of a manuscript's siglum in the collation: was the reading altogether absent from the manuscript, did the reading coincide with the base text, or did the collator miss a variant in a moment of inattention? With Willis' cards system, there should be no doubt when a witness' text is the same as the reference text. Willis considered it the best and only way to significantly lessen collation errors while improving precision.

The problem of space is common to all the above methods that record collation on paper. This problem is often difficult to manage, but has become irrelevant to collations now recorded in electronic format, since new material can be added at any point without worrying about leaving blank spaces. Another way to save space

and help with the collation is to give each manuscript a siglum, either following established conventions (see Macé et al. 2015, 331) or using numbers (De Strycker 1975, 350-351).

Whittaker (1991, 126-128) criticises the above methods, regretting that individual characters of scribes and manuscripts are getting lost in the process:

One fundamental misconception underlies all these methods, namely their common assumption that the collation of a plurality of manuscripts is nothing more than the accumulation of all their variant readings upon the same sheet of paper. This assumption is based in turn upon the illusory expectation that readings so accumulated will constitute a synoptic vision of the evidence — a sort of *totum simul* in which, like God, we will see and embrace in the twinkling of an eye all the ramifications of a complex textual tradition.

Whittaker rather collates each manuscript in a separate notebook and correlates his results through an inventory of omissions completed by an electronic register of variants that will help to amend the errors inevitably present in the collation, and a profile of each copyist. The inventory of omissions ‘will provide the skeletal outline of one’s stemma. In this process minor omission may well turn out to be the most significant, since they are the most likely category of error to have escaped correction through contamination or scribal conjecture’ (Whittaker 1991, 128). Whittaker’s method has a considerable amount of redundancy, with the same information appearing both in notebooks and electronic inventories. The same criticism applies to De Strycker’s concordances.

1.4.2 *What to Collate?*

The advice from most sources presented here on collation methods is centred on the procedure rather than the content. The choice of a base text and the form collation should take — the ordering of variants on a paper sheet, a notebook, cards, or an excel spreadsheet — seem to be the major concerns expressed. More precisions can be found with the COMST handbook (Macé et al. 2015), with Parker (2008) and Bourgain and Viellard (2002).

Macé et al. (2015, 331) assert that collation should contain ‘not only all variations, including orthographical ones, but also punctuation, abbreviations (and numbers)’. However, when editing the *Florilegium Coislinianum*, ‘punctuation and purely

phonetic differences of no morphological significance' were disregarded (Macé, De Vos, and Geuten 2012, 113). Bourgain and Vielliard (2002, 50) list five items which should appear in the content of a collation: (1) titles, chapters and subdivisions of the text, which will be useful to classify the witnesses; (2) variant readings, i.e., additions, omissions and transpositions; (3) the different states of the text, that is readings before and after correction by different scribes (exceptions are made for corrections made by the copyist while writing); (4) punctuation, at least if it indicates a different interpretation; (5) graphical variants in some cases (see discussion p. 42 below).

Parker (2008, 96-97) recommends to decide before collating what kind of differences will be recorded, and to remain consistent. The collator should record corrections from the first or later hands, lacunae, damages to the document. Superscriptions, subscriptions and colophons must also be recorded. Among the differences that might not be worth writing down, Parker lists punctuation, accentuation, itacisms and other spelling variations, movable *nu*, various abbreviations, and diaeresis. However, this list is not fixed, it will depend on the context: Greek accents may be useful when editing Byzantine texts but not in the case of the New Testament.

Stählin (1914, 33) advises to record every minor detail ("Kleinigkeiten") in order to establish relationships between witnesses, because it is impossible to know in advance what will prove significant. These details cover orthographical variants, accents and breathings, and even interpunction in some circumstances. However, he quickly admits that there can be no general rule. What to include in the collation will depend on the manuscript value and the purpose of the collation, without explicitly giving examples of other purposes³².

De Strycker reflects on when to record *orthographica*, orthographic differences, accents etc. He does not give any fixed rule but to rely on experience in order to learn 'which orthographic irregularities deserve to be considered as variants' (De Strycker 1975, 364)³³. Stussi (1994, 124) gives an example of collation with ortho-

³²'Bei der Kollation selbst sind in der Regel auch Orthographica, Spiritus, Akzente u. dgl. (unter Umständen auch die Interpunktion) zu notieren, obwohl sie nicht in den Apparat kommen, weil sie für die Erklärung von Korruptelen oder für das Erkennen des Handschriftenverhältnissen wichtig sein können. Eine allgemeine Regel lässt sich aber auch hier nicht geben; wieviel in der Kollation berücksichtigt werden muss, hängt von der Bedeutung der Handschrift und dem Zweck der Kollation ab. Nimmt man von einer Handschrift eine Probe, um daran das Verhältnis anderer Handschriften festzustellen, dann wird man jede geringste Kleinigkeit verzeichnen, weil man nicht im voraus weiss, ob nicht vielleicht eine solche Kleinigkeit die Frage nach dem Verhältnis der Handschriften entscheiden kann' (Stählin 1914, 33).

³³'L'expérience seule apprendra quelles irrégularités orthographiques méritent d'être considérées

graphic differences ('graphical variants'), but without word division differences. The orthographic, phonetic or morphological variants are important to assign the right linguistic colouring to the text, except in the case of Latin Classics where the timespan between the oldest manuscripts and the author is so long that the problem is solved by standardising the graphical and phonetic aspects (Stussi 1994, 139-140).

For Willis (1972, 13), on the other hand, orthographical variation is not a necessary part of the collation: 'to collate a manuscript so that its affiliation may be determined [...] involves the recording of every variant other than the purely orthographical'. Although Dain (1964, 174-175) does not describe a collation method, he supports Willis as to the *orthographica* or form variants: 'orthographical variants, when they are indifferent, accents, punctuation, everything which is not a part of the tradition, can usually be ignored, since they are only evidence of the writing style at a particular period of time, and often in a given milieu' (Dain 1964, 175)³⁴. This does not apply for critical editions meant to be complete and definitive: in this case everything should be included in the collation (Dain 1964, 175). Abbreviations are also part of any exhaustive collation: 'I know one [student] who wished to note in his apparatus, as it is done and must be done in a complete collation, the difference between syllables fully spelled out and syllables abbreviated *per compendium* or *per arctationem*, distinguishing with different signs between the two categories' (Dain 1964, 174)³⁵.

The purpose of collation for Chiesa (2002, 48) is to gather differences, but his discussion on collation does not refer to its actual content and is not explicitly connected to the other aspects of textual criticism. Whittaker only gives untheoretical suggestions for a successful collation, but describes the collator's work as recording 'each instance in which his manuscripts diverge from the base text' (Whittaker 1991, 122). Finally, Lendle seems not to be interested in the definition of difference or variants, but in the practical aspect of how to write down differences on paper in the most helpful way: 'the technical process can be divided in two phases, first the

comme des variantes' (De Strycker 1975, 364).

³⁴ 'Donc, sauf dans les éditions de type spécial, la règle sera de relever comme variantes de manuscrit les leçons de texte seulement, et non la manière dont on écrit ces leçons. Notamment, les variantes d'orthographe, quand elles sont indifférentes, l'accentuation, la ponctuation, toutes choses qui ne font pas partie de la tradition, pourront d'ordinaire être négligées, comme ne marquant qu'une manière d'écrire à une certaine époque, et souvent dans un milieu donné' (Dain 1964, 175).

³⁵ 'J'en sais un qui voulait noter dans son appareil critique, comme on le fait et doit le faire dans une collation complète, la différence entre syllabes écrites en toutes lettres et celles écrites *per compendium* ou *per arctationem* en distinguant d'ailleurs par des signes différents ces deux catégories d'abréviations' (Dain 1964, 174).

actual comparison of manuscripts with the base text, and then the recording of the differences thus identified along with other observations. [...] the second phase leads us to the question of the most appropriate notation method' (Lendle 1968, 51)³⁶.

In summary, there are no fixed rules (Stählin 1914, 33; De Strycker 1975, 363), but the question of the *orthographica* divides opinion. The *orthographica* include abbreviations, punctuation, as well as accents and breathings in Greek manuscripts. They are absolutely necessary for Macé et al. (2015) and Stählin (1914) who argue that anything can become a significant variant in order to build a stemma. On the other hand *orthographica* are useless to Willis (1972). Stussi (1994) offers a more nuanced judgement, concluding that they are not important in the case of Classical languages because of the time-lapse between the author and the first manuscripts. They cannot help characterise the author's language but rather the scribe's language (Dain 1964, 174-175). Bourgain and Vielliard (2002) also state that Latin spelling was more unified than for romance languages, and regional or chronological variants less likely, because Latin was taught in school. They argue that for this reason, Latin spelling variants have only a limited linguistic interest, and cannot be used to reconstruct the original (Bourgain and Vielliard 2002, 50). However, it is also true that an orthographic difference, if misunderstood by a later copyist, can lead to a real textual variant that changes the meaning of the text. In addition, the difference between a textual variant and a mere spelling difference may not be evident at first sight. As a result, it is better to be cautious and record as much as possible in the collation, and sort out the variants at a later stage.

There seems also to be an agreement that 'what to collate' can vary, depending on the purpose of the collation or edition (Stählin 1914; Dain 1964). Moreover, the field of study can influence collation and its content: 'the notion of "variant" changes according to the academic discipline within which it is being defined' (Winters 1991, 133). Thus, the term 'variant' needs to be defined precisely and will be discussed further on in the dissertation (see Chapter 4). The next chapter will introduce automated collation, and how it differs from the manual collation process described in this chapter. Naturally, many concerns related to 'how to collate' do not apply anymore in the context of automated collation, for instance regarding the comparison process or the notation method of the variants on paper.

³⁶'Mann kann [der technische Ablauf] ebenfalls in zwei Phasen untergliedern, das eigentliche Kollationieren, d.i. Vergleiche der Handschriften mit dem Kollationsexemplar, und das Notieren der dabei ermittelten Abweichungen und sonstigen Beobachtungen. Während die erste Phase im ganzen als unproblematisch gelten darf, führt die zweite (jedenfalls bei vielen Zeugen) notwendig zu der Frage nach der zweckmässigsten Notierungsmethode' (Lendle 1968, 51).

1.4. Issues of Collation

On the other hand, the choice of a base text will be discussed as one of the major differences between the two methods, and further issues will be raised, such as the need for the transcription of witnesses.

The Theory of Automated Collation

2

AUTOMATED collation involves some form of help, from a machine or a computer, to the person who is collating a text. Mechanical devices have been developed since the 1940s to facilitate collation, and the name and definition of the practice have evolved as researchers worked on collation and on its automatisation. The progress of automated collation may have in turn influenced the definitions of collation itself (see Andrews 2017). From ‘automatic collation’ to ‘computer-supported collation’, scholars became more cautious with respect to the name of the process, as they realised that collation was not a perfectly mechanical activity and that computers still needed a significant amount of manual input to guide and correct collation tools.

In this chapter, and in the thesis in general, I will use the terms ‘automated collation’, drawing attention to how much thought and effort has gone (and is still going) into how scholars automate the collation process¹. ‘Automated’ here concentrates on the automatisation effort, whether or not collation is fully automatic, and whether or not it is successful.

2.1 Automatisation of Collation

2.1.1 *Optical Collation*

The first successful documented attempt to use a machine to alleviate the bulk of collation work dates from the 1940s. Charlton Hinman is known as the inventor of the ‘Hinman Collator’, a machine which let him compare different versions of the First Folio of Shakespeare’s plays. During the printing process of Shakespeare’s

¹There is a very slight difference of nuance between automatic and automated: ‘automated’ is passive, a process which has been automated (maybe to a certain extent only) and ‘automatic’ is active, a process which works by itself (with little or no human control). Froger makes the same nuance when he says that textual criticism cannot be fully automatic but rather partially automated (‘la critique textuelle ne peut être entièrement ‘automatique’, mais seulement ‘automatisée’ de façon partielle’, Froger 1968, 218).

2.1. Automatisation of Collation

editions, many differences in punctuation or orthography would be introduced so that the first printed book would be different from the last printed book of the same edition². The Hinman Collator was designed to compare only printed books, and moreover copies printed from the same edition, with the same typesetting. The machine merges the images of two folios superimposed, via a set of mirrors. When the two images have been properly aligned by the user, a system of projected lights illuminates the folios alternatively: 'if the pages are identical, they more or less appear as one; if they are not identical, the points of difference are called to the operator's eye by appearing to dance or wiggle about' (Smith 2000, 131). The Hinman Collator would perform what was called 'mechanical collation' or 'optical collation' (Smith 2002). The former, 'mechanical collation', was not exactly appropriate, since the collation performed with this machine was more optical than mechanical. Although collation is made easier to perform with the help of a device, collation is actually still performed by a scholar who will manually align the images of the pages and locate variations in the text. For this reason, this method is not considered a part of automated collation in the context of this thesis. However, optical collation was already considered part of automated collation for Dearing (1962, 3), because it made use of a mechanical device to speed up the collation process.

Optical collation suffered from several issues (see Smith 2000, 141)³. For instance, the collator would often block the light from the projectors while getting closer and trying to spot minute differences. In addition, it could be difficult to obtain proper images, photocopies or microfilms, that would fit into the device. Images could be distorted by a curvature effect, and variations in images size would also be problematic (Guffey 1968). Similar machines were later created to compare page images of editions, such as the Lindstrand Mark I Comparator (Lindstrand 1971), Hailey's Comet or McLeod's Portable Collator (see Smith 2002). The optical collation method with analogue devices was popular until the end of the 1960s, when automated collation with digital computers and algorithms started to grow in importance and took over optical collation. However, while automated collation is a good solution for comparing medieval manuscripts, optical collation cannot be discarded completely. The preparation of accurate transcriptions, for pages with the same typesetting, is difficult enough that optical collation remains a valid option. As a matter of fact, several tools for optical collation with a computer have

²These corrections are called stop-press corrections.

³For a cheap way of addressing these issues, it is possible to train oneself to do a sight comparison, by eye-crossing (to become a 'Human Hinman', see a 2008 blog post by Wesley Raabe, where he shares Randall McLeod's method: <https://wraabe.wordpress.com/2008/05/13/how-to-be-a-human-hinman-collator/>, Accessed May 4, 2017).

2.1. Automatisation of Collation

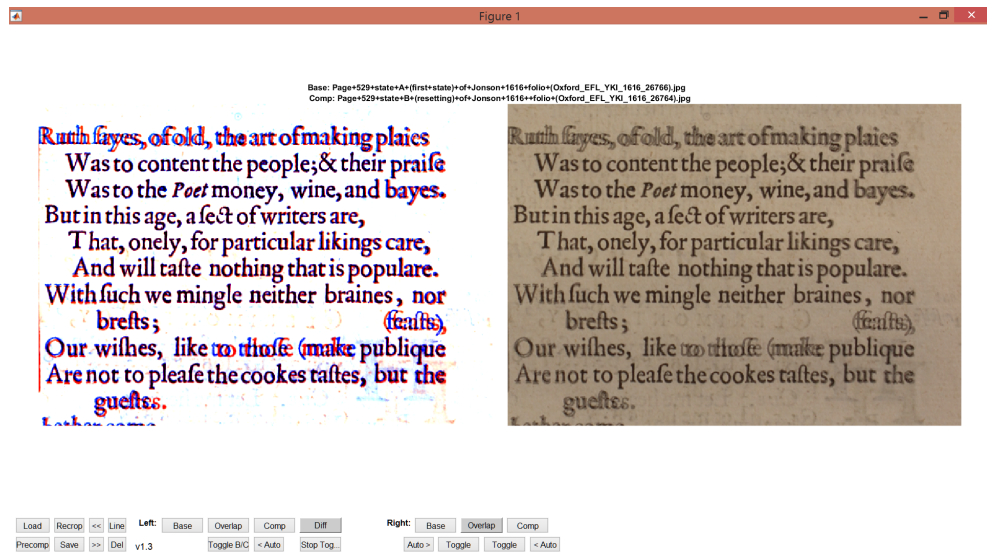


Figure 2.1: Optical text collation with the Oxford Traherne. Retrieved from <http://oxfordtraherne.org/traherne-digital-collator/> (May 5, 2017).

been recently created, such as the Oxford Traherne Digital Collator⁴, or Paragon⁵ at the University of South Carolina.

2.1.2 'Algorithmic' Collation

The difference between optical collation and the tools developed later is the use of algorithms to align texts. An algorithm is 'a process or set of rules to be followed in calculations or other problem-solving operations, especially by a computer' (Oxford Dictionary). In practice, an algorithm is a precise sequence of instructions for the computer to perform, in a specific order, until a result is achieved. Algorithms can be expressed in various notations: for example, Dearing (1970, 258) gives a collation algorithm in natural language, describing each steps of the program in a list (figure 2.2). On the other hand, Froger (1966, 153) shows the collation algorithm in a flowchart (see figure 2.3).

The alignment of texts is not done manually any more, as it was in optical collation. Instead, this task is now delegated to the computer: the algorithm is applied to electronic transcriptions until the desired result is achieved. In the case of Froger, for instance, the desired result was a list of variant readings similar to an apparatus.

⁴<http://oxfordtraherne.org/traherne-digital-collator/> (accessed May 05, 2017).

⁵<http://tundra.csd.sc.edu/paragon> (accessed May 05, 2017).

2.1. Automatisation de Collation

1. Compare the base word in the base text with the next word in the other text.
2. If they differ go to 6.
3. Compare the next word in the base text with the base word in the other text.
4. If they differ go to routine for word added in other text.
5. Go to routine for transposed words.
- ...

Figure 2.2: A collation algorithm in natural language (Dearing 1970, 258).

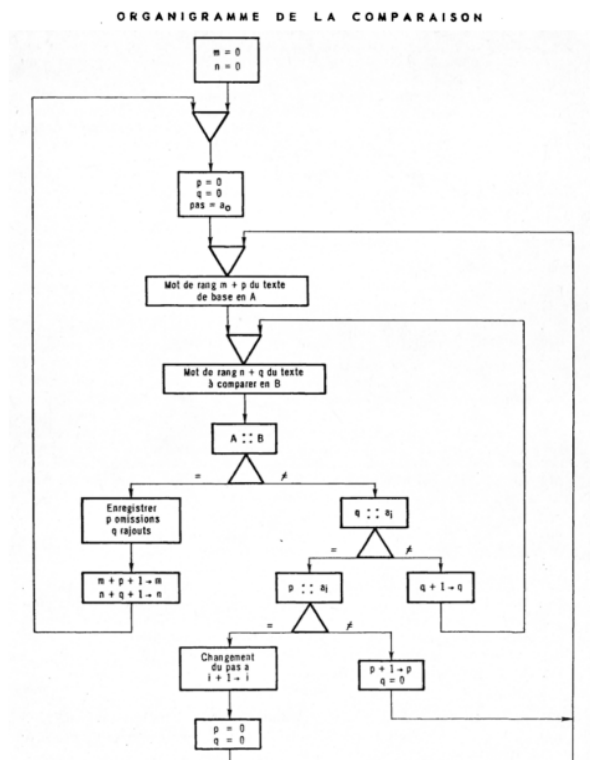


Figure 2.3: A collation algorithm in flowchart (Froger 1966, 153).

2.1. Automatisation of Collation

Computers were already used as early as the 1950s for textual criticism in biblical studies (Kraft 1995). Recording and manipulating variant readings is one of the main areas of interest, according to Kraft (1995, 268-269), along with statistical analysis of manuscripts relationships and the production of critical edition with apparatuses. It may be difficult to pinpoint the precise moment when the concept of automated collation was invented. The earliest known attempts were some versions of ‘diff’ algorithms and used word processing tools (Marín 1991). Diff algorithms are used to compare two files — two electronic documents — line by line, and to signal the lines where the two files differ. Diff is an essential tool for computer scientists to check their code files and quickly find where and how they have been updated. Diff algorithms, however, are hardly a practical solution for collation, since the algorithms can compare only two files at the same time. On the other hand, collation usually involves more than two witnesses, sometimes hundreds of them. In addition, collation needs to be more precise than just indicating that a line was modified. Rather, collation must be precise at the word level at least, or even at the character level. Although some diff algorithms are more powerful, such as the file comparison of the oXygen editor which can compare up to three documents and highlight word differences in each line⁶, diff is still a limited solution to collation. See for instance a blog post by Laiacona (2007), for more details about the issues of adapting diff algorithms to collation.

2.1.3 Automated Collation

When Dom Froger published his thesis on automating textual criticism in 1968, he spoke of ‘automatic collation’, a term still in use today (Pierazzo 2015; Andrews 2017; Jänicke and Wrisley 2017)⁷. Early terms used for naming the process of automated collation also included: ‘computer collation’ (Hockey 2000; Robinson 1989a; Shillingsburg 1996), or ‘computerized collation’ (Robinson 1989a, 1989b), which are equivalent to the French ‘collation par ordinateur’ (Froger 1968, 230).

These names may suggest that at the time, the difference of method between automated and manual collation had not been discussed in depth: computer collation works like a manual collation, except done with the help of a computer. For instance, Froger (1968, 233) explains that the machine proceeds in the same way as the philologist, by comparing each manuscript to a base text, and then gathering together all the variant readings which were identified⁸. Cannon also

⁶<https://www.oxygenxml.com/doc/versions/19.0/ug-author/topics/file-comparison.html> (Accessed July 28, 2017).

⁷‘Collation automatique’ in French (Froger 1968, 243).

⁸‘La machine, comme le philologue, procède à la collation en comparant d’abord chacun des

stated that ‘automatic collation should proceed as it would be performed manually’ (Cannon 1976, 33), and Hockey explains that ‘the machine is used to simulate a series of human operations’ (Hockey 1980, 145).

This discussion about how the machine proceeds refers in fact to the algorithm behind the collation tool. The next section is dedicated to the collation algorithms and their evolution.

2.2 Collation algorithms

2.2.1 *Early algorithms*

The algorithm of Froger’s program would compare two texts word by word, the base text and a comparison text. The algorithm compares one word from the base text to one word from the comparison text, until it finds a difference. When it finds a difference, it takes two words from each text and compares them together, a total of four comparisons. The algorithm may recognise that one word has been added or omitted in the comparison text, or that a word in the base text was substituted to another word in the comparison text (which would be expressed as a combination of one addition and one omission), or that two words were transposed. If none of these options correspond to the situation, and the algorithm does not find words that are matching in that section of two words, then longer sections of text will be compared: first a section of five words, and eventually a section of twenty-five words. In each case, every word of the base text is compared to every word of the comparison text (see Froger 1966, 153-156).

This technique of collating text with a computer was common to many of the early programs. Schmidt (2009) refers to it as the ‘sliding window’ technique. The ‘window’, or the length of the section of text compared, would ‘slide’ from one word to the next, after a match has been found. The maximum size of the window would vary among the programs, getting larger as the computer’s memory capacities were improved: the shortest is probably a section of 25 words (Froger 1966), while the longest may be a three hundred word window (Gibson and Petty 1970). Cabaniss (1970) lets users decide the length of the windows through parameters. Cabaniss’ program also includes a coarse scan, in which every tenth word from the base text is compared to the comparison text. The use of parameters offers more flexibility to the user than other tools (Hockey 1980, 151). If the window is too short, it may

manuscrits à l’exemplaire de référence, et rassemble ensuite les variantes’ (Froger 1968, 233). It can be noted, however, that previously Froger (1966, 157) remarked that the computer does not proceed like a philologist.

miss a match that occurs outside the section of text under examination. But if the window is too long, the risk becomes greater that two words will be mismatched because of a repetition (Raben 1979, 259).

There were two main issues with this window technique. The first one is that it is easy to obtain a mismatch. Some usual words, which are often recurring in a text, can confuse the algorithm because it looks like the texts are matching when there are actually two different instances of the same word. For example, the two-words Latin expressions *id est*, *potest esse*, or in English ‘to the’, ‘of a’ and so on, can be frequently repeated over a short portion of text and would lead to a wrong alignment (Gilbert 1974). There have been several attempts to offset this issue, by using either a small window of text to compare, as Gilbert has done in COLLATE (Gilbert 1973, 110), or by creating an exclusion list of words to ignore during collation because they are not significant (Cabaniss 1970, 6). The ‘stop words’ to ignore include usually very common content words (such as ‘want’, ‘do’, etc.) or function words, which have little lexical meaning and are often repeated (such as articles, prepositions or pronouns). On the other hand, Gilbert (1974, 112) did not provide an exclusion list because she estimates that the choice should be left to the scholar who is collating. Nevertheless, after a variant has been located, it remains very difficult for the algorithm to find the place where the two witnesses are matching again, especially in prose texts.

The second problem of this technique is that the algorithm functions with exact matching. As a result, two words with an orthographic difference would not be considered as matching. For instance *republica* and *re publica* would not be considered matching. Some tools require that the transcription already be normalised (Roelli 2014), other tools make use of a list of orthographic regularisations (Robinson 1989a, 1994) or a fuzzy matching function (Robinson 1989a). Gibson and Petty (1970, 283) have opted to keep all differences, letting the elimination of insignificant variant happen at a later stage: it is easier to include all differences and then select the ones to be ignored, rather than omitting them in the transcription and realise afterwards that there was in fact a significant difference (it would mean that new transcriptions must be prepared). Froger (1966) and Gilbert (1974, 112) also leave to the scholar the decision to eliminate insignificant variant after the collation has been performed. In addition to those two issues, the ‘sliding window’ technique would not recognise transpositions, and would fail to match correctly omissions or additions which are longer than the section of text compared in the window (Schmidt 2009).

The method of these early collation algorithms was also criticised by computer scientists. Raben (1979) remarks that this technique follows too closely the method of manual collation, instead of fully taking advantage of the computer's capacities. Raben uses the analogy of haystacks and needles, where texts are the haystacks, and significant variants are the needles, and argues that the computer should be used as a 'magnet' to find variants quickly, instead of going through the whole haystack of texts word by word to sort out what is only hay, and what is a needle. Raben proposes a method that would first find the major differences between two texts, such as large omissions, and then align the smaller differences such as punctuation variations.

Cannon (1976) also criticised the existing algorithms for their redundancy. Due to the 'sliding window' technique, which gradually compares larger sections of text, many comparisons are in fact done twice or more. Cannon compares the three algorithms of Gilbert, Cabaniss, and Petty and Gibson, and analysed how many comparisons must be made between two texts of 100 words each in order to decide that the two texts cannot be aligned. He concludes that Cabaniss' algorithm reaches a conclusion sooner thanks to the coarse scan option, but that the decision is not based on enough data. On the other hand Gilbert's algorithm reaches the right conclusion after too many comparisons.

Cannon proposed a new algorithm called OPCOL (for 'optimal collation'), which stores tokens from the base text and comparison text in tables, in order to avoid redundant comparisons. This algorithm is 'optimal' in the sense that it is not possible for any algorithm to collate two texts with a lesser number of comparisons, and still produce a valid output with any possible input. OPCOL is built on the assumption that the two texts collated have in fact tokens in common and not only differences. Hockey (1980, 155) objects to Cannon's approach because it was not tested on a real text, and did not explain how to deal with a total mismatch. Hockey was concerned that Cannon was only interested in the problem from a computer science point of view, but that the solution proposed would not be applicable by literary scholars to real textual traditions. However, Cannon's algorithm was based on a concept that would prove later very useful in automated collation: the edit distance.

2.2.2 *Edit distance*

In computer science, the edit distance quantifies the difference between two strings of characters by counting the number of operations required to transform one string into the other. For instance, the distance between 'London' and 'Lisbon' is three,

because it requires the substitution of three letters 'ond' into 'isb' to transform 'London' into 'Lisbon'. There are four types of operations performed to transform strings: insertion, deletion, substitution and transposition⁹. These operations are very familiar to textual scholars, as they are similar to the operations performed by scribes to alter the text of a manuscript.

There are different definitions of edit distance. For instance, the longest common subsequence (LCS) allows for insertions and deletions only, in which case the distance between 'London' and 'Lisbon' is actually six (three deletions and three insertions). On the other hand, the Levenshtein distance allows for insertions, deletions and substitutions, while the Damerau-Levenshtein distance allows for the four operations. There can be costs associated with each operation to refine the alignment, and a higher cost results in a larger distance.

Vladimir Levenshtein is the Russian scientist who is credited with the invention of the edit distance in 1965. Levenshtein was working on information theory and in particular on an algorithm to correct errors in texts¹⁰. However, the first algorithm to find the minimum edit distance between two strings of text was implemented by Frederick Damerau (1964). Damerau was a scientist at IBM working on natural language processing, and he published a paper titled 'A technique for computer detection and correction of spelling errors' (Damerau 1964). The concept of edit distance has proved very useful for solving problems involving the comparison of strings of characters. In computer science, the edit distance is used for instance to provide automatic spelling corrections, such as the corrections proposed by search engines, or to correct OCR errors. In bioinformatics, the edit distance is used to compare strings of DNA sequences (see Section 2.4.3). The various applications of the edit distance shows the method's usefulness for automated collation (Horton 1994, 90).

Cannon (1976) used the LCS edit distance in OPCOL for comparing two texts. When the algorithm reaches a point where the two texts start to differ, the tokens of each text are stored in a matrix that calculates the longest common subsequence of words matching between the texts. Tokens are added little by little to the matrix until the distance between the two texts starts decreasing, which indicates that tokens match again together. One problem identified with the OPCOL algorithm is how it deals with mismatches (Hockey 1980). Horton (1994, 97) points out

⁹A substitution can be simplified as a pair of insertion and deletion, and a transposition can be simplified as a pair of substitutions.

¹⁰He published in Russian the 1965 article 'Binary codes capable of correcting deletions, insertions, and reversals', which was translated to English (Levenshtein 1966).

that OPCOL does not recognise so easily the end of a variant location. Horton (1994, 98-99) describes a simpler approach to decide when two texts are matching again. Horton's method also stores the texts in a matrix. However, when an exact match is found between two tokens, the algorithm goes backwards to compare the previous tokens and see if there is a match: taking the two strings 'ABCDEFGF' and 'XYZCWABC' for instance, the C in the first string will match the C after XYZ in the second string, but the previous tokens do not match. On the other hand, the C in the first string will also match the second C of XYZCWABC, and the previous tokens AB match in both strings. Therefore, the algorithm will successfully conclude that 'XYZCW' was inserted in the second string.

In addition, Horton (1994) uses the edit distance to find near matching tokens, and thus recognise spelling variants. Horton considers a Levenshtein distance function which will return a value of 1 for two tokens matching exactly, and a value of 0 for two tokens which have nothing in common¹¹. The algorithm is then able to determine that two tokens with a value close to one are likely orthographic differences and should be considered as matching tokens. Horton decided for a similarity threshold (*match threshold*) of 0.8, so that the tokens *own/owne* or *point/poynt* are considered equivalent. Furthermore, a *query threshold* lets users decide interactively whether some pairs of tokens with a value slightly lower than 0.8 should also be considered spelling variants (such as *sences/senses* or *lockt/lokt*). The use of parameters can further refine the algorithm. For instance, some operations may receive a lower cost: the substitution of *d* and *t* can have a lower cost (Horton 1994, 95), so that the two words *search'd* and *searcht* would be closer to each other than for instance *said* and *laid*: between *said* and *laid* there is also only one substitution of the letters *s* and *l*, but these are two different words.

Until now, we have discussed algorithms which compare only two texts at a time. This is called pairwise alignment, but there are other approaches, namely progressive and non-progressive multiple alignment. In the next section, we will see what are the issues of pairwise alignments, and see how those issues can be addressed with multiple alignment.

2.2.3 Multiple Alignment

One of the advantages of automated collation is that its results may be reused for further purposes, such as the creation of a stemma. However, Spencer and

¹¹Although this seems contradictory, the reason is that the distance is expressed as a fraction of the maximum possible distance between two strings (Horton 1994, 99). The reason for that choice is to obtain an output analogous to another function investigated by Horton, the 'Proximity function'.

Howe (2004) argue that a pairwise alignment does not provide ideal results for calculating a *stemma codicum* of relationships between witnesses. The issue of pairwise alignment is that all texts are compared to a base text only, and therefore some data is missing from the results, because the differences between all witnesses are not recorded. For instance, in a place where only one witness is different from the base text, the result will not indicate that this witness is also different from the others (Spencer and Howe 2004, 257).

Since neither pairwise alignment was a suitable option, Spencer and Howe turned to progressive multiple alignment. In progressive multiple alignment, the algorithm starts collating first the most similar witnesses, and gradually adds witnesses which are more and more different (Spencer and Howe 2004, 257). This method was developed by Hogeweg and Hesper (1984), and is widely used in bioinformatics for aligning sequences of DNA or proteins (Pirovano and Heringa 2008). Spencer and Howe (2004) adapted this technique to the collation of textual witnesses. There are three steps in the process:

1. Perform a pairwise collation to determine the distances between all witnesses.
2. Build a 'guide tree', which is the equivalent of an unrooted stemma (i.e. a stemma where the 'root' or archetype has not been selected among the nodes).
3. Collate progressively the witnesses from the most similar (with a smaller distance) to the most different (with a larger distance).

The advantage of a progressive method is that the insertion of gaps to align the witnesses is easier: if the witnesses collated are very similar, the location of a gap is more certain (Spencer and Howe 2004). The issue, on the other hand, is that it takes a long time, especially the first step: to calculate the best order in which to collate the witnesses is the most difficult part (Spencer and Howe 2004, 263). In addition, if a mistake is introduced early in the process of the actual collation (the third step), it is impossible to correct it later: 'once a gap, always a gap' (Pirovano and Heringa 2008, 146; Spencer and Howe 2004, 265). Finally, there is a problem of circularity: an optimal collation requires that the witnesses are compared in the right order, which means that some inferences must be made regarding the witnesses' relationships; but then to make inferences about the witnesses' relationships, collation is needed. This issue was also highlighted for

2.2. Collation algorithms

W1	a	b	c	d	F	g	h	i	!	K	!	q	r	s	t
W2	a	b	c	d	F	g	h	i	!			q	r	s	t
W3	a	b	c	d	E	g	h	i			!	q	r	s	t

Table 2.1: Incorrect progressive alignment.

W1	a	b	c	d	F	g	h	i	!	K	!	q	r	s	t
W2	a	b	c	d	F	g	h	i	!			q	r	s	t
W3	a	b	c	d	E	g	h	i	!			q	r	s	t

W1	a	b	c	d	F	g	h	i	!	K	!	q	r	s	t
W2	a	b	c	d	F	g	h	i			!	q	r	s	t
W3	a	b	c	d	E	g	h	i			!	q	r	s	t

Table 2.2: Correct non-progressive alignments.

the construction of phylogenetic trees in bioinformatics by Hogeweg and Hesper (1984), who argue that alignment and the creation of a tree must not be treated separately, but as a single problem. The solution proposed by Hogeweg and Hesper (1984) is an iterative process: the alignment and guide tree are built in parallel, and improve each other in turns. The guide tree helps improve the alignment, which in turn helps correcting the guide tree. Another solution to the issues of progressive alignment is ‘non-progressive multiple alignment’, such as PicXAA (**P**robabilistic **M**aXimum **A**ccuracy **A**lignment, see Sahraeian and Yoon 2011).

The main difference between progressive and non progressive alignment is the dependence on order in which witnesses are collated: non-progressive alignment has less dependence on collation order, but does not always produce the same results if a different order is used (Spadini 2016, 136-137). In addition, non-progressive alignment may be more efficient in certain situations. Tables 2.1 and 2.2 show an example where progressive alignment is less accurate than the non-progressive alignment (Ronald Dekker, private correspondence). The wrong alignment in progressive alignment is caused by the longest common sequence (LCS) between W3 on the one hand, and the alignment of W1 and W2: after W1 and W2 are aligned, the sequence ‘!qrst’ becomes the new longest common sequence with W3.

In CollateX, no guide tree is created in order to determine the best order of collation, as was the case in Spencer and Howe (2004). Although the developers of CollateX did not witness a dependence order while testing the program, it could have been an issue in some circumstances (Dekker et al. 2015, 10). Since version 1.7, CollateX

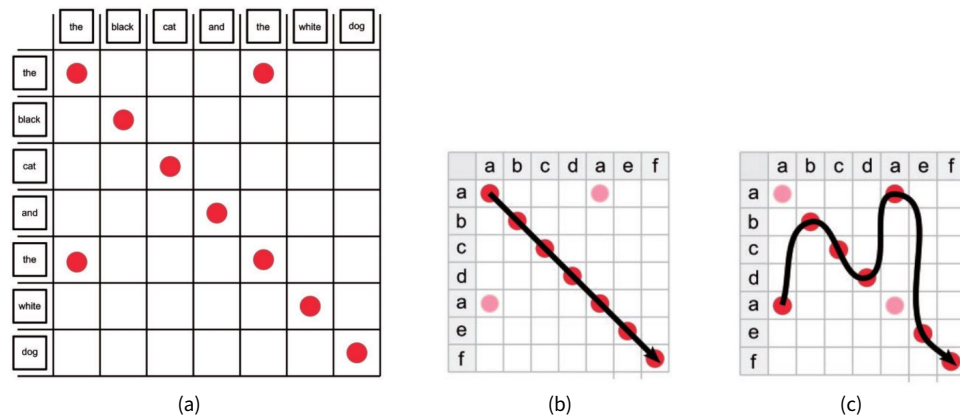


Figure 2.4: Dekker algorithm, alignment matrix (Dekker et al. 2015).

implements a non-progressive alignment algorithm. There are three possible algorithms available in CollateX:

- **Needleman-Wunsch** is an algorithm widely used in bioinformatics to align sequences of proteins or nucleids (Needleman and Wunsch 1970). According to the CollateX documentation, it does not take into account transpositions.
- **MEDITE** is an algorithm that was specifically created to align sequences of textual transcriptions. It was created in France, in collaboration between genetic critics at ITEM (Institut des Textes et Manuscrits Modernes) and Artificial Intelligence specialists from LIP6 (Laboratoire d'informatique de Paris 6 - Université Pierre et Marie Curie) (Ganascia 2014). The implementation of this algorithm in CollateX is still experimental (The Interedition Development Group 2013).
- The **Dekker algorithm** created by Ronald Haetjens Dekker is the 'most mature algorithm offered by CollateX thus far' (The Interedition Development Group 2013).

The Dekker algorithm takes into account the reader's common sense to find the most natural alignment: given a matrix of two documents' text, the best alignment is the one that is closest to a diagonal from the upper left corner to the bottom right corner, which represent a natural way for readers to read through the text (Dekker et al. 2015). Figure 2.4(b) shows a more natural alignment than figure 2.4(c).

To evaluate recent collation algorithms, Dekker et al. (2015) focus on three criteria: transposition detection, support for flexible token matching (such as fuzzy matching, described above), and influence of the base text or comparison order on the final collation result. This set of criteria provides a first basis to compare the various collation algorithms, and is thus useful to analyse the newest collation tools regarding their algorithm implementation. This was particularly useful to evaluate the Classical Text Editor (CTE) collation algorithm. Since CTE is a commercial tool under license, its algorithm is not described. However, CTE's creator Stefan Hagel was willing to explain broadly the strengths and weaknesses of his algorithm with respect to these three criteria: the main advantage of CTE's algorithm would be its use of the Levenshtein distance in order to correctly align texts with many orthographic variants. On the other hand, CTE does not deal well with transpositions of long segments of texts, and as it performs a pairwise alignment with a base text, the order in which witnesses are compared will influence the collation result.

The tool iAligner applies a modified version of the Needleman-Wunsch algorithm, which reduces the number of comparisons: instead of comparing each token from one witness *m* to each token of another witness *n*, each token in *m* is compared with ten tokens of *n*, thus limiting the amount of comparisons needed (Yousef and Palladino 2016). In addition, the Levenshtein distance may be used to determine a matching threshold and make the alignment more flexible for texts with orthographic variation (Yousef and Palladino 2016). The algorithm of iAligner is 'syntax-based' because it takes into account the order of words in a sentence. However, it does not use lexical information such as part-of-speech tagging (Yousef, Palladino, and Crane 2017). Figure 2.5 shows three examples of alignment with iAligner, created with the online demo¹². The second example figure 2.5(b), which uses the Levenshtein distance, is more efficient than figure 2.5(a): the three words *erit*, *egit* and *agit* are aligned together. On the other hand, *pater* and *pauper* are still not properly aligned. iAligner does not recognise transpositions (they are visually analysed by the user), and the developers do not mention any effect of the collation order on the results. Figure 2.5(c) shows that, by inverting the second and the third witnesses, the resulting alignment will be different, and *pater* and *pauper* are then correctly aligned. However, this is an early demonstration tool, and as iAligner is further developed, the alignment may improve (Yousef, Palladino, and Crane 2017).

¹²The demo seems to be most efficient with Greek and English texts: <http://www.dh.uni-leipzig.de/tools/Alignment/multiple.php> (Accessed July 17, 2017).

2.2. Collation algorithms

Figure 2.5 illustrates three examples of text alignments using iAligner, labeled (a), (b), and (c). Each example shows three input texts and their alignment results based on different criteria.

(a) Shows an alignment where the Levenshtein Distance algorithm is used. The input texts are:

- First text: dives iure erit pater praemio petit
- Second text: dives iure egit pater praemio petit
- Third text: dives iure agit pauper praemio petit

The alignment results are as follows:

dives	iure	erit	pater	praemio	petit
dives	iure	egit	pater	praemio	petit
dives	iure	agit	pauper	praemio	petit

(b) Shows an alignment where the Levenshtein Distance algorithm is used. The input texts are:

- First text: dives iure erit pater praemio petit
- Second text: dives iure egit pater praemio petit
- Third text: dives iure agit pauper praemio petit

The alignment results are as follows:

dives	iure	erit	pater	praemio	petit
dives	iure	egit	pater	praemio	petit
dives	iure	agit	pauper	praemio	petit

(c) Shows an alignment where the Levenshtein Distance algorithm is used. The input texts are:

- First text: dives iure erit pater praemio petit
- Second text: dives iure agit pauper praemio petit
- Third text: dives iure egit pater praemio petit

The alignment results are as follows:

dives	iure	erit	pater	praemio	petit
dives	iure	agit	pauper	praemio	petit
dives	iure	egit	pater	praemio	petit

Figure 2.5: Examples of alignments from iAligner. Created with iAligner's online demo <http://www.dh.uni-leipzig.de/tools/Alignment/multiple.php> (July 17, 2017).

```

112 -def create_html_for_table(table, layout):
113   # create visualization of alignment table
114 - if layout == "vertical":
115     prettytable = visualizeTableVertically(table)
116 - else:
117     prettytable = visualizeTableHorizontal(table)
118     return str(prettytable)

96 +def create_html_for_table(table):
97   # create visualization of alignment table
98 + if table.layout == "vertical":
99     prettytable = visualizeTableVertically(table)
100 + elif table.layout == "horizontal":
101     prettytable = visualizeTableHorizontal(table)
102 + else:
103 +     raise Exception("Unknown table layout: "+table.layout)
104     return str(prettytable)

```

Figure 2.6: Diff example. Retrieved from CollateX’s Github repository <https://github.com/interedition/collatex> (Octobre 6, 2017).

2.2.4 Diff Algorithms

While several collation algorithms were inspired from research in the field of bioinformatics, other collation tools have adapted algorithms from the field of computer science. As we have seen above, diff algorithms are designed to compare two digital files, most often two different versions of the same file of code, and to signal the lines where the two versions are different (see figure 2.6). In practice, it means that with a diff algorithm, text files are tokenised at carriage return characters.

This is enough for comparing short lines of code and quickly finding where the code has been modified. On the other hand, diff is not precise enough for collation, especially of prose texts where carriage returns are not predictable and are used to separate entire paragraphs (Laiacona 2007). The two main issues are that diff compares only two texts, and that the comparison is not precise enough.

However, diff algorithms can be adapted for the purpose of textual collation, with a pairwise comparison. Juxta’s collation algorithm, for instance, has implemented the diff algorithm by Myers (1986). One important change was to use words and punctuation marks as tokens instead of lines. The precision of the diff algorithm was also improved by a multi-pass approach: some areas of text where a change has occurred are compared a second time, in order to solve ambiguities (Laiacona 2007). Myers diff was also used to collate witnesses in the *Dravyasamuddeśa* project (Li 2017), and the collation algorithm developed for Petrus Alfonsus was adapted from a diff algorithm as well (Roelli 2014). Creating a diff algorithm was the aim of the project eComparatio, however it was not without difficulties: they needed to trade off speed for efficiency (eComparatio Github page).

The multi-pass approach essentially repeats twice the alignment step of the Gothenburg model, and in some cases the tokenisation step as well. For instance in Prahbed, there are two stages of ‘gross collation’, at chapter and paragraph levels, and then a ‘fine collation’ at the word level (Chaudhuri 2015). The tools LERA and LAKomp also operate on two stages of tokenisation and alignment, first for

larger segments, and then another comparison of those segments at the word level, although the algorithm is not based on diff but on the Levenshtein distance (Pöckelmann, private correspondence).

2.3 The Evolution of Automated Collation

The work of Froger (1968) is often listed as the pioneer in automated collation (Gilbert 1973; Hockey 1980; Schmidt 2009)¹³. When he started to work on a collation program in 1960, no such program existed yet (Froger 1966, 135). However, between the moment when Froger started to work and his well-known essay of 1968, a few publications appeared which report on the creation of automated collation programs. Vinton Dearing published a paper delivered at a Seminar on Bibliography in May 1962, where he describes what is now considered the first collation program (Dearing 1962, 18-19). The contributions from Zarri certainly also provided crucial groundwork in the developments of automated collation, (for instance Maretti and Zarri 1967). In addition, Andrews (2014b) describes TUSTEP as ‘one of the first successful computer programs for the scholarly processing of texts... [including] a limited facility for the automatic comparison of text’ (Andrews 2014b, 182). TUSTEP was thus one of the first collation tools which was not limited to a specific project, but was used with success to edit multiple texts¹⁴.

Froger, like some of his successors, was rather optimistic regarding the possibility of automating the collation process with a computer: ‘the collation of manuscripts is an operation similar to accounting, it may very well be performed by a computer’ (Froger 1968, 230)¹⁵. Robert Marichal, Froger’s advisor who wrote the preface to the essay, considered that automated collation, although possible in theory, was unachievable in practice, independently of technical progress (Froger 1968, X)¹⁶. Marichal thought that a computer would never be able to make a difference between substantial variants and accidentals, so that collation would hide the few significant differences among the many trivial ones. Kraft (1995) also describes pessimistic attitudes towards the use of computers in the 1970s, but for differ-

¹³The early history of automated collation can be found in particular in Gilbert (1973), Hockey (1980) and Marín (1991), whereas more recent progress is described by Andrews (2014b).

¹⁴According to the International TUSTEP User Group (ITUG), more than 900 critical editions have been produced with the help of TUSTEP since the first one in 1972 <http://www.itug.de/Geschichte.html> (Accessed May 19, 2017).

¹⁵‘La collation des manuscrits est une opération qui relève en quelque sorte de la comptabilité: la machine peut fort bien l’exécuter’ (Froger 1968, 230).

¹⁶‘Collation: en principe, aucune difficulté; la machine le fera impeccablement’, but in practice, on the matter of collation, ‘l’emploi d’une machine est impraticable et le sera probablement toujours’ (R. Marichal, preface to *La critique des textes et son automatisation* 1968, pages VIII and IX).

2.3. The Evolution of Automated Collation

ent reasons. In New Testament studies, the use of computers would require vast amounts of work to prepare electronic resources.

Nevertheless, the idea that collation was an activity perfectly suited for computer processing was accepted in the scholarly community, and many scholars followed Froger's path: at least nine collation tools were published between 1968 and 1973, such as Vinton Dearing's program to edit Dryden (Dearing 1970), the OCCULT project (Gibson and Petty 1970), Gilbert's COLLATE (1973) and TUSTEP (1972). In 1980, Susan Hockey was claiming that collation is a mechanical process, easily implemented with a computer: 'provided that there is no doubt about the readings in the manuscripts, the collation of manuscripts is a purely mechanical process, consisting of merely identifying places where two or more texts do not match' (Hockey 2000, 144-145). However, Hockey later revised her opinion when it became clear that fully automated collation was not yet possible, and that collation tools were still facing thorny issues: 'It was initially thought that collation was a fairly mechanical process which could be simulated by computer. However, the kinds of comparisons needed for complex text are much more sophisticated than those provided by standard software for comparing computer files' (Hockey 2000, 125).

The shift between the two points of view seems to have taken place in the late 1980s, at the time when Robinson published about a new collation program, Collate, that he developed while collating and editing forty-four manuscripts of two Old Norse poems (Robinson 1989a, 1989b). In fact, the late 1970s and the 1980s were a productive period for collation tools: no fewer than six programs were implemented, such as Miriam and Peter Shillingsburg's PC-CASE (Shillingsburg 1978; Shillingsburg 1980), the Donne Variorum Collation Program DV-COLL (Stringer and Vilberg 1987) or URICA!, User Response Interactive Collation Assistant (Cannon and Oakman 1989). This intense scholarly attention on automated collation, therefore, likely helped to refine the concept. While working on a second version of URICA!, Hilton (1992) attempts one of the first definitions of automated collation. He distinguishes between 'fully automated collation', and 'interactive collation', where the computer program may need the help of the editor to correctly recognise variant readings:

There are two basic strategies for collating texts by computer: fully automated, batch collation and interactive, computer-assisted collation. The first strategy, fully **automated collation**, has received the lion's share of attention. Its goal is to find and record all textual variants without human interaction. [...]

The second strategy, **interactive collation**, [...] provid[es] the com-

2.3. The Evolution of Automated Collation

puter with human assistance whenever necessary (Hilton 1992, 139-140).

The issue of automated collation, according to Hilton, is that computers have trouble locating the end of a variant reading. The program would therefore locate the start of a variant, and let the editor decide where the variant would end. The interactions between the editor and the program would thus make collation easier and faster than a fully automated collation, which requires a large amount of corrections in the output. Robinson's Collate program, similarly, may be interrupted by the editor at any moment during the collation process in order to correct the results or enter additional information such as regularised orthographic forms (Robinson 1994). Based on his experience with automated collation, Robinson devised seven principles of computer collation, of which the third stated that 'there should be interactive collation' (Robinson 1994, 35). The reason for this principle is that there would always be some difficult passage which would require the judgement of an editor, no matter how efficient the collation algorithm.

Collate had already evolved between 1989 and 1994. The first version, which Robinson called 'Collate 0', was a command-line tool used only by himself during his doctoral thesis (Robinson 2007b). The program was then developed for a wider audience and a more general purpose, with a complete rewriting and the addition of a Graphical User Interface. This new version, 'Collate 1', was first released in 1991 (Robinson 2007b). Later, Collate went again through a 'drastic reshaping and enlargement', to such an extent that Robinson came to think of it as 'Collate 2' (Robinson 2007b). Collate 2 was released in 1996. A key difference between Collate 1 and 2 was a change of purpose: Collate 2 was oriented towards the creation of electronic editions, instead of printed editions (Robinson 2007b). There were other differences, such as the abandon of the base text for instance (see also Section 2.4.2 below). Collate 2 was maintained until 2007, when a change in the Macintosh operating system meant that Collate needed to be rewritten again.

In the early 1990s, Manuscript was developed by Morrill and Lewis for collators of the International Greek New Testament Project, inspired by Collate (Parker 2008, 103). At the time, TUSTEP was not adapted to the demands of the New Testament (Kraft 1995, 271). Collate, although promising, required the creation of transcriptions which were not yet available. Manuscript was 'aimed at facilitating the entry of data, as well as their organization and their ultimate presentation' (Kraft 1995, 272).

2.3. The Evolution of Automated Collation

While Robinson was setting out the design of Collate's successor, Juxta was developed by the Applied Research in Patacriticism (ARP) group at the University of Virginia in 2005. Juxta was first released as a desktop application (Wheeles and Jensen 2014). The project was then taken up by the scholarly organisation NINES¹⁷ and the company Performant Software. Juxta was transformed into a web service with the web interface Juxta Commons (Wheeles and Jensen 2014). Another interface, Juxta Editions, was built on top of the web service and released in 2015.

By 2015, CollateX, the successor of Collate, was then available. In 2007, Robinson had decided to collaborate on the development of Collate with colleagues from the Hague: he shared with them a series of blog posts in which he outlined his ideas for the future design of Collate (Robinson 2014). Some features from Collate would be retained, such as its ability to compare 'word objects' and not strings of characters (Robinson 2014). Word objects for a distinction between the original word of the text and a normalised form (see below p. 79 on normalisation, and also Chapters 4 and 7). New features of Collate would include for instance the possibility to handle XML input, and a better division of the different stages of alignment. Robinson (2014) identified four stages: alignment, storage and adjustment of the result, and finally the identification of variants.

The division of the collation stages was formalised with the Gothenburg model in 2009. A workshop organised in the context of *Interedition* gathered in Gothenburg the developers of Juxta, CollateX, and Text::TEI::Collate, who discussed the various issues they were facing with automated collation¹⁸. The Gothenburg model thus emerged from the discussions between experts of automated collation, both software developers and scholarly editors who were using digital tools (Dekker et al. 2015).

The Gothenburg model divides the process of collation into five basic successive tasks, a separation of concerns that would improve the flexibility of collation tools¹⁹:

1. Tokenisation: division of the text in segments called tokens;

¹⁷See <http://www.nines.org> (Accessed May 19, 2017).

¹⁸Interedition is a COST Action that aimed to promote the interoperability of the tools and methodology used in the field of digital scholarly editing: <http://www.interedition.eu/> (Accessed May 05, 2017).

¹⁹Although (Dekker et al. 2015) gives only four successive tasks, with normalisation a part of the first tokenisation step, CollateX documentation divides tokenisation and normalisation as separate steps.

2.3. The Evolution of Automated Collation

2. Normalisation: considering some tokens as identical for collation purposes, for instance by transforming all tokens to lower case;
3. Alignment: identifying similarities and differences among the tokens in various versions;
4. Analysis: interpretation of the result;
5. Output: format and visualisation of the result.

This modular approach of automated collation was not a new idea. In 1973, Gilbert already adopted a modular design and highlighted the advantages of this approach over other existing tools (Gilbert 1973, 147). TUSTEP, as well, lets the user ‘solve complex problems in small and controllable steps’ (Ott 1991, 435). Actually, collation programs such as the one from TUSTEP were not created in isolation, but rather as a part of a bigger infrastructure aiming at automating the whole scholarly editing process, from gathering and collating variant forms of a work, to printing and publishing the edited text and the critical apparatus (Hockey 1980; Shillingsburg 1996). For instance, UNITE was part of an integrated system for producing critical editions (Marín 1991, 109), as well as Shillingsburg’s Computer Assistant PC-CASE; Collate also refers to a suite of twenty-five programs that Peter Robinson used to transcribe, collate and edit manuscripts (Robinson 1989a, 99). When creating Collate, Peter Robinson’s first goal was to feed transcriptions to the computer and extract a critical apparatus at the end (Robinson 1989a, 99). DV-COLL was intended to ‘computerize the work’ of editing Donne’s *Variorum* (Stringer and Vilberg 1987). The purpose was to implement a workflow that would imitate the manual process of text editing, and this was done by isolating specific differences between texts, namely the variants (Stringer and Vilberg 1987; Hilton 1992).

Collation was only one aspect of those editing systems, but it was an important and complex one. The importance of collation is reflected partly in the names of those tools: both Robinson’s and Gilbert’s tool were called COLLATE. In addition, Gilbert defines the purpose of her tool, the creation of a critical editions with a computer, as an activity centred on collation: ‘computer-aided critical editions, i.e., the use of the computer to compare texts or manuscripts and to indicate variants’ (Gilbert 1973, 139)²⁰. This definition of a computer-aided critical edition is in fact quite close to a definition of collation itself. Since collation was a very difficult step to implement in a digital, automated workflow for the creation of critical editions, it was a natural

²⁰In a later publication, Gilbert updated her definition: ‘Automatic text collation—the use of the computer to locate variant readings in manuscript copies of a text’ (Gilbert 1973, 106).

2.3. The Evolution of Automated Collation

evolution to further break down collation into smaller and simpler tasks. The next generation of collation tools, such as Juxta and CollateX, thus usually follows the Gothenburg model of collation.

2.3.1 *The Terms Used in the Context of Automated Collation*

'Automatic collation' was disregarded by the scholars who authored the two main publications on CollateX (Dekker and Middell 2011; Dekker et al. 2015): they preferred the term 'computer-supported collation', which is more accurate since it does not imply that collation is fully automated. However, computer-supported is a rather vague term which could refer to any kind of use of a computer, even to a collation done manually but recorded in a digital format such as a spreadsheet or a Microsoft Word document. The term of 'semi-automatic collation' is therefore another suitable alternative, and seems to gain weight in the scholarly community (Gabler 2008; Dekker et al. 2015; Spadini 2015; Yousef and Palladino 2016). Semi-automatic collation may refer in particular to the third stage in the Gothenburg model: the analysis. Analysis requires some additional input from the editor to interpret the results from the program or to adjust some parameters, and that interpretation can be fed again to the collation algorithm, in order to improve the next output produced by the program. Such an interpretation may be to recognise that two or more words have been inverted in the text as one intervention, instead of being two unrelated actions (one addition and one deletion). In practice, however, this analysis step is rarely implemented, since it requires additional coding capacities. The most important step in the Gothenburg model is in fact the alignment phase. During the alignment, the different witnesses are compared to one another, to determine which segments of text are equivalent, and which segments are different. The importance of alignment is also evident in the name of the most recent collation tools. TRAViz (2015) stands for 'Text Re-use Alignment Visualization', and iAligner (2016) is presented as a tool for 'text alignment' which can facilitate several degrees of textual comparison, including text reuse (Yousef and Palladino 2016). Robinson (2014) envisioned that the successor of Collate should be able to deal not only with collation, but also with other situations such as plagiarism and intertextuality; thus the purpose of CollateX is 'text comparison' in general (Dekker et al. 2015)²¹.

This survey of the different names given to the process can shed light on the evolution of the scholarly understanding of this practice. While technology was improving, scholars seemed sometimes less inclined to speak of automatic or automated

²¹'[T]ext comparison is pivotal to any kind of textual scholarship' (Dekker et al. 2015, 2). For Andrews (2017), this is their definition of automated collation.

2.3. The Evolution of Automated Collation

collation, but adopted new terms such as ‘interactive’ or ‘computer-supported’ collation, which would take into account the manual input required to make a proper use of automated collation tools. There are four terms that have been used more often than others: automatic, semi-automatic, interactive, and computer-supported collation. ‘Interactive collation’ is a term that was specifically applied to tools such as URICA! (Cannon and Oakman 1989; Hilton 1992) or Collate (Robinson 1994), where the user intervenes directly during the collation process. The three other terms conjure up different nuances, although it is impossible to know in retrospect if these nuances were intended by prior scholars. In summary, the most generic term is computer-supported, and in theory it could refer to any kind of collation involving the use of a computer. It is a term broad enough to cover all tools discussed in this chapter, including interactive collation. The term ‘semi-automatic collation’ might apply to tools with an even higher degree of automatisation than interactive collation: here the user intervenes in the collation algorithm by tweaking parameters or adding a level of interpretation to the collation, but without necessarily interrupting the process. Finally, fully ‘automated’ or ‘automatic collation’ could apply to tools such as Juxta and CollateX, when there is no user intervention in the algorithm. However, some users who need to do intensive correction of collation results may rather define the process as semi-automatic instead of automatic.

The purpose of automated collation evolved as well, as this complex task was divided into smaller steps: while the first collation tools were focused on replicating the manual workflow of an editor with the aim of getting a print critical edition at the end of the process, the next generation of collation tools was more concerned with an accurate text alignment than the creation of a critical apparatus. The important point for collation tools became to offer a good alignment in various output formats which can be used so as to satisfy different needs, such as preparing a critical apparatus for instance, but also grouping witnesses into families and establishing a stemma codicum, or identifying instances of text reuse.

The method of automated collation needs to be discussed as well. As Andrews pointed out in a contribution during the European Society for Textual Scholarship (ESTS) in 2016, it is necessary to understand what automated collation means for scholars from different backgrounds, Humanities researchers or software developers, in order to allow for an effective collaboration (Andrews 2017). In the next section, I will discuss the methodology of automated collation and especially its differences with respect to manual collation, as well as the Gothenburg model of automated collation.

2.4 Methodology of Automated Collation

The methodology of automated collation differs from the methodology of manual collation on two main points: the base text, or absence thereof, and more importantly the need to first transcribe the full text from the witnesses in a digital format before letting a computer program do the comparison (see Macé et al. 2015, 333 ff.). The latter is a particularly relevant issue, since transcribing manuscripts is one of the main arguments against automated collation from literary scholars, especially classicists²².

2.4.1 Transcription versus Collation

As we have seen previously, there is no transcription step in a manual collation: variant readings are recorded while comparing the base text and another witness. However, the collator needs to write down each variant reading, whether in an existing critical edition, a notebook or a digital document. In effect, this means that a transcription must occur, but it is a very partial transcription of a few words scattered across the complete text of the witness. Only variant readings deemed significant, or potentially significant, are transcribed. The collation and transcription are thus combined and the two activities can hardly be separated in the traditional method. On the other hand, automated collation requires full transcriptions of the witnesses to be collated by a computer (Andrews 2014b, 177). Transcription remains the most laborious part of the work in a digital workflow (Andrews 2012; Roelli 2014) and yet a very important one, since the success of the next steps, such as collation, will depend on an accurate transcription (Mordenti 2001).

This change in methodology has not always been welcomed by scholars for several reasons. First, a transcription is considered by many to be much more time-consuming than a traditional collation: Reeve argues that even complete collation of the 215 witnesses of Geoffrey of Monmouth's *Historia regum Britanniae* would have been unrealistic, let alone complete transcriptions, especially since not all of the 215 manuscripts have editorial value (Reeve 2011, 389). Manual collation was also deemed faster than full transcriptions in editing the *Florilegium Coislinianum*, a Medieval Greek anthology (Macé, De Vos, and Geuten 2012, 113, note 11).

In addition, Whittaker considers that a transcription is not only longer but also more difficult than a traditional collation because the transcription represents

²²See for instance a discussion of September 2015 about 'Supporting MS collation' in the Digital Classicist mailing list: <https://www.jiscmail.ac.uk/cgi-bin/webadmin?A1=ind1509&L=digitalclassicist#17> (Accessed July 31, 2017).

2.4. Methodology of Automated Collation

actually a diplomatic edition: ‘a diplomatic edition without the support of a base text is a more lengthy and indeed risky task of potentially monumental intricacy and minimal utility’ (Whittaker 1991, 128, note 1). Fischer (2012) also seems to equate the transcription with a diplomatic edition. However, a diplomatic edition is more than a transcription, and it includes line breaks, page breaks, abbreviations and differentiated letter shapes (Pierazzo 2011, 463-464). Here lies perhaps the greatest difference between the approach of literary editors like Whittaker and Reeve on the one hand, and digital humanists like Andrews and Robinson on the other.

Both Andrews and Robinson have argued that a transcription is not any more difficult or time-consuming than a collation, if the editor proceeds this way: first identify a witness already transcribed that is similar to the one that needs transcription, and then alter the transcription of the first witness to match the text of the new witness (Robinson 1989a; Andrews 2012). Indeed, according to Robinson, the separation between transcription and collation made the task ‘both less exacting and more accurate’ than if he had done it in a traditional way (Robinson 1989a, 100). Moreover, they have highlighted advantages of transcription over traditional collation. A transcription file may be easier to correct against a manuscript, since it contains only one text and not several in parallel (Andrews 2014b, 178). It also prevents the editor from being tempted to ignore a detail which is considered irrelevant at the time of collation, but might actually become important later. Robinson gives as an example the hyphenation of two particular words, a phenomenon that is usually not noted, but which turned out to be crucial to distinguish a family of manuscripts (Robinson 1989a, 101). The issue is that editors often decide while they are collating what is significant; however, it is difficult to make such decisions without having read through every witness, and two editors might well make different decisions (even the same editor might make different decisions at different times).

Some scholars may argue against full transcriptions of the witnesses, because it is often the case that merely a few sections of a witness must be collated in order to evaluate its editorial value. In practice, when dealing with a large manuscript tradition, only a few witnesses will be fully collated, i.e., the ones selected as the most relevant to edit the text. However, a transcription of extracts, combined with their automated collation, could also be adopted (Roelli and Bachmann 2010). A manual collation would be helpful to see the pattern of manuscript groups emerging, and could be a good way to grasp a complicated tradition. However, it could happen that during collation, an editor will get an impression, and then tend

2.4. Methodology of Automated Collation

to interpret evidence so that it will confirm the first impression. Transcription may also enable the scholar to discover patterns within the tradition, but likely to a lesser extent: although similarities between witnesses may be noted during transcription, groups of shared variant readings will only become apparent after collation. It may be argued then that the combination of transcription plus automated collation gives less room for the editor to rely on intuitions built during collation. Therefore, transcription samples could be done the same way as collation samples are done. This solution would still prevent the editor overlooking a small yet significant variant, and it would also limit the possibilities of editorial bias in collation.

It is worth noting here that manual collation is not only combined with transcription, but also with spelling regularisation. On the other hand, diplomatic transcriptions record primarily non-normalised spelling, and the recording of normalised orthographic forms in the transcription may be necessary in order to facilitate the automated collation (for instance in the context of medieval traditions with many spelling variations) or to help select significant variants in the collation results and minimise the influence of trivial differences (Roelli and Bachmann 2010). It would of course be possible to transcribe directly a regularised text instead of a diplomatic version, as Roelli and Bachmann did while working on the tradition of Petrus Alfonsi, but in general it is not a recommended practice (Robinson and Solopova 1993, 33; Macé et al. 2015, 333).

Boschetti (2007, 3) raised the issue of transcribing entire documents. According to (Boschetti 2007), it is not always possible to transcribe a full witness: an important indirect tradition (secondary sources quoting extracts from the edited text), or a rich secondary literature with many conjectures are examples where a transcription of a full witness may not be feasible because the context is missing. For instance, modern readers of ancient commentaries (*scholia*) would not be able to know from which manuscript the commentator was reading the extracts that are quoted in the *scholia*. Conjectures as well can lack a context, since it is not always possible to know with which edition a scholar was working when they proposed a conjecture. Boschetti (2007) considered that since those incomplete witnesses could not be fully transcribed, therefore they could be collated with automated collation tools. However, incomplete witnesses do not prevent automated collation in practice. This issue was avoided in the Digital Mishnah project by creating special documents for readings from incomplete secondary sources (Lapin 2013, 449).

In summary, Parker (2008, 101) highlights several advantages of transcription: it can be reused, so that the next scholars will not have to do again the same work

2.4. Methodology of Automated Collation

from scratch; information relevant to other than just text critics can be encoded in the transcription; it facilitates collaboration; and transcriptions are easier to correct with minimal risk of adding more errors.

2.4.2 Base text

According to Macé et al. (2015), the absence of a base text is one of the core principles of automated collation: ‘rather than choosing a base text (or reference text) against which all subsequent texts should be compared, the scholar refrains from any selection or comparison at all’ (Macé et al. 2015, 333). However, this was not always the case in automated collation. As we have seen above p. 48, the first scholars to attempt automated collation, such as Froger, would follow closely the method of manual collation. In manual collation, the choice of a base text is one of the first step during the collation process, and an indispensable one (see Section 1.4.1.1 above, p. 31). As a result, the early collation programs would also make use of a base text. One text is chosen as the base against which all the other witnesses are compared consecutively, and the variants are gathered at the end.

Froger prefers to call the base text ‘reference text’ (*texte de référence*) to avoid confusion with the text chosen as a base for editing by scholars who follow Bédier’s ‘best text’ method (Froger 1966, 139). The base text is also called sometimes ‘master text’ (Cannon and Oakman 1989; Robinson 1989a). As for manual collation, in an automated procedure the choice of a base text is mostly dictated by practical reasons: scholars are recommended to avoid difficulties such as many corrections by a second hand or lacunae (Froger 1966, 139). Dearing (1962, 14) as well advises to choose the longest text. Gibson and Petty (1970, 285) on the other hand selected the shorter text as a base text, which may not be the best methodology according to Hockey (1980, 150). The presence of lacunae in the base text can be particularly problematic, because it means that collation programs cannot find variants among the other witnesses in the portion of text that is missing from the base text (Andrews 2014b, 182). Since each witness is compared to the base text only, and not to all the other witnesses, the portion of text absent from the base text will not be compared in the other witnesses. In order to address this issue, Robinson’s Collate would also compare all witnesses to each other at the points where there is a difference from the base text (Robinson 1989a, 99).

The base text is often thought of as a kind of clothesline for hanging up the variants of the other witnesses (Robinson 1989a, 102; Reeve 2000, 197). For Robinson, the best results can be obtained from Collate when the base text is split into minimal units of sense. For example, the old Norse manuscripts that Robinson was collating

2.4. Methodology of Automated Collation

have the following readings: *Mimameidr*, *Munameidr*, in various combinations of *Mima/Muna* and *meidr/meidir* (either in one or two words). If the base text has the reading *Mimameidr*, the collation results will make it difficult to see at a glance which manuscripts agree in the first part of the compound and read *Mima*, and on the contrary which manuscripts read *Muna*. On the other hand, this information becomes much more obvious if the base text divides *Mimameidr* into two readings, *Mima* and *meidr*. In this case, the base text is not an existing witness, but an artificial creation to support the need of the researcher during collation: ‘the final master was worthless as a text—but it provided a splendid series of pegs on which the variants might hang’ (Robinson 1989a, 102).

Despite the issues related to the base text with automated collation, some literary scholars expressed their concerns regarding the possibility to collate without a base text: Whittaker (1991) was arguing that transcription, without the support of a base text would be even more difficult (see p. 67). On the contrary for Robinson, the base text is the ‘greatest weakness’ of editors of medieval texts for one reason: collating with help of a base text let scholars make decision about the (in)significance of some differences before they have surveyed the complete manuscript evidence. As a result, the editor might omit to record an orthographic variant that seems trivial at the time of collation, but would appear significant at a later stage (see Robinson 1989a; 1991, 86, note 22). Nevertheless, even recent collation programs such as the one implemented in the Classical Text Editor (CTE) in 2015 still make use of a base text. The reason is likely to be that CTE was designed to support the traditional editing workflow, and therefore followed the manual collation process closely.

While Collate was still working with a base text, the shift to a real ‘baseless’ automated collation process seems to have happened around 1998 (Robinson 2007b). At the time, Robinson was working on Collate 2, attempting in particular to apply phylogenetic methods to the creation of a stemma from Collate’s collation results. Phylogenetics is a field of biology concerned with studying the evolution of living organisms. Scientists who study DNA mutations face similar issues to textual critics. DNA is a molecule made of amino-acid sequences (indicated by a sequence of letters), which contains genetic information: by comparing the sequences of acids from the DNA of different organisms, scientists can determine how these organisms are related to each other and how they evolved. In a similar manner, the textual critic compares texts (a sequence of characters such as letters, spaces and punctuation marks) from witnesses in order to understand how those witnesses are related and how the text evolved through successive copies. Robinson’s collaboration with evolutionary biologists on the one hand (see Barbrook et al. 1998), and

2.4. Methodology of Automated Collation

<p>This] 54 witnesses Carpenter] 54 witnesses hadde] 54 witnesses wedded] 53 witnesses; E wedded 1 witness wedded newe] newe wedded 1 witness, newli wedded 1 witness newe] 26 witnesses; newly 1 witness; omitted 1 witness newe a] a newe 23 witnesses a] 30 witnesses wyf] 53 witnesses</p>	<p>wedded newe a] wedded newe a 25 witnesses wedded a newe 23 witnesses newe wedded a 1 witness E wedded newe a 1 witness wedded newly a 2 witnesses newli wedded a 1 witness wedded a 1 witness</p>
---	---

(a) Base text collation.

(b) Parallel segmentation.

Figure 2.7: Collate — base text collation vs. parallel segmentation (Robinson 2004).

with the editors of the New Testament in Münster on the other hand (Wachtel 2000; Parker 2006), led to improvements in Collate regarding the collation process: the base text was abandoned in favour of a new procedure called ‘parallel segmentation collation’.

The parallel segmentation method was created to deal with three problems of the base text. The main issue of the base text, as used at first by Collate, was that this artificially constructed text was taking too much importance (Robinson 2004). The base text was neither an existing witness, nor did it represent editorial decisions about the text. However, the base text was the main access to the collated text and the variant readings for readers, who could be biased in favour of the base text. Another issue of the base text was that it made it difficult to analyse the relationships between witnesses when using phylogenetic methods (Robinson 2004). The collation results, with a base text, divides overlapping variants into several apparatus units: this prevents the text of all witnesses from being automatically recreated at each point of variation, which was required in order to apply phylogenetic methods. For instance in figure 2.7(a), the apparatus has five separate entries for the reading ‘wedded newe a’ in Chaucer’s Miller’s Tale, line 35, when using the base text method of collation. On the other hand, the parallel segmentation method would group all the readings under one apparatus entry (figure 2.7(b)).

In figure 2.7, the base text apparatus on the left is problematic because it is difficult to reconstruct the text of each witness based on single apparatus entries (figure 2.7(a)). For instance, there is no indication that the two apparatus entries ‘wedded’ and ‘wedded newe’ represent twice the same ‘wedded’ and not two different instances of the word. Consider the entry which says that one manuscript has ‘newe wedded’ instead of ‘wedded newe’: from this particular entry, it is impossible to derive automatically that another witness reads ‘E wedded newe’ instead of ‘wedded newe’ because this information can only be accessed from the previous apparatus entry. Finally, the base text was seriously limiting the display possibilities.

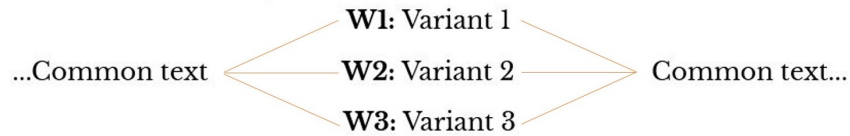


Figure 2.8: Parallel segmentation of a text.

The collation can only be displayed in relation to the base text, and not in relation to a witness chosen by the user (Parker 2000, 34).

The solution to those issues was to remove overlapping variation, by grouping all variants together under a single lemma (see figure 2.7(b)). The advantage is that the base text is not necessary anymore in the output, and the same collation may serve several purposes, such as creating a stemma with phylogenetics methods or creating a variant database (Robinson 2004). Although the base text has been removed from the output, Robinson (2014) indicates that the base text was still used during collation, for the identification of variants.

2.4.3 *The Variant Graph Data Model*

The parallel segmentation mentioned above makes a very useful representation of textual variation. This representation divides the text into segments as in figure 2.8, so that variant readings from different witnesses can be represented in parallel.

Parallel segmentation is one of the underlying data models for the encoding of the critical apparatus in a scholarly edition produced following the TEI Guidelines, P5 version (TEI Consortium eds. 2017c, §12.2.3). Peter Robinson was not the first to propose this kind of data model. Colwell and Tune (1964), two scholars working on the International Greek New Testament Project during the 1960s, have offered a very similar representation of variation in a diagram (or scheme) where variants are grouped into variation-units, as shown in figure 2.9. In fact, Colwell and Tune anticipated that computers would become necessary tools to deal with the New Testament tradition, and they expected that their model would be easily translated into a mathematical format suited to electronic manipulation (Colwell and Tune 1964, 256). However, their contribution remained largely unknown in the wider scholarly community and the variant graph did not appear as such until the end of the 2000s, with a publication of Schmidt and Colomb (2009).

Another attempt at a graph was proposed by Sperberg-McQueen (1989), using the metaphor of a river's delta to illustrate how texts can separate at a variant location

2.4. Methodology of Automated Collation

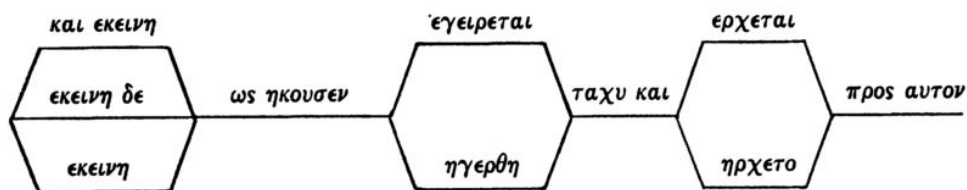


Figure 2.9: An early version of the variant graph (Colwell and Tunc 1964, 254).

and then merge back together like a stream. Sperberg-McQueen (1989) outlines that his graph, the Rhine Delta structure, would not be biased towards a specific base text. The difficulty is to represent such data structure with markup. Markup is not well suited to deal with the overlapping hierarchies of textual variation: markup such as XML enforces a strict hierarchical division of the text in a structure of ordered elements which can be nested but cannot overlap, following the OHCO model (Ordered Hierarchy of Content Objects) elaborated by DeRose et al. (1990). It happens often that textual variants are not best represented in well-defined parallel segments, but rather overlap with other variants and textual features such as the division into paragraphs, lines, etc.

Schmidt and Colomb (2009) were interested in a data structure that would represent efficiently overlapping hierarchies. They describe a graph model for representing multi-variant texts that is directly inspired by the developments in the field of bioinformatics and more precisely phylogenetics. In fact, there have been several attempts to adapt phylogenetic methods from evolutionary biology in order to create stemmata for textual traditions (Barbrook et al. 1998; Roos and Zou 2011). The similarities between textual criticism and bioinformatics led Schmidt and Colomb (2009) to adapt the 'partial order alignment' graph structure of Lee, Grasso, and Sharlow (2002) for the representation of variant texts (see figure 2.10). Partial order alignment graphs had a few characters that were not suited to textual variations, such as multiple beginnings and endings, or redundancy. Therefore, they were combined with a different structure, the PERT graph, which describes a workflow of tasks for project management.

Finally, the variant graph was adopted as data model for CollateX, with a slight modification (Dekker and Middell 2011). A graph is a structure made of nodes — or vertices — and edges that link the nodes together. The variant graph of Schmidt and Colomb places the text of the witnesses along with their sigla on the edges of the graph, whereas CollateX's graph places the text of the witnesses on the nodes of

2.4. Methodology of Automated Collation

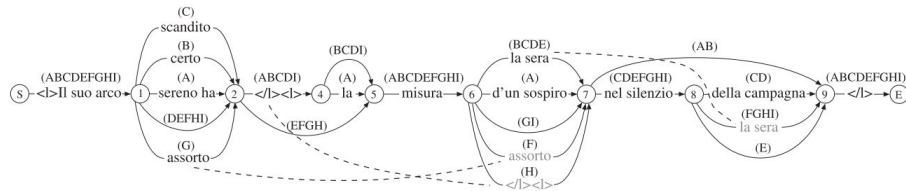


Figure 2.10: The variant graph by Schmidt and Colomb (2009, 502).

the graph, and the sigla on the edges (figure 2.11). One reason for this modification is that it makes it easier to implement in CollateX the separation of concerns of the Gothenburg model (see Section 2.4.4 below). In addition, the CollateX graph model is easily transformed into other visualisations such as a variant table (see CollateX Documentation). To have the text on the nodes of the graph instead of the edges also improves readability (Jänicke, Büchler, and Scheuermann 2014). However, both versions of the variant graph share common characteristics:

- The graph is **directed**: the edges that link nodes have a precise direction, which follow the order of the text. By following the edges associated with the siglum of one witness, it is possible to obtain again the text as it was present in this witness. See for instance in figure 2.11, the text of witness W2 can be retrieved by following the edges labeled W2 and highlighted in red. A path that goes from start to end of a directed graph is called a traversal.
- The graph is **acyclic**: it is not possible to loop back at a previous point in the graph, it must be followed from start to end. If loops were allowed, it would introduce repetitions of words that were non-existent into a witness.
- There is a **start** and an **end** node, which have no textual content. They serve as a marker for the beginning and end of the text, but must allow for the addition of more text between the source and the first node of text (in case a new witness has text at the beginning or end which was not present in the previously collated witnesses).
- **Transpositions** are represented with dashed lines that have no label. The transposition's edges cannot be followed while retrieving the text of a witness, otherwise it would introduce cycles into the graph, which is forbidden (Schmidt and Colomb 2009, 501). According to CollateX Documentation, transpositions are superimposed on the graph but do not integrate well with the graph, i.e., transpositions are rather considered as an added layer to the graph than a part of it.

2.4. Methodology of Automated Collation

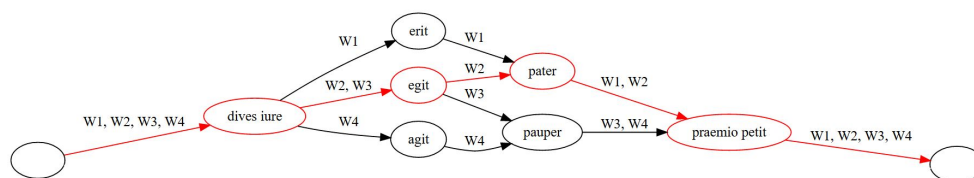


Figure 2.11: Example of a CollateX variant graph. Created with CollateX's online demo <http://collatex.net/demo> (June 5, 2017).

The variant graph of Schmidt and Colomb is implemented in the collation tool *nmerge* (later renamed *Compare*) and the Multi-Variant Document format, while the modified graph of Dekker and Middell is implemented in CollateX and TRAViz, as well as Stemmaweb, but it does not seem to have been used in other collation tools²³. Andrews and van Zundert (2014) have argued that the variant graph could serve as the interface of a scholarly digital edition, replacing the traditional critical apparatus of a printed edition. The idea was implemented to prepare a digital critical edition in the form of a graph database, for the *Chronicle* of Matthew of Edessa, an Armenian priest of the 12th century (Safaryan, Kaufmann, and Andrews 2016).

2.4.4 The Gothenburg Model

The Gothenburg model is now the standard accepted model of an automated collation workflow, and a discussion of the methodology must therefore include a discussion about this model. As said before, the Gothenburg model reflects a separation of concerns: automated collation is divided into smaller tasks, which make the whole process more manageable. These tasks were namely: tokenisation, alignment, analysis, and output (or visualisation). An additional step of normalisation may also be performed, during or after the tokenisation stage.

The description of the Gothenburg model is available in Dekker et al. (2015) and in CollateX's online documentation²⁴. A workshop on automated collation, organised in November 2016 in Amsterdam by members of the DiXiT Network, provided participants with comprehensive materials which are available online as well: it includes a detailed explanation of the model and the issues related to each stages

²³Stemmaweb is an interface to analyse textual variation and stemma hypotheses from collated texts: <https://stemmaweb.net/> (Accessed November 5, 2016).

²⁴<https://collatex.net/doc/> (Accessed May 12, 2017). A similar description is available in the TEI Wiki website: https://wiki.tei-c.org/index.php/Textual_Variance#The_.E2.80.9CGothenburg_model.E2.80.9D:_A_modular_architecture_for_computer-aided_collation (Accessed May 12, 2017).

(Bleeker and Spadini 2016)²⁵. The examples in this section will mostly focus on CollateX and Juxta, since these are the two programs from which the Gothenburg model was conceived.

2.4.4.1 Tokenisation

The first task is to split the entire text of each witness into smaller units, called tokens, to be compared. Tokens are commonly used for lexical analysis in computer science: a sequence of characters with an identified meaning is converted into a sequence of tokens. Those tokens are then further processed by other programs, such as a parsers, for syntactic analysis. In the Gothenburg model, a text is divided into a list of tokens which are textual units (a sequence of characters). According to Dekker et al. (2015, 4), a token is a textual unit at ‘any level of granularity, for instance, on the level of syllables, words, lines, phrases, verses, paragraphs, text nodes in a normalised XML DOM instance, or any other unit suitable to the texts at hand’. The CollateX documentation more explicitly considers a token as a textual unit that ideally carries meaning, thus above the character level. At letter level, phenomena such as transposition are much more frequent and reduce the efficiency of the alignment algorithm. For this reason, collation is preferably performed at a higher level, rather than at character level. Gibson and Petty (1970) have tried to use the sentence level as collation unit. However, they realised that it was very complex to recognise sentence boundaries automatically, and opted instead for a segment of twelve words (Gibson and Petty 1970, 285). Difficulties would arise from ambiguities, such as the presence of punctuation marks inside a sentence (for instance after abbreviations such as ‘Mr.’ or ‘Mrs.’).

Tokenisation is usually performed by a computer script, the ‘tokenizer’, and can be achieved in different ways. First, plain text can be split into words at whitespace. The main issue here is that white spaces are not always valid word separators. For instance, texts may be written in *scriptio continua*, without white spaces, or with an inconsistent word division. This is often the case in Latin: even when words are divided, it is very frequent that word division is not consistent across witnesses. For instance, the words *res publica* (the state) might indifferently be written as two separate words or as one *respublica*. Elements of punctuation can also raise questions. In French, *aujourd’hui* (today) is one word containing an apostrophe inside; on the other hand, the same apostrophe becomes a word separator in *j’ai* (I have). Enclitic particles may be problematic as well. These are particles which are appended at the end of the preceding word. The English possessive ‘s is an

²⁵http://nbviewer.jupyter.org/github/DiXiT-eu/collatex-tutorial/blob/master/unit4/Theory_of_collation.ipynb (Accessed May 12, 2017).

2.4. Methodology of Automated Collation

example. In Latin, the enclitic *-que*, which means ‘and’, is attached to a word so that *respublicaque* is equivalent to *et respublica* or even to *et res publica* (and the state). Should these examples be treated as one token only for the purpose of collation? Should punctuation marks be counted as tokens? Researchers need to consider those theoretical questions and make their own decisions.

Another way to perform tokenisation is to select a list of nodes from an XML document, with an XPath expression²⁶. For instance, the XPath expression `/TEI/text/body//w` would select every ‘word’ element (`<w>`) of a TEI encoded text. In this case, the `<w>` elements are likely added manually, which means that tokenisation is actually prepared by editors who can have full control over the division of the text into tokens. The issue of markup encoding is that it may interfere in the comparison process. For instance, let us consider two sentences, one of which is in roman font, and the other one has been encoded as italics:

Her tanned face turned black at the sound
<i>Her tanned face turned white at the sound</i>

A plain text comparison between these strings would return not only the variant between white and black, but also the ones between Her and <i>Her, sound and sound</i> (Shillingsburg 2014). Here <i> and </i> are elements of markup which interfere in the comparison of the text. For this reason, markup must be removed from the text for collation purposes. However, it may also be valuable to keep the markup context of each words, either for a possible usage during the comparison or for displaying the results in the final visualisation stage. In addition, some markup may convey textual information, such as the presence of an abbreviation or of a correction. Therefore, a tokeniser could take into account the markup context of each word: in figure 2.12, tokens (the letters a, b, c and d) keep attached to them the list of the elements in which they are wrapped. Token ‘c’ is wrapped in an element e2 which is itself nested in element e1. As a consequence, after the tokenisation ‘c’ will keep as a feature its markup context (elements e1 and e2). Tpen2tei, a collection of tools to work with the transcription platform T-PEN, retains the markup context of XML elements when tokenising a transcription for collation with CollateX²⁷.

Finally, if none of the solutions described above are satisfying, researchers should

²⁶XPath is a language used to navigate through elements and attributes in an XML document.

²⁷See the update of September 19, 2017 ‘overhaul of XML literal and XML context data in tokens’ <https://github.com/DHUniWien/tpen2tei/commit/81c92195967f90a14f287dbc7e85eafe87c5f666#diff-1009ffb46f7d4bc3cd106481ea6ffa44> (Accessed October 10, 2018). The transcription tool T-PEN, and tpen2tei, are discussed in Chapter 6.

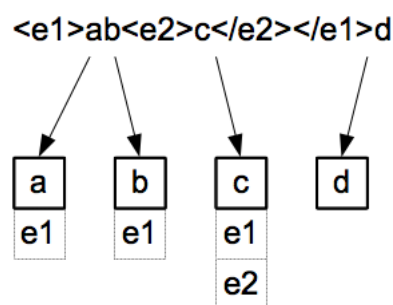


Figure 2.12: Tokenisation with markup context. Retrieved from <http://collatex.net/doc/> (May 12, 2017).

also be able to provide a pre-tokenised instance of their texts according to their needs instead of adopting the tokenisation provided by the tool they are using. In theory, anything could become a token, even non-textual elements (such as diagrams, illustrations, and so on), and could be compared with an appropriate equivalence function: the equivalence function determines when two tokens are considered equivalent for collation purposes. As a basic example, in order to compare drawings instead of words, one could define an equivalence function that compares shape and color instead of letters. If one token has the features 'color: red' and 'shape: square', while a second token has the features 'color: blue' and 'shape: circle', they are not equivalent and should probably not be aligned together. Such equivalence functions are yet to be created, and only text comparison is possible with most current collation tools. In the case of CollateX, it is possible to collate strings of characters, which could represent a more complex structure than a single word. If the Gothenburg model is followed, however, and tasks are properly divided, the modular structure makes it easier to adapt the collation tools to new functions.

2.4.4.2 Normalisation

In digital format, the most basic form of a token is a simple string of characters, a linear sequence of one or more symbols representing letters, but with no linguistic interpretation attached to them. However, the issues discussed in the tokenisation section p. 77 above show that a single string of characters is not necessarily a perfect representation of a token. Tokens may need to include other features, such as a markup context for instance.

In addition, a normalisation feature may be helpful. Collation tools usually offer to normalise tokens in order to minimise what is perceived as insignificant variation: typically, normalisation makes it possible to remove upper case, punctuation or other aspects (such as, for instance, hyphenation or line breaks in Juxta, white space

2.4. Methodology of Automated Collation



Figure 2.13: Examples of incorrect alignments due to hyphenation and whitespace. Created with CollateX's online demo <http://collatex.net/demo> (May 12, 2017).

characters in CollateX) from the tokens that will be compared, so that these would not be considered differences. The first reason to add a normalised form to the original token is to help the collation algorithm to produce an accurate alignment.

The two simple examples in figure 2.13 show how whitespace or hyphenation can lead to what a scholar would regard as a wrong alignment: here the program sees that *re-publica* or *re publica*, in witness W2 on the middle line, is different from both witness W1 and W3, from *republica* in witness W1 as well as *De* and *Republica* in witness W3. With normalisation, on the other hand, it is possible to declare that *republica*, *re-publica* and *Republica* are equivalent, and consequently the three forms of the word would be correctly aligned.

The second reason to include a normalised form is to allow for a better visualisation of significant variants. Significant variants are commonly understood to be the ones which impact the meaning of the text, whereas accidental variations such as orthographic differences, word division and abbreviations for instance, are often neglected for the purpose of a critical edition. However, the transcriptions which serve as an input to collation tools usually contain a diplomatic representation of the witnesses, and not a normalised version of the text. As a result, when variants are displayed in the visualisation, there is no way to distinguish between what is a substantial variation, and what is accidental. Figure 2.14 shows an example of how normalisation would impact the visualisation: the text is Catullus' second poem, transcribed from three manuscripts, and collated with Juxta. On the right, the collation shows a normalised text, while on the left side, orthographic variations and abbreviations were included. In this visualisation called 'heat map', the variant locations are highlighted blue, and a darker shade of blue indicates a stronger disagreement between the text displayed and the other witnesses. It is already difficult with only three witnesses to locate significant variants in the right-side

2.4. Methodology of Automated Collation

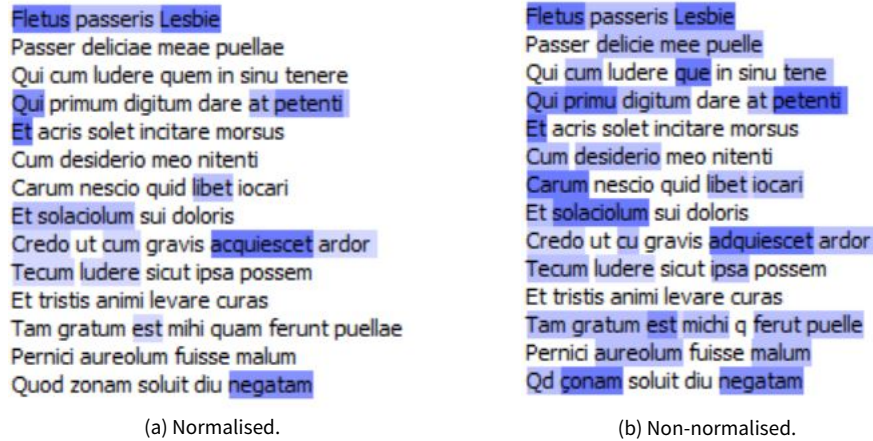


Figure 2.14: Normalised and non-normalised texts. Created with Juxta Desktop application (May 2, 2016).

example where the text is not normalised. If there were many more witnesses collated, the text could become entirely blue, and the visualisation would not be helpful at all. This is especially the case for medieval traditions which have countless orthographic variants.

In traditions with a very large amount of orthographic differences, it may be desirable to normalise the words to their root form (or lemma) instead of normalising words to an inflected form. This process is called stemming. It is also possible to normalise words to their part-of-speech category (POS) or to a combination of lemma and part-of-speech. There are tools called lemmatisers which are designed to do this automatically, such as the Classical Language Toolkit (CLTK)²⁸ or Tree-Tagger²⁹. For instance, the words in the following sentence can be normalised to their lemmas with CLTK:

Original: *quae terras frugiferentis concelebras*³⁰;

Normalised: *qui1 terra frugiferens concelebro.*

Researchers using collation tools must therefore make important decisions regarding normalisation, so as to obtain a good alignment and a useful visualisation. It should be noted as well that the normalisation intended for a good alignment may be different from the normalisation intended for a useful visualisation. The advantage of tokens having both an original and a normalised form is that it is

²⁸<http://docs.cltk.org/en/latest/about.html> (Accessed May 15, 2017).

²⁹<http://www.cis.uni-muenchen.de/~schmid/tools/TreeTagger/> (Accessed May 15, 2017).

³⁰'[Venus, you] who make the fruitful earth teem with life' Lucretius, *De Rerum Natura* I.3-4, translated by Englert (2003).

possible to hide insignificant variants without a loss of information: the original form is always available, should it be needed. The preparation of tokens that include both original and normalised forms is discussed more in detail in Chapter 4 and Chapter 7: for instance original and normalised forms may be encoded with markup elements such as <orig> and <reg> in TEI. Normalised forms can also be created automatically, for instance by transforming upper case characters in lower case. In Section 8.4, I will explain how the collation results can be filtered in practice, to ignore insignificant variants.

2.4.4.3 Alignment

Alignment is the core of a collation program, and also one of the most complex parts. During the alignment stage, the sequences of tokens from the witnesses are first compared in order to find the equivalences (or matching tokens). Tokens are exactly matching when they share the exact same sequence of characters (letters and/or punctuation marks). Then, gaps are inserted in between matching tokens to obtain the best possible alignment. For example, let us consider the three following witnesses:

W1 The brown fox
W2 The brown quick fox
W3 The quick brown fox

When the first two witnesses are aligned, the tool recognises that the words ‘The’, ‘brown’ and ‘fox’ are equivalent because they share the exact same sequence of characters. The three words must be aligned, but ‘quick’ in W2 is not matching any token of W1. Therefore, a gap is inserted in W1 between ‘brown’ and ‘fox’. The alignment is now as follow, and the gap is rendered as a dash:

W1 The brown - fox
W2 The brown quick fox

Then the third witness W3 is compared to the first two. ‘The’ and ‘fox’ are aligned with ‘The’ and ‘fox’ in W1 and W2; ‘quick’ is aligned with ‘quick’ in W2 and with the gap in W1, and a gap is inserted in W3 between ‘The’ and ‘quick’. Now, ‘brown’ in W3 is matching ‘brown’ in W1 and W2, but to align them would mean to change the word order of W3 and place ‘brown’ before ‘quick’. It is not possible to change the word order at this stage³¹, and since ‘brown’ does not match any other tokens in

³¹It is not a good idea to change the order of tokens, because it would change the word order of

2.4. Methodology of Automated Collation

either W1 or W2, gaps are inserted again in W1 and W2. The new alignment would look like this:

W1	The	brown	-	-	fox
W2	The	brown	quick	-	fox
W3	The	-	quick	brown	fox

It must be noted that at this point, the program does not make any assumption about the transmission of the text: the collation tool cannot recognise whether 'quick' was omitted from W1, or whether it was added in W2 and W3. The transposition is not recognised either, but for now it is considered as two distinct operations: the addition or omission of 'brown', first before 'quick', and then again after 'quick'.

There are different ways to perform the alignment, with different algorithms. The witnesses can be compared in a 'pairwise alignment' such that each witness is compared to one witness that is selected as the base text. The variants are then gathered at the end. Another way to compare witnesses is to compare each witness to all other witnesses. The alignment could also proceed by layers, just as we have just seen: each time a new witness is compared, the alignment of the previous witnesses is updated to take into account the new witness. This is called a 'multiple alignment', or more precisely a 'progressive multiple alignment'. Although pairwise alignment is easier to implement than multiple alignment, its main issue is the collation order: depending on the order in which witnesses are collated, the results of the alignment may be affected. The issues and differences between the various approaches will be discussed in the section about collation algorithms.

Finally, there are two notions central to the alignment stage: the use of a heuristic method, and the scoring system. Collation, the comparison of multiple textual witnesses, is a complex task to implement on a computer. In theory, each token of each witness needs to be compared to one another, so as to consider every possible alignment and find the perfect alignment. However, this involves a lot of computations which make the whole process very long, and the computer is not always able to decide which solution is the best one. For this reason, collation tools operate with heuristic algorithms. Heuristics 'provide a simple means of indicating which among several courses of action is to be preferred' (Pearl 1984, 4). Even if heuristics do not guarantee that the best course of action will be preferred, they will do it sufficiently often (Pearl 1984). Heuristic algorithms can solve complex

the text and thus its meaning. A text is an ordered list of words, and it matters in which order they are written.

2.4. Methodology of Automated Collation

problems fast and efficiently, by sacrificing other aspects such as accuracy. ‘Heuristic algorithms are most often employed when approximate solutions are sufficient and exact solutions are necessarily computationally expensive’ (Kenny, Nathal, and Saldana 2014). Therefore, the collation program does not need to find all possible alignments, but rather only one that is considered good enough. This makes it possible to reduce the collation process to a reasonable amount of time. The scoring system is provided in order to implement this heuristic algorithm, and to help the program select a good alignment. When different options for the alignment are present, the scoring system gives more weight to one of the options. An exact match between tokens is the best option, and therefore it receives the highest score. A substitution ranks higher than an addition or omission and therefore would receive a higher score. This ensures for instance that in the two witnesses of table 2.3, ‘quick’ and ‘brown’ are aligned as a substitution, and not as a series of addition and omission. The parameters of the scoring system have thus a very important role to play in the heuristic alignment, and changing the scoring system may result in very different alignments.

W1	The	brown	fox		The	brown	-	fox
W2	The	quick	fox		The	-	quick	fox

Table 2.3: Preference for substitution over addition and omission.

Sometimes, ambiguities arise when the tokens do not match exactly, and the program makes an arbitrary decision. This was the case in our example of figure 2.13 above, when *re-publica* in W2 was not considered equivalent to the other tokens in W1 and W3. Therefore, it was arbitrarily aligned to the first token *De*, even though an editor would obviously have aligned it with *republica*. Transpositions and repetitions as well may lead to ambiguities. If ambiguities cannot be solved with the help of normalised forms, then it is possible to refine the alignment thanks to the analysis stage.

2.4.4.4 Analysis and Feedback

The heuristic approach and the scoring system we have seen above do not always yield a good enough alignment, and the analysis stage gives the opportunity to improve the result obtained from the alignment stage. It can be improved either manually by the user or automatically. It is possible to change the parameters of the scoring system, and give more or less weight to some phenomena. Or the user may predetermine the alignment of some tokens prior to collating the witnesses, which may help the program to align correctly incomplete witnesses, for instance.

2.4. Methodology of Automated Collation

W1	I bought	this glass, because it matches	those	dinner plates	.
W2	I bought		those	glasses	.

Figure 2.15: Comparing exact matches first can result in a wrong alignment. Created with Collatex's online demo from the sample 'I bought this glass, because it matches those dinner plates.' <https://collatex.net/demo/> (October 10, 2018).

A feedback cycle is introduced when additional knowledge is submitted by a user back to the program in order to obtain better results with the next collation.

'Fuzzy matching' is a good example of improvement to the alignment. Fuzzy matching — or 'near matching' — is a technique used to detect if two tokens which do not match exactly are similar enough to be considered equivalent (for instance *re-publica* and *republica*, or any kind of orthographic variant). If fuzzy matching had to be applied to each token, the cost in terms of computations would be too high during the alignment stage. For this reason, it is more effective to compare first exact matches with the heuristic alignment (it is quicker); and later in the analysis stage, fuzzy matching can be applied only to tokens that were not matching previously. However this does not always provide the correct result: in figure 2.15, for example, 'those glasses' is aligned with 'those dinner plates' instead of 'this glass'. Fuzzy matching may require the use of a parameter in order to decide the threshold of token equivalence: when can two tokens be considered similar enough to be equivalent? When should they be considered as non-matching?

Finally, in the analysis stage, the program may interpret some combination of additions and deletions as transpositions: the author or copyist of a text has moved a portion of the text in one witness from one place to another in a new witness so that the copy has a different word order from its exemplar. Transposition can be a very hard phenomenon to detect for a computer, especially when the text involves repetitions. Transpositions can at best be suggested by the program, and the editor's judgement may be necessary to decide what happened in the text (Dekker et al. 2015). Example of a problematic case for a computer (Spadini 2016, 123):

W1	this	dear	boy	has	paid	-
W2	this	-	boy	has	paid	dear

Table 2.4: The issue of aligning homonyms.

2.4. Methodology of Automated Collation

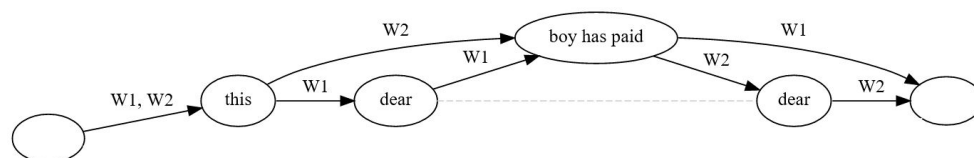


Figure 2.16: Homonyms and transpositions in CollateX. Created with CollateX's online demo <https://collatex.net/demo/> (May 12, 2017).

In figure 2.16, the homonyms 'dear' (an adjective in W1, and an adverb in W2), may be interpreted as a transposition. It is actually how CollateX interprets the relationship between the two tokens, and the transposition is represented as a line of grey dashes. However, since the tokens are different part of speech, a combination of addition and omission may be a better interpretation. In order to make a correct interpretation, the program would need more indications, such as the part of speech of each token.

The analysis stage was included in the Gothenburg model explicitly in recognition that ambiguities cannot always be solved by computing means, and that the intervention of an editor may be needed at some point (Dekker et al. 2015). In this sense, collation is only semi-automatic and not fully automatic.

2.4.4.5 Output Format and Visualisation

The last step in the model is concerned with the form that the result of collation will take. The program may offer various options of output format (XML TEI, JSON, GraphML, etc.) and visualisations in a human readable form (such as a side-by-side display of witnesses in columns, a collation table, a text with critical apparatus, a variant graph, or a heat map with a single text and variant locations highlighted in colour). In this chapter we have already seen examples of visualisations: the collation table (figure 2.13), the heat map (figure 2.14) and the variant graph (figure 2.16). Different collation tools will attach more or less importance to visualisation in human form. On the other hand, some tools which do not perform automated collation will focus on visualisation only, such as the Versioning Machine³² or CATview³³ for instance. Again, the modular approach of automated collation should facilitate the integration of collation tools with visualisation tools: for instance, Juxta implements a visualisation with the Versioning Machine in the web interface Juxta Commons.

³²<http://v-machine.org/> (Accessed May 19, 2017).

³³<http://catview.uzi.uni-halle.de/> (Accessed May 19, 2017).

2.5. Advantages of Computers and Black Box Issue

The main issue with collation visualisations is their often fixed nature. A static visualisation may tend to convey an undeserved impression of correctness or authority (Andrews and van Zundert 2013). The results of the collation tool cannot be challenged or refined, despite the fact that the program sometimes resolves ambiguities by choosing arbitrarily one option, which may not be the best one for a scholar. Therefore, it is important that users do not place too much trust in collation results, but evaluate them critically. The solution to this issue is to provide an interactive interface for a user to inspect, manipulate and eventually modify collation results. According to Andrews and van Zundert (2013), the essential interactions proposed to a user should include at least the following:

- Annotating variants with information on their relationship.
- Combine or split tokens into readings, for a better representation of the variation.
- Update the alignment where tokens are not correctly aligned.

In summary, there are two sides of using a computer to automate the process of collation: it may be advantageous in some aspects (such as rapidity or consistency), but it can be very difficult to understand the alignment algorithms and how they work to produce a specific output. In the following section, we will explore more in detail the reasons why scholars have adopted the computer for the collation process and the issue related to trusting a computer's output, especially when the processes that led to such results are not directly available to the user, as if hidden in a black box.

2.5 Advantages of Computers and Black Box Issue

2.5.1 *Benefits*

It is a common trope, among scholars adopting automated collation, to describe collation as a boring or tedious, and error-prone activity in order to highlight the benefits of working with computers (Gibson and Petty 1970; Shillingsburg 1978; Robinson 1989a; Hilton 1992; Dekker and Middell 2011; Andrews 2012). Automated collation should in particular relieve editors from mechanical and repetitive tasks, thus saving time and efforts which may be better spent in critical activities which the computer cannot perform such as choosing a reading among variants (Robinson 1989a; Marín 1991; Shillingsburg 1996). Computers are also praised for their rapidity and their reliability: computers work faster than human

2.5. Advantages of Computers and Black Box Issue

beings, are more accurate and more consistent (Froger 1968, 218; Hilton 1992, 140; Reeve 2000, 196). However, as we have seen above in Chapter 1, collation is already a critical activity that involves important decisions. In addition, the change of methodology, from manual collation to full transcriptions of witnesses, may not involve a lesser amount of work for an editor. Transcription may be as time-consuming and error-prone as manual collation.

When computers were first adopted in textual criticism, the machine was considered merely as a slave at the editor's service, an effective research assistant (Shillingsburg 1996, 141; Vanhoutte 2010, 121; Pierazzo 2015). In the context of automated collation, the computer can compare texts more quickly and efficiently than the graduate students who used to collate (Gibson and Petty 1970, 280). However, scholars should not limit themselves to replicating manual methods on a computer, but embrace the new possibilities offered by computing methods because it will ultimately reduce the time and efforts needed (Raben 1979, 256). Raben also showed how the adoption of computing methods can improve the task of automated collation (see p. 50 below).

As Father Roberto Busa noted, 'the use of computers in the humanities has as its principal aim the enhancement of the quality, depth and extension of research and not merely the lessening of human effort and time' (Busa 1980, 89). Therefore computers should not be considered only as time and labour saving devices, but also as problem solving (Reeve 2011, 364). Fischer (1970, 300-301) warns that until 1970, Humanities scholar have "let the scope of their questions be arranged by the capacities of the computer rather than use the new possibilities it affords to enlarge their horizons and ask questions which hitherto could not be answered". For Fischer (1970, 304), computers are especially useful in the field of textual criticism. In particular, the computer allows to work with the complete material available and not only a selection, an important fact also highlighted by Fischer (1970) and Parker (2012). Fischer (1970, 308) also cite as advantages the fact that we can investigate orthographical differences or ignore them, and the fact that we can see how often groupings of manuscripts can be found.

In addition, the computer encourages scholars to be more precise, and more systematic (Busa 1980, 89; Marín 1991, 103). Parker (2006, 26) and Dubuisson and Macé (2006, 27) list a series of advantages of computer methods for collation, including among others, being able to reuse the material in various ways (such as adding new witnesses, altering the base text, analysing the relationships of witnesses, selecting the most important witnesses and preparing a critical edition),

2.5. Advantages of Computers and Black Box Issue

collaborating efficiently, and presenting the data according to the editor's need in different visualisations. Collaboration between different projects may be greatly enhanced by a shared methodology (Parker 2006), and data exchange may be easier with a shared format (Dubuisson and Macé 2006).

2.5.2 Challenges

Although these benefits make automated collation worthwhile, it would be risky to trust the results of algorithms and tools because the computer is considered as more objective and thus reliable than a human editor, as Marín (1991) suggested. On the contrary, human biases are built into the tools used by scholars, but hidden by the 'black box' that is a computer program (ter Braake et al. 2016): Robinson (1994) experienced this when he tried to use his first collation tool Collate with data from other projects (see p. 100 below). Sculley and Pasanek (2008) have shown that using three different algorithms for data mining to interpret a text could lead to different and contradictory conclusions. Even a simple function such as a word count may yield different results according to different tools (van Ossenbruggen 2015). For this reason, it is important to always be critical of results obtained with computational methods, and not draw hasty conclusions based on those results, but rather compare the outputs of different methods.

Another issue related to hiding the research process behind the black box of a computer is that research is not reproducible (Marwick 2015). If a scholar's results cannot be recreated by someone else, how can these results be reliable? The problem is that much of the process that led to a specific result is kept private, while only the end result is publicly available. The main objections to publishing the research process are first the time necessary to clean up the files and prepare an appropriate documentation, and second receiving credit for this work (Marwick 2015). There has been some effort, especially in the scientific community, to open up the research process to other scholars. For instance platforms such as Figshare allow for citing the outputs from various stages of research that happen before the final publications. However, these platforms are not yet widely adopted or recognised, especially in the field of the Humanities³⁴.

2.5.3 The Black Box Issue

Black box refers to a device or a tool for which the exact functioning is unknown: a collation program, for instance, takes transcriptions as an input and generates an

³⁴Figshare is an online repository for sharing all kind of research outputs, which can be cited by others <https://figshare.com/> (accessed May 22, 2017).

2.5. Advantages of Computers and Black Box Issue

output, but the algorithm that produced the output, its implementation, is hidden from the user. It can be argued that manual collation is already a kind of black box, since collation is not often published, and the exact procedure followed by the scholar is rarely explained: there are few descriptions of manual collation procedure, especially regarding what to include in the collation (see Chapter 1). Andrews (2017) argues that automated collation tools are a different kind of black box than (for instance) tools used to create a stemma with algorithms, because the collation tool does not make any judgement about the correctness of the alignment. This decision is left to the editor, who should review the alignment and correct it when necessary. However, scholars should be aware that different tools and algorithms will produce different alignments: for instance CollateX may be a good solution to deal with transpositions, but the Classical Text Editor may be a good solution to deal with widespread orthographic variation. This issue becomes especially important if the collation results are used as an input to a phylogenetic program in order to create a stemma, since different alignments can in turn generate different stemmata.

The best way to counter the black box issue may be to document very accurately what is done. For example, Reeve (2011) complains that Viré (1986) does not give enough information about her procedure and reasons in classifying Hyginus manuscripts, so that it is difficult to evaluate some parts of her work. Sculley and Pasanek (2008) recommend a set of best practice for data mining which can be expanded to any field adopting computational methods: to always make assumptions explicit, use multiple methodologies, report failed trials, since they can bring valuable information, and to engage in peer-review of computational methodology. Chambers et al. (2017) also highlight the need for a detailed documentation on how to use tools and their functionalities. If this information is absent, it may be difficult for users to trust the results obtained through a tool, and its usefulness with regard to solving a specific problem.

Collation tools and algorithms have been described in various levels of detail, and several criteria to evaluate them have arisen. In the next section, we will examine how to assess the various collation tools which have been created. As we will see, the progress between early collation programs and the latest tools means that in practice, different criteria might apply to the programs across different periods of time.

2.6 Comparing collation tools

In this section, I will present the criteria which will serve as a framework to evaluate collation tools, instead of describing those tools and how they work one by one. Indeed, many of the programs listed in the appendix are now obsolete; even the most up-to-date tools such as Juxta or CollateX are still under development, and may further evolve in the near future, while the latest tools created may not yet be available. However, a set of criteria would not only be helpful for scholars to find the software most convenient for their needs, but it would also facilitate the comparison of potential new tools against already existing solutions. As noted by Siemens (1994), when comparing three collation tools for PC machines, there may not be one universal solution for automated collation. The best tool to choose will depend on the kind of text to collate, specific problems related to a textual tradition, and the purpose of the collation project. Consequently, this section must be set in the broader context of tool criticism.

Tool criticism is emerging as an answer to the concerns about the growing use of tools in the field of Digital Humanities (Gibbs and Owens 2012; Chambers et al. 2017). While more and more scholars adopt digital tools and methods in their workflow, there is a lack of communication between the production side (documentation about the tools' functioning and purpose) and the side of users (who uses the tools, to do what, how efficient is the tool). As a result, tool criticism is meant to provide a framework to analyse and evaluate tools, and develop critical thinking about tools. To this end, several sets of guidelines and criteria have been proposed both for developers to prepare better tools (and for funding bodies to support better tools), and users to select the most appropriate tools according to their need (ter Braake et al. 2016; Chambers et al. 2017). It must be noted here that there is no clear distinction between developers and users, as it is possible that the same person is both developer of a tool and user of their own tool or other tools.

The criteria proposed by Chambers et al. include four broad categories: usability (user experience, interface), documentation (all information pertaining to how the tool work, its purpose and target audience, the algorithms implemented), maintenance (integration of user feedback in the tool development) and flexibility (extent of application). ter Braake et al. (2016) attempt to equip historians with a set of criteria to help them assess the values of the tools that they use for research purposes. They focus on questions such as understanding the barriers that digital methods introduce between scholars and primary data, discovering assumptions built into tools that are necessary to interpret the results, verifying the tool's results by repro-

2.6. Comparing collation tools

ducing its results, and so on. Data appears as an important aspect in tool criticism. The format of data may influence a scholar's choice for a particular tool (Chambers et al. 2017), and on the other hand, researchers should consider if the data they are using are best suited to answer the question at hand (ter Braake et al. 2016).

Regarding collation software, little has been done to formalise criteria for comparison and evaluation of the tools. However, several contributions have been published, which compare various collation tools (Gilbert 1973; Hockey 1980; Siemens 1994). In addition, Hans Walter Gabler's *Remarks on Collation* (2008) reflects upon the fundamental requirements of collation tools: the paper explores requirements according to different groups of users, such as the textual critic, the editor, and the literary critic. From a textual critic's point of view, the tool should help users to provide accurate input (i.e. the tool should make it possible for users to correct errors introduced in the transcription), persistence of coordination and intelligence of coordination, that is, the algorithm should be efficient enough to perform a good alignment, albeit always providing users with the possibility to interactively correct the alignment output. The editor would require additional features, such as a critical apparatus presentation, and being able to locate variant readings in their broader context (i.e. a parallel text presentation). Building on Gabler's remarks, a survey was carried out for the Modernist Versions Project, which compared several automated collation tools according to a set of criteria, in order to adopt a tool for collation in their own workflow (Huculak and Richardson 2013). Having identified the different activities involved in collation, Huculak and Richardson devised a series of nineteen questions to ask when testing a tool: for instance, they ask if the program is open source, what are the input and output formats, what algorithms are used to align texts, or what are the manipulations and visualisations available after processing. Regarding collation algorithms, Dekker et al. (2015) have proposed criteria to evaluate existing algorithms, and useful considerations about the assumptions built into CollateX's code can be found in van Zundert and Dekker (2017).

In addition to tool criticism, van Zundert and Dekker (2017) highlight the need for code criticism in the Humanities, in order to evaluate the scholarship nature of code and software. van Zundert and Dekker (2017) define code criticism as asking 'What does this code do?' They argue that the key to revealing the scholarly nature of code is the difference between 'enabling' and 'performing' a scholarly activity. To demonstrate this, van Zundert and Dekker (2017) compare two tools: eLaborate and CollateX. On the one hand, eLaborate enables, and even facilitate, transcription, but this activity is still performed by a scholar. On the other hand

CollateX makes decisions about textual alignment during collation, and in that sense CollateX performs a scholarly activity. However, van Zundert and Dekker (2017) recognise that it can be difficult to ascertain whether scholarly decisions are indeed delegated to code: code can be hard to read, assumptions are usually tacit, and it may be impossible to deduce the conceptual model by looking at the code alone (van Zundert and Dekker 2017, 128). Code is not always available for collation tools, and it would be difficult to interpret assumptions built into the tools without discussing with their creators. Furthermore, the capability of ‘reading’ code is limited to those that can read code, which is a small number with respect to those that need to evaluate code. For these reasons, I have decided to focus on broader criteria to compare and evaluate collation tools, and help scholars choose the tool that is best suited to their needs. Those criteria can broadly be divided into four categories:

1. Interface
2. Data preparation
3. Collation process
4. Analysis of the results

I will now consider the issues related to each category, and how the various collation tools have dealt with those issues. Since collation algorithms are central to collation tools, and yet very complex, I will examine them in a separate section (below).

2.6.1 *Interface*

The first contact of a user with the collation tool is the interface. There are several issues related to the interface, such as the degree of user-friendliness: what is the level of technical knowledge required to use the tool? Is there an appropriate documentation? What are the installation requirements, in terms of the equipment needed or difficulty to install the tool? On which platform(s) is it available: PC, Mac, Linux, or as a web-based application? What is the programming language, and does it have an influence? Is it open source, or under a commercial license? etc.

It is difficult to evaluate the interfaces of early collation tools, since they are no longer available. From Peter Shillingsburg, we learn for instance that UNITE was the most user-friendly program of those available at the time (Shillingsburg 1996, 140). The developers of the Donne Variorum program, DV-COLL, recognise that

2.6. Comparing collation tools

humanist scholars can be ‘downright technophobic’ and therefore devoted special efforts to make their program user-friendly (Stringer and Vilberg 1987, 89). The *URICA! II* system was presented as a simple and easy to use tool for collation of modern texts, as opposed to Robinson’s Collate whose complex user interface was designed for rich medieval traditions (Hilton 1992, 143). Stringer and Vilberg (1987, 86) also argued that a multiplatform tool was a highly desirable feature. However as for now, the DV-COLL program is available only on Windows machines, while most of the other tools currently available are platform independent. In this regard, Juxta is the tool which offers the most possibilities, with a desktop application for Macintosh, Windows and Unix, a web service and two web interfaces.

While web interfaces require no installation, other tools involve a more complex process: TEI Comparator was in particular noted for its difficult installation requirements including a database and a web server (Huculak and Richardson 2013). Although CollateX requires only the installation of Java in order to be used (CollateX Documentation), it is not in reality so simple: Wesley Raabe, a scholar at Kent State University working on modern texts such as *Uncle Tom’s Cabin*, describes in a couple of blog posts his experience with installing CollateX on a Macintosh, and the various issues related to updates of both the Mac Operating System and of CollateX itself (Raabe 2014, 2015). The experience shows the issues faced by a literary scholar who is working on his own. It contrasts also with the opinions of Huculak and Richardson (2013), who included a computer scientist in the group that reviewed collation tools: Huculak and Richardson underlined the exceptionally clear documentation of CollateX, while Raabe (2015) speaks of a ‘technical hellscape’ and reflected on the need to attend a CollateX workshop in order to make CollateX work properly.

The choice of a programming language was discussed in the context of several tools. Dearing (1962, 257) argues that the best choice for a programming language, at least for beginners, is the one most familiar to the staff of the computer center on their campus, because it will be taught free of charge and advice will be easy to obtain. On the other hand, Dearing also recognises that it may be better to choose a language well established in the wider academic community to share a program with other scholars. Later on, Petty and Gibson as well as Peter Robinson chose the SNOBOL language to develop their programs because of its capacity to handle text and perform pattern matching or string concatenation (Robinson 1989a, 101; Gibson and Petty 1970, 281). In addition, SNOBOL was taught in Oxford at the time when Robinson was working on his program, while Petty and Gibson received the advice from programmers at Princeton University who were developing Snobol

4. Twenty years later, Andrews (2009, 43) turned to the language Perl for the same reason: it was the best programming language at the time to perform text processing operations. Perl was also supported by a wide user community, and many modules of code were already freely available on the Comprehensive Perl Archive Network (CPAN)³⁵, so that it was not necessary to write the whole program when a problem could be solved by an existing module.

The TUSTEP modules were written partly in SIMULA and partly in assembly language³⁶. It requires users to learn a scripting language to execute the program, TUSCRIPT, which is quite complex and necessitates training (Huculak and Richardson 2013). In addition, the documentation is available only in German, a factor that can discourage non-German speaking users, despite efforts to translate the documentation and some commands to English³⁷. For these reasons, the developers of TUSTEP implemented a new front-end interface to the program called TXSTEP. The new interface makes TUSTEP available from an XML editor such as Oxygen, which is a familiar interface for many scholars working in the field of digital scholarly editing. Some annotations and instructions are available in English as well. However, TXSTEP is not yet successful in its goal to help overcome the language barrier and provide a self-teaching environment (Huculak and Richardson 2013).

As Robinson's program Collate evolved, so did the choice of programming language: while the first iteration of the program was used by Robinson himself only as a command-line tool, the next version Collate was meant for a wider use and needed a proper graphical user interface (GUI). So the programming language C was chosen, among other reasons, because it was highly compatible with Macintosh operating systems, which provided the best user interfaces in the 1990s (Robinson 1994, 37). Although C may have permitted to port Collate to another operating system, the program remained available on Macintosh only. When the Macintosh systems switched from 'Classic' to 'OS X' in 2001, the task of porting Collate to the new version did not seem worth the effort (Robinson 2007b). Instead, the new program CollateX was designed and Robinson turned to Java for its availability on multiple platforms with complex graphical interfaces, its text and string-handling features, and its popularity among developers of XML (see Robinson 2007b). Since 2014, the development of CollateX continues in Python, which is a relatively easy programming language to learn and is gaining popularity in the Digital Humanities

³⁵<https://www.cpan.org/> (Accessed June 06, 2017).

³⁶<http://www.txstep.de/prot/prot19.html> (Accessed June 06, 2017).

³⁷The documentation was first translated to English in 1989, but is now out of date (Nyhan and Flinn 2016), and later in 1991 TUSTEP commands and messages were also translated to English (Ott 1991).

community (Dekker 2014).

Almost every collation tool created since 2000 is open source or freely available. One notable exception is the Classical Text Editor (CTE), which is under a commercial license for individual scholars or institutions. A yearly subscription is available as well as a free 30-day demo version. One of the Juxta web interfaces, Juxta Editions, offers subscriptions plans with additional features such as web publishing and collaboration with other users³⁸. Other collation tools have not been made publicly available: LERA and LAKomp, Prabhed, or the Collator program in the Multi-Variant Editor for Documents (MVED). The reason why LERA/LAKomp have not been made open source is that a high level of technical knowledge is required to install and maintain the tools; however both tools can be made available on demand (Markus Pöckelmann, private email). The collation program created by Roelli and Bachmann was not published, but is also available from Philipp Roelli upon request. Among the latest tools, iAligner's code is available on Github³⁹, and although eComparatio is not yet ready for use, it will be made available as a free online service for researchers (Schubert et al. 2016).

Finally, the interface issues may seem secondary compared to the other features of a collation tool (such as the amount of data preparation, the efficiency of the collation process, or the output obtained). However, the interface may be the decisive argument to choose one tool over another. For instance, Huculak and Richardson (2013) recommended the use of Juxta over Collatex, despite the fact that Collatex offered more choices of algorithms. The reason for preferring Juxta was in particular the user-friendly interface for beginners, while Collatex required more technical knowledge to operate. Siemens (2009) notes that one of the most successful tool for editing the *Henry VIII* Manuscript and the *Devonshire* Manuscript was the Versioning Machine: although it does not perform automated collation in itself, it is an open source tool which does not require the installation of any plug-in.

2.6.2 Data preparation

What is the input to a collation program? Any text to be collated with a computer needs to be prepared in a digital format in some way. As we will see, OCR is still difficult for manuscript documents which need to be transcribed, and even a printed text which has been scanned and optically recognised would need to be corrected. Moreover, the text may need to be tagged according to some scheme, such as the TEI for instance. Finally, a normalisation may be performed on the text,

³⁸<http://www.juxtaeditions.com/plans> (Accessed June 07, 2017).

³⁹<https://github.com/OpenGreekAndLatin/intra-language-alignment> (Accessed June 07, 2017).

2.6. Comparing collation tools

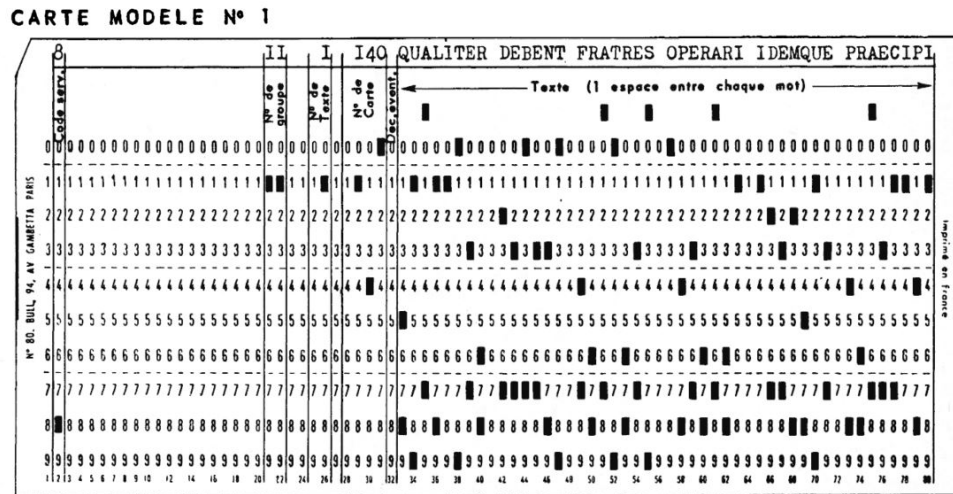


Figure 2.17: Example of a punched card (Froger 1966, 143)

to unify orthographic variation, which could result in a loss of information if the original form is not recorded alongside. For the earlier programs, the amount of data that computers were able to process would require further preparation, in order to divide the material into manageable units of text.

The computer program needs to receive the texts to be compared in a proper format, that is a machine-readable format. For the first generation of collation tools, it meant that texts had to be transcribed on punched cards. Punched cards are paper cards which contain information encoded by the presence or absence of holes perforated in specific places on the card. Computers treat every piece of data as bits, i.e. sequences of zeros and ones, and the absence or presence of a hole in the card is how data is translated into zeros and ones so that the computer can read and understand it. In figure 2.17, you can see an example of a punched card.

The line at the top of the card shows in plain text what is actually encoded in the punched card, for a human reader to check it. Then the card is read by a machine, column by column. The combination of holes in various lines of a column translates as a letter, and the absence of any hole translates as a blank space. In addition, only a restricted set of characters were available: Froger could not distinguish between upper or lower case letters for instance, and any special characters such as accents or punctuation marks would need to be encoded as a pre-defined combination of letters, arabic numbers and full stop (Froger 1966, 144). Cabaniss had more symbols available on larger punched cards, but needed for instance to use the dollar sign before a word to indicate that it would start with a capital letter (Cabaniss 1970, 3).

One card can encode the text of one witness, and only a limited portion of the text: for instance Froger could encode only forty-eight characters on each card (Froger 1968, 232). Strategies were needed to know when a word is split between two cards or not: Froger represented this with a blank space at the end of the card if the word is entire; the absence of a blank space at the end meant that the word was divided between this card and the next. As a result, a very large quantity of cards were needed to encode the full text of many witnesses, which could make transcription a very long and complicated process. Even when magnetic tapes started to replace punched cards as an input method, scholars would still prefer punched cards for a variety of reasons: punched cards were easier to check and correct (Dearing 1970, 264), more durable and less expensive (Widmann 1971b, 57), or better suited to a repeated usage during testing phases (Gibson and Petty 1970, 281). The whole process of transcribing texts into punched cards was so cumbersome that input was actually considered the major issue of early collation programs, and more generally of the use of computers in the humanities (Gibson and Petty 1970, 300). In fact for a long time, automated collation was considered possible but not practical, in particular because of the input issue and high costs associated with it (Oakman 1972, 345; Robinson 1994, 33).

However, the appearance of user-friendly personal computers and progress in input methods later encouraged again scholars to engage in automated collation (Shillingsburg 1978; Robinson 1994). Two options were available to create a machine-readable input: scanning pages with an Optical Character Recognition device (OCR), or typing the texts by hand on a keyboard. The possibility that a computer could scan a page and recognise the text automatically has raised the hopes of many scholars (Andrews 2014b). Despite the optimism of Robinson (1994, 33) who suggested that transcription could soon become unnecessary, even for manuscript handwriting, OCR has faced many issues.

Froger (1968, 277) remarked already that handwritten manuscripts were too irregular for a scanner to recognise the text, due to abbreviations and ligatures for instance. OCR would fail especially when corrections or erased words would occur, which are the most interesting features for an editor. Handwriting recognition has not been solved yet (Andrews 2014b, 178), but the problem is being actively addressed (Fischer et al. 2009; Naji and Savoy 2011; Arvanitopoulos Darginis and Süssstrunk 2014). It seems that the recognition of handwritten medieval manuscripts may be a simpler task than for modern handwriting (Fischer et al. 2009, 142).

2.6. Comparing collation tools

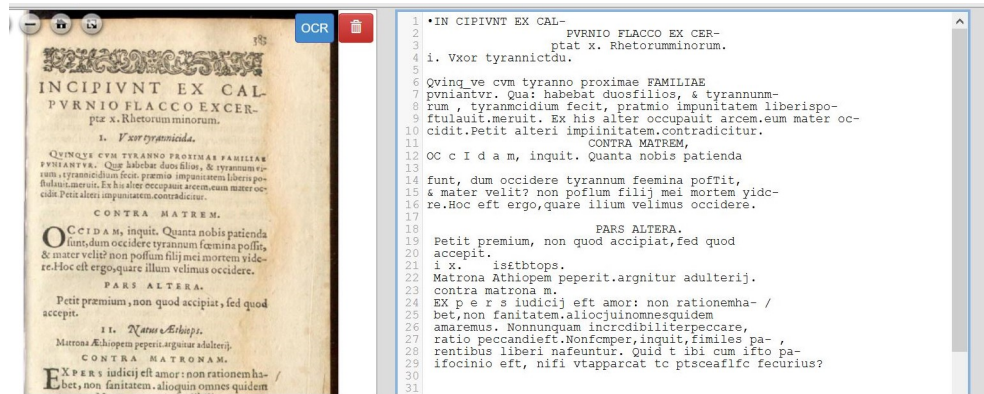


Figure 2.18: OCR with ABBYY FineReader. Created in Juxta Editions (June 10, 2017).

In the case of early printed texts, the OCR option would not prove practical either: scanners would not deal well with the irregularities of older printed editions, which are the primary sources of many scholars (Shillingsburg 1996, 137). Shillingsburg notes as well that scans may be more helpful when correcting the transcription, because scanner's errors are in general 'goof' nonsense that are easier to spot than the errors of a typist. Recent efforts in improving OCR accuracy for critical editions of Classical texts (Boschetti et al. 2009) may have proved more successful: the tool iAligner has been used to compare good OCR outputs and to correct them (Yousef, Palladino, and Crane 2017). Juxta Editions, in its free account, offers the possibility to apply an OCR tool to a limited number of pages. The OCR tool used is ABBYY FineReader, an industrial leader in the field⁴⁰. However, the example below shows that ABBYY FineReader was far from perfect for Calpurnius. The first page of Calpurnius' *editio princeps* was scanned with ABBYY FineReader in Juxta Editions, and the result was not really usable. For instance, many letters are not properly recognised, especially the long *s*, or the diphthongs *ae* and *oe*, and the last five lines are barely readable. The words are not well separated with blank spaces, the tail of an upper case *Q* in line 6 was transcribed as an underscore, a mark in the margin was transcribed as a slash / in line 24, and so on. In fact, so many corrections were needed that it seemed more practical to transcribe the page by hand directly instead of correcting the OCR output (see figure 2.18). Another tool, Transkribus, uses ABBYY FineReader to provide handwritten text recognition services, when enough material has been transcribed by hand (see Chapter 6).

Several collation programs have been criticised for their rigid input format requirements (Gilbert 1973; Robinson 1991). For instance, Widmann's program would

⁴⁰<https://www.abbyy.com/en-eu/finereader/> (Accessed June 09, 2017).

2.6. Comparing collation tools

require a precise line numbering system, so that each line of Shakespeare can be compared in 60 editions. In practice, it means that an initial collation must be done by hand, in order to correctly pre-align the editions (Hockey 1980, 148). While it may be a good solution for the text of Shakespeare's play *Midsummer Night's Dream*, it would not be practical for a medieval prose text: 'the task of pre-editing alone might require nearly as much time as traditional methods' (Gilbert 1973, 142). The problem of rigid input format was not limited to Widmann's program. UNITE and DV-COLL were both criticised by Peter Robinson for the same reason (Robinson 1991, 85, note 20; 1994, 44, note 7). UNITE did not require 'pre-editing' with the addition of tags or labels, and could cope with any transcription format according to Marín (1991, 109). However, each file containing an encoded text must be formatted in a specific way, with text identifiers at the beginning of each file, and stanzas of maximum 5 lines must be separated by a blank line, and so on. The exact requirements are described in Marín (1991, 110). The system of stanzas makes it difficult to use UNITE for any other kind of text. Prose texts cannot be collated at all, and even poetry would be difficult in the case when the text is not divided into stanzas, such as Chaucer's *Canterbury Tales*. DV-COLL requires as well that input files follow a strict format, containing five items: a header with metadata, the title of the poem, the poem written line-by-line, a potential subscription, and finally a footer to record particular features of the transcription (Stringer and Vilberg 1987, 84).

When Robinson tried to use his first program Collate on other texts than an Old Norse saga, the results were not as good as expected (Robinson 1994, 36). The reason was that many assumptions about the text, its content and its structure, were incorporated into the program. If the input files did not comply to specific requirements, it would affect the collation results. Consequently, when Robinson designed Collate, he included as a main principle of automated collation the fact that 'texts for collation must not be rigidly formatted, but can be richly marked up' (Robinson 1994, 36). This would leave as much liberty as possible to scholars, and ensure that Collate could deal with many different kind of texts. A markup scheme was designed for Collate, of which only one element is mandatory: an identifier of a block, so that the text is divided in blocks to be collated together. The length of these blocks is not pre-defined, and can vary from 1 to 32,768 characters. Other markup tags are available to indicate a page break, abbreviations, or add an editorial comment (Robinson 1994, 38-39).

Recent programs have become more flexible in terms of input format, but not all of them offer more than one option. The most common input formats are plain text

2.6. Comparing collation tools

and XML. For instance, the TEI Comparator works only with XML input. On the other hand, Philipp Roelli's program takes plain text where the orthography should be normalised already (in order to find significant differences only), and the text should start with the witness' siglum followed by eight blank spaces and a vertical bar |. TRAViz also has a fixed input format in the form of an array in JSON format⁴¹: each item in the array is an edition, which contains both an identifier (or siglum), and a plain text version of the witness. The Classical Text Editor accepts plain text input and XML as well. Nmerge can deal with plain text, XML (especially the TEI XML encoding), HTML (including EPUB) and some custom formats⁴². According to Yousef, Palladino, and Crane (2017), iAligner allows for input in plain text or in a tabular format, but not in XML. The tabular format can be a CSV file, where each witness is in a separate column, and each line represent a specific block of text (in which case, the witnesses need to be pre-aligned before collation)⁴³.

In TUSTEP it is possible to enter plain text directly into the program, or to import data from a different format. TUSTEP Handbook does not make it very clear which format can be imported and how the data will be treated, in the case of an XML input for instance. The documentation seems to focus rather on the encoding scheme (ASCII, ISO, utf-8, etc.) and how to transform it into TUSTEP data⁴⁴. According to TUSTEP Wiki, it should be possible to import files in XML, HTML and Microsoft Word documents⁴⁵.

Regarding the input format, Juxta seems to be, again, the tool which offers the most options: plain text in several formats (typing directly in Juxta, simple text files, rich text files, Word or OpenOffice documents, pdf and epub documents), HTML files or links to a Wikipedia web page are all valid inputs in Juxta Commons⁴⁶. In the desktop application, it is possible to add plain text or XML documents, whereas in the new interface Juxta Editions, it is possible to transcribe directly in the TEI Lite format, with a limited set of tags, or to import an XML document.

Both nCritic (Andrews 2009) and CollateX take three different input formats: plain text, XML TEI and JSON. The JSON format of CollateX comes in two different

⁴¹JavaScript Object Notation (JSON) is a lightweight data-interchange format. It will be described in detail later in the thesis (Section 7.1.2sec:collation-json-structure).

⁴²See the Ecdosis website: <http://ecdosis.net/main/node/14> (Accessed June 13, 2017).

⁴³An example can be found on iAligner Github code: https://github.com/OpenGreekAndLatin/ILA_python/tree/master/examples/data (Accessed June 13, 2017)

⁴⁴See the manual on the #UMWANDLE command, p. 236: <http://www.tustep.uni-tuebingen.de/pdf/handbuch.pdf> (Accessed June 13, 2017)

⁴⁵<http://tustep.wikispaces.com/Grundlagen+Import+-+Export> (Accessed June 13, 2017).

⁴⁶A link to a Wikipedia page lets users see the different revisions of this page collated together.

versions: the whole texts of the witnesses may be included in a JSON array, which is the same structure described above for TRAViz. Or, the second option is to provide a text that is already divided into tokens. The advantage of this input format is that each token can receive a number of additional properties which will be ignored during collation, but will still be available in the output. While XML files collated are usually stripped from all kind of markup, the pre-tokenisation in JSON offered by CollateX allows for retaining information about a token and its context. This is a useful feature, as we have seen in the tokenisation section of the Gothenburg Model (p. 77 above). For instance, a token may include a property that saves the XPath to an XML node. It is also possible to attach all kinds of information to a token, which can be extremely useful when analysing and visualising the output, such as its location in the witness, or a link to a digital facsimile, and so on (see Chapter 4 and more specifically Section 7.1.3 in Chapter 7). This option of input format is a decisive advantage of CollateX over other tools, and it was adopted in various projects such as the Beckett Digital Manuscript Project, the International Greek New Testament Project, and the Digital Mishnah Project, as well as the preferred collation method in this dissertation.

2.6.3 Collation

The issues regarding the collation process may include: a possible limitation to one kind of text such as poetry, the number of texts that can be processed together by the software, the duration of the process (is it sustainable?), and how the process is performed: can it be interrupted? How to deal with mistakes in the collation? Does the whole process need to be started all over again if there is a mistake? Is it part of a modular infrastructure? What is the algorithm behind the alignment, is it pairwise, or a multiple alignment?

Shillingsburg (1996, 135) stresses that ‘every stage in the process must be interruptible, reviewable, revisable, and, if necessary, re-doable’. Indeed, an issue of the early collation tools was that they did not store the results of collation. If an error occurred during the process, it was necessary to start again. This requirement came at the time when interactive collation was considered to be the best option (Hilton 1992, see; Robinson 1994). However, interrupting the collation process is not necessary anymore.

Poetry is easier to collate with computer programs than prose, for obvious reasons: the structure of lines allows for short, well-defined units of text to be compared. This is not the case in prose text, where paragraphs are randomly divided in lines, which are different in each witness. The first two programs created to deal with

2.6. Comparing collation tools

prose were both published in 1970, by Petty and Gibson, and by Cabaniss. However, until the end of the 1980s, tools limited to poetry were still prepared, such as DV-COLL (1987) or UNITE (1989). Limitation to poetry is not the only issue: many programs were built to edit a specific textual tradition, and thus are not flexible enough to be adopted by other scholars. In fact, few collation tools have been reused later for other projects. The first tool to be successfully adopted for multiple projects is TUSTEP: according to the International TUSTEP User Group, over nine hundred editions have been prepared with TUSTEP⁴⁷. URICA! was installed in a number of machines according to Hilton (1992). Peter Robinson's Collate has been used in very different textual traditions, from Chaucer to Dante, the Greek New Testament or Old French and Sanskrit texts (see Robinson 2009). Collate was replaced by its successor CollateX for the edition of the International Greek New Testament Project in Birmingham (Houghton 2013). CollateX is also implemented in the Beckett Archive project, the Digital Mishnah Project and a series of editions which make use of the Virtual Manuscript Room Research Environment⁴⁸: CollateX seems to be adopted by big projects with support from professional developers. Juxta, on the other hand, seems to be more popular for classroom use: a number of blog posts describe how Juxta was used in classroom context (Wheeles 2013; Poulos 2014; Ravy 2015; Jakacki and Faull 2015). The Classical Text Editor (CTE) has been successfully used to prepare critical editions, however it is impossible to know to what extent these editions make use of the automated collation solution provided in CTE⁴⁹.

The number of texts collated was highlighted by Hockey (2000, 125) as a major issue of automated collation. Some programs were limited to collating two texts at a time (Gibson and Petty 1970; Cabaniss 1970), sometimes with an added facility to merge — or 'conflate' — the variants obtained through pairwise comparisons (Hilton 1992; Shillingsburg 1996). From 6 witnesses in UNITE (Marín 1991) to up to a hundred in DV-COLL (Siemens 1994, 210), the number of texts processed at the same time could vary widely from one tool to the next. In more recent tools, that number may still be limited: Juxta Commons can handle only up to fifteen witnesses⁵⁰. For instance this was an issue for Zeevaert (2015) who needed to collate sixty-three witnesses, a number far exceeding what Juxta allows. On the other hand, Robinson's Collate had to dramatically increase the number of texts handled in

⁴⁷A list of critical editions prepared with the help of TUSTEP can be found here: http://www.tustep.uni-tuebingen.de/ed_eng.html (Accessed July 13, 2017).

⁴⁸<http://vmrcr.org/> (Accessed July 13, 2017).

⁴⁹See here a list of editions created with CTE: <http://cte.oeaw.ac.at/?id0=pub> (Accessed July 13, 2017).

⁵⁰According to the user guide: <http://juxtacommons.org/guide> (Accessed July 14, 2017).

order to collate near to two hundred witnesses (Wachtel 2000, 47) and later over a thousand transcriptions for the Greek New Testament (Parker 2006). CollateX may be able to process virtually any number of texts with blocks of paragraph-sized text, but the speed of collation degrades with larger blocks (Dekker et al. 2015).

To compare various algorithms and judge their effectiveness, it was a common practice to translate the algorithms into the same programming language (Cannon 1976), and then evaluate how well they performed in terms of speed, quantity of text processed and percentage of correct matching (Gilbert 1973; Hockey 1980). For instance, Gibson and Petty (1970, 287) claim that over ninety percent of matching sequences of text can be aligned with OCCULT. However, Hockey (1980, 150) shows that it may be an overestimation: many short variants could still result in a long matching sequence, hence preventing OCCULT from discovering two lines as variants of each other. Now the best practice would be use benchmarking, i.e., assessing the algorithm's performance against a text that has already been correctly aligned. A corpus of benchmark collated text is a recognised need for further testing and improvement of algorithms (Dekker et al. 2015).

2.6.4 *Correction, analysis and visualisation of the results*

The possibility to correct, analyse or process the output in any suitable way is highly dependant on the output format. What kind(s) of output is available from the software? How readable is it? How much correction does it need? What are the reuse options, that is, what can be done next with the output: visualisations, establishment of a stemma, displaying a critical apparatus for a print edition? Is the output compatible with other tools?

The first programs' outputs were not saved for further reuse on a computer, but printed on paper to imitate a printed critical apparatus, or a manual collation. The need for a practical and readable format was therefore so important that the desired output influenced the comparison process (Dearing 1970, 257; Widmann 1971b, 58). Gilbert (1973) and Hockey (1980) show a few examples from those early programs and discuss their respective advantages and problems. Some outputs were particularly difficult to read: for instance Froger's approximation of a critical apparatus needs to be interpreted according to conventions (Froger 1966, 169). A system of reference numbers, allocated to each token and each gap inserted during collation, makes it difficult to find back the reading in the base text (Gilbert 1973, 140). The least practical output was the one from Cabaniss, because the variants were not gathered together. Each pairwise comparison would produce one output: for a tradition of twenty-five witnesses, this represents twenty-four pages printed

2.6. Comparing collation tools

```

2179 And this dittie after mee, Sing; and daunce it trippingly.
q2          Ditty    me, Sing  dance
f1          Ditty    me, sing  dance   trippinglie.
f2          Ditty    me, sing  dance   trippinglye.
f3          Ditty    me, sing  dance
f4          Ditty    me, sing  dance   trippingly.>w
r1          Ditty    me, Sing  Dance
r2          Ditty    me, Sing  Dance
r3          Ditty    me>w
p1          ditty    me>w
p2          ditty    me>w
t1          ditty    me>w
t2          ditty    me>w
h1          ditty    me>w
wa          Sing, and dance it trippingly.>w
j1          ditty    me>w
ca          ditty,   me,>w
h3          ditty,   me,>w
ma          ditty,   me,>w

```

Figure 2.19: Example of early collation output (Widmann 1971b, 61).

for each line of text (Gilbert 1973, 143). In practice, this means that the variants must be gathered by hand, and manual collation may thus be preferable given the amount of data preparation for an automated procedure.

On the other hand, Widmann's program output was considered the most useful and elegant, even though it was meant for poetry (Gilbert 1973, 142). Widmann's output would print the entire line of the base text, and the variant readings would be printed underneath, similar to a collation table or a manual collation (figure 2.19). The collation table visualisation will be discussed in detail in Chapter 8. The output from OCCULT was also a good solution for prose texts (Hockey 1980, 151). The first program to store variants on the computer for further manipulation was Gilbert's program (Hockey 1980, 154). The variants could be reused for instance to correct the variant readings (Gilbert 1973) or to print a critical apparatus along the text (Gilbert 1979). The critical apparatus display, or at least an approximation of a critical apparatus, has remained popular for a long time. It is available in many programs such as TUSTEP, Collate, PC-CASE, UNITE, Juxta (as a prototype visualisation), and the Classical Text Editor. CollateX also provides an XML output in TEI which encodes readings and their variant forms in <app> elements that can be later displayed as a critical apparatus.

However the critical apparatus format may not be the most practical, to analyse the witnesses relationship and create a stemma. Robinson (1989b) created a database of variant readings to compare (see also Section 8.3.2.1). In Collate, the output can be reused with phylogenetic software to create a stemma: a single collation serves therefore several purposes (Robinson 2004). Other collation tools let users create a

2.6. Comparing collation tools

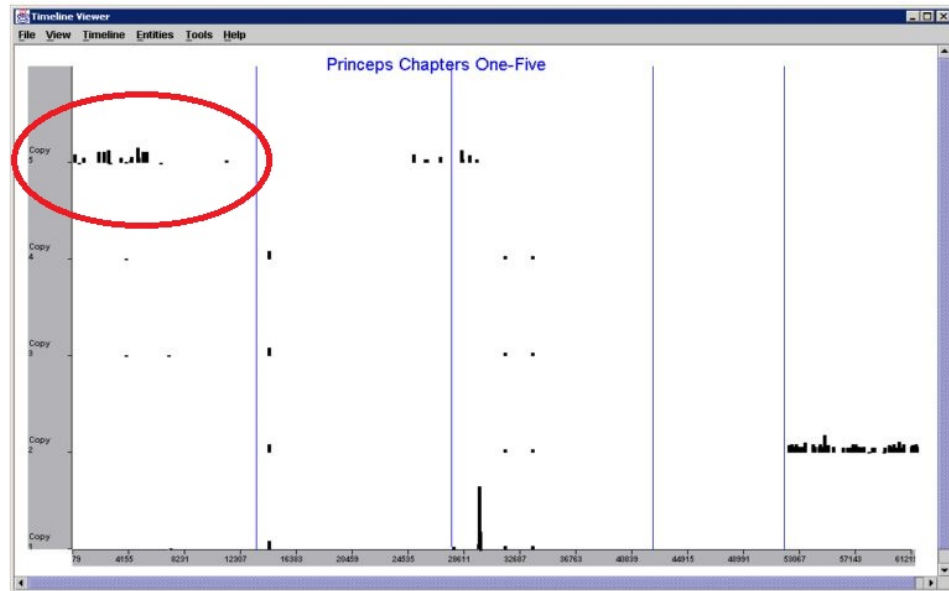


Figure 2.20: A collation visualisation in histogram (Monroy et al. 2002).

stemma by reusing the collation results as input to another program: for instance CollateX's results may be reused in Stemmaweb, and Philipp Roelli's program is used in conjunction with the phylogenetic software Phylip. Ecdosis, the interface which implements the collation tool Compare, offers a 'tree view' as well created with Phylip.

The first innovation, in terms of a visualisation in human readable form, seems to come from the Cervantes Project (Urbina et al. 2002; Monroy et al. 2002). Instead of displaying textual content, the visualisation shows the location of variants in a text compared to a base text, and the length of those variants represented by vertical bars. For instance in figure 2.20, a group of short vertical bars in the top left corner represent a series of short variants in chapter 1 (first vertical column) of the edition number 5 (first horizontal line). This kind of visualisation in histogram was adopted by Juxta as well as by the tool CATview. This visualisation makes it possible to get an overview of the places in the text which have variants, and to see if there are smaller or larger variants.

Another innovative visualisation, the Heat Map, was adopted by Juxta. The principle is similar to the histogram, except that it includes the textual content as well. A witness is chosen as a base text, and the readings of this base text are highlighted in blue when a variant reading is available in one or several other witnesses. If the reading is highlighted in light blue, it means that the other witnesses are very

2.6. Comparing collation tools

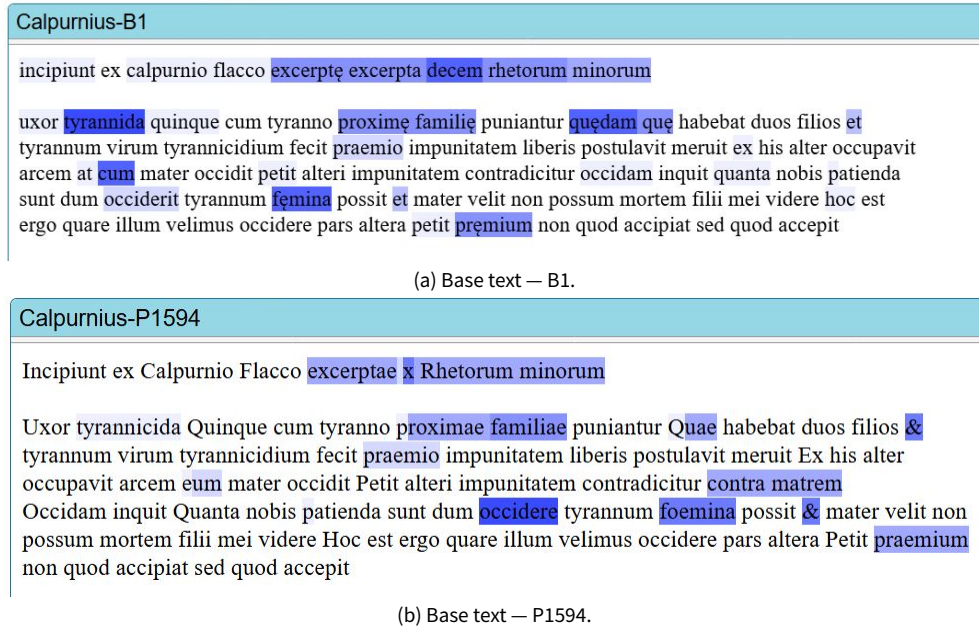


Figure 2.21: Different highlightings in Juxta's Heat Map, from the same text sample. Created in Juxta Commons (June 10, 2017).

similar to the base text at this point. On the other hand, if the text is highlighted in dark blue, it means that there are many witnesses which disagree with the base text at this point. However, a dark highlight does not mean that there are many different variant readings: if all the witnesses are different from the base text but they all share the same reading, this will be displayed as a dark highlight, even if there are only two competing variants. This may become even more confusing when the base text is changed, because that variant will then be highlighted in light blue instead of dark blue. Figure 2.21 shows two extracts from the same collation sample of Calpurnius Flaccus, but with two different base texts and two different patterns of highlighted readings. In the first extract of figure 2.21 (a), where witness B1 is the base text, the word *tyrannida* (line 2) is in very dark blue, and *occiderit* (line 5) is in light blue. On the other hand, the corresponding readings *tyrannicida* and *occidere* have a different highlighting intensity in the second extract, where the base text is witness P1594 (figure 2.21 (b)).

A popular visualisation is to show texts in parallel columns, side by side. In Juxta it is possible to view either two texts side by side, or several texts thanks to the integration of the Versioning Machine visualisation. The side by side view is also available from Ecdosis, and in TUSTEP. The advantage of Ecdosis is that the parallel versions are always synchronised when a user scrolls down one version. This is

2.6. Comparing collation tools

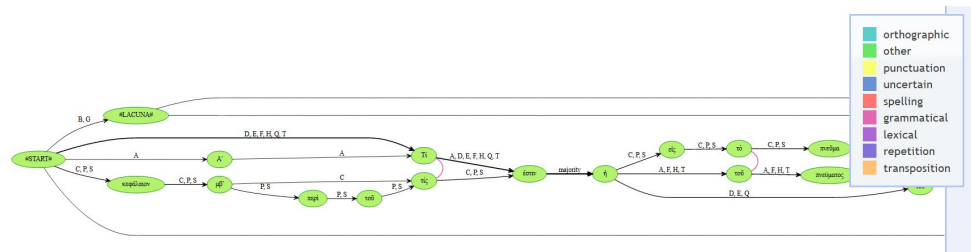


Figure 2.22: Stemmaweb example of a graph. Retrieved from <https://stemmaweb.net/stemmaweb/relation/E7D1901A-AB49-11E1-9EA5-EEF06FF5D3E7> (July 5, 2017).

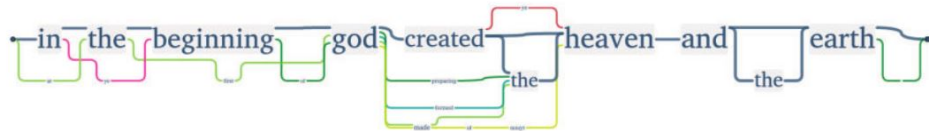


Figure 2.23: TRAViz example of a graph (Jänicke et al. 2015, 90).

not the case in TUSTEP, and it may be slightly difficult in Juxta when one version is significantly shorter than the other.

Collation may also be visualised as a variant graph: although this is not a common way to visualise texts, it is still the best visualisation for transpositions. We have seen above examples of CollateX variant graph (figure 2.11 p. 76, or figure 2.16 p. 86). CollateX's output may as well be visualised as a graph within Stemmaweb (figure 2.22). TRAViz proposes four new features which are meant to improve the graph visualisation: the removal of the circle shapes around tokens, the addition of colours to differentiate witnesses instead of labels on the edges of the graph, the linear arrangement of the text, and the resizing of tokens to reflect how many witnesses actually have these tokens present (Jänicke et al. 2015). The graph visualisation is further discussed in Section 8.3. If many witnesses have a reading, this reading will appear bigger, and vice-versa (see figure 2.23). The collation table visualisations, available from CollateX, nmerge and iAligner, will be discussed in Section 8.2.

As we have seen above, a fixed visualisation can be problematic. A few interfaces offer to correct the alignment. This is the case for TRAViz, stemmaweb, and the collation editor (as well as a future visualisation for the Digital Mishnah project). The common aspect of these three visualisations is the use of a 'drag-and-drop' feature, which let users select a token and drag it to its correct place in the alignment. Special care must be taken to prevent users from altering the order of tokens in

a witness. In addition to corrections, Stemmaweb also provide the possibility to annotate two readings with information about how the readings are related (if the difference is grammatical, lexical, and so on), and about the significance of the variant with respect to the stemma (would a scribe correct the first reading to the second? Is it a likely polygenetic error? And so on). The collation editor offers also a drag-and-drop feature to regularise one reading to another (Houghton and Smith 2016). TRAViz lets users modify the relative edit distance, so as to decide if all differences should be displayed in the graph, or if orthographic differences should be ignored from the visualisation (Jänicke et al. 2015).

In this dissertation I have preferred the collation table visualisation, and I have worked on how a simple table format can be enhanced with additional information (see Section 8.2.2). However, the table format is not well suited to represent transpositions: in this case, a graph format such as the one in Stemmaweb or TRAViz may be better suited.

2.7 Conclusion

Comparing collation tools is not as straightforward as it seems: it can be difficult to judge the efficiency of different algorithms, and there are many aspects to take into account besides the accuracy of the results, such as the level of technical knowledge required, the use or not of a base text, the ability to adapt the algorithm, or the available outputs and visualisations. There is no universal solution, but a diversity of options: although I have preferred CollateX for this thesis, in other circumstances another tool may be more suited. For instance Huculak and Richardson (2013) chose Juxta for its user-friendly interface, while other scholars felt the need to create new collation tools to fit their needs.

The reason why new tools were created can provide insights to the issues of automated collation tools, and the criteria defined above can serve as a useful framework to analyse those issues. For the SaDa projects, two new programmes LERA and LAKomp were created in response to issues of interface and of the collation process: TUSTEP was not user-friendly enough, and required considerable effort to master. On the other hand Juxta was a base text comparison, while CollateX's algorithm was difficult to adapt and faced trouble with larger texts (Pöckelmann, private correspondence). In the case of iAligner, the issues were related to interface again, and to visualisation: while the algorithm of CollateX was satisfying in theory, a more user-friendly interface was needed, as well as better collation tables for visualisation purposes (Palladino, private correspondence).

The list of all collation tools that were discussed in this chapter can be found

2.7. Conclusion

in Appendix A.1, in chronological order. In the next chapter, I will discuss the theoretical problems related to the data preparation, that is the transcription of the witnesses in digital format. I will argue that transcription is not so much of a change compared to manual collation, but there are specific issues that will arise in the context of transcription for the purpose of automated collation.

Transcription: a Prerequisite for Automated Collation

3

As we have seen in Chapter 2, transcription of witnesses in digital format is a prerequisite for automated collation. This chapter deals with transcription, more precisely from the point of view of automated collation. There are different reasons for transcribing manuscripts and printed editions, and collation is only one amongst many. In a first section, I will review the theory of transcription, in particular the three aspects which are covered by this term: the act of transcription, the transcription document resulting from the act, and the relationship of a document being a transcription from another one. The next section studies how transcription and collation are related, and how the two activities may be formalised under a common model. Finally, I will review the issues related to transcription for the purpose of automated collation, and how different scholars have dealt with those issues.

3.1 Theory of Transcription

3.1.1 *Defining Transcription*

Transcription is a ‘written or printed version of something’ (Oxford Dictionary). Transcription involves the translation from one system of signs to another system (Robinson and Solopova 1993). Transcription takes different forms in various disciplines: in genetics, transcription is the copy of a DNA sequence into RNA so that information stored into an organism’s DNA can be expressed in an observable trait; in music, transcription is the adaptation of a score for a specific instrument or ensemble (for instance the adaptation of Beethoven’s symphonies for piano solo by Liszt); in linguistics transcription is the representation of language into written form, either from a speech or signing, or from a text into another writing system. In scholarly editing, transcription refers to the latter, the translation of one text and its writing system into a new text, a new document. As collation may refer both to the act of comparing witnesses, and to the result of this comparison (Chapter 1),

transcription as well refers both to the act of translating a text from one source document (or exemplar) to another and to the new document obtained as a result of this act of translation. In addition, transcription may denote the relationship between two documents: one document is a transcription of another (Huitfeldt, Marcoux, and Sperberg-McQueen 2008).

3.1.1.1 Transcription - Act

The act, or process, of transcription would be very difficult to formalise (Huitfeldt, Marcoux, and Sperberg-McQueen 2008). It follows loosely defined rules which are not always applicable, and draws from external sources of information such as the context, or the transcriber's knowledge. Nevertheless, the act of transcription is generally divided into two different stages: the first stage is decoding, or deciphering, the source document; the second stage is the encoding, or transposition, of the source text into a new document (Robinson and Solopova 1993; de Biasi 2004). In the context of automated collation, more precisely, the new document must be in electronic format and machine readable form, i.e. the sequence of characters, spaces and punctuation which constitutes the text must be readable by a computer and therefore it cannot be an image of the text (pdf or other image format such as jpg).

3.1.1.2 Transcription - Document

The result of this act of transcription is a new document, which is also called 'transcription'. A transcription is never a perfect representation of the original source document, but rather a selection of a set of features among an infinity of features: it is impossible to record everything in a transcription (Lavagnino 2006). In that sense a transcription is a simplification of the source document and therefore a model of the source document (McCarty 2004). Transcription documents are often described in reference to the number of features they retain from the source document: the different levels of transcription form a spectrum ranging from very faithful diplomatic transcriptions, which retain as many features from the original as possible, to modernised transcriptions which select a limited number of features to record, the features considered most significant (Driscoll 2006). The selection of features depends on the purpose of the transcription, and the need to balance the tension between an accurate representation of the source document and the readability of the transcription for untrained readers (Burnard 2014a).

Different levels have been identified in the spectrum of transcriptions, depending on how 'accidental features' are treated, such as letter shapes, abbreviations,

capitalisation, punctuation, structure or layout of the text on the page (including page, line, and even word division), scribal errors or misspelling (Driscoll 2006). The different levels are namely:

- **Ultra-diplomatic:** the main difference with facsimile is the use of typographic characters (D'Iorio 2010, 52). The ultra-diplomatic transcription aims at reproducing the layout of the document, and not only the text. This level seems to correspond to the 'graphic' level of Robinson and Solopova (1993).
- **Diplomatic:** a transcription which retains as many features as 'may reasonably be reproduced in print' (Driscoll 2006). This includes all the accidentals listed above. The diplomatic level of Driscoll (2006) is equivalent to the 'graphetic' level of Robinson and Solopova (1993).
- **Semi-diplomatic**, semi-normalised (Driscoll 2006), or 'graphemic' (Robinson and Solopova 1993): in this intermediary level, some but not all of the accidental features are kept in the transcription. For Robinson and Solopova (1993), the graphemic level retain the original spelling, but not the original letter forms of the source document.
- **Modernised** (Driscoll 2006) or regularised (Robinson and Solopova 1993): every accidental feature is normalised for modern readers (Driscoll 2006).

In practice, however, transcriptions do not always clearly belong to one level, but may borrow aspects from another level (Robinson and Solopova 1993). The content included in a transcription will depend on its purpose. Pierazzo (2011) has discussed a set of criteria and parameters to help editors with this decision. The criteria to take into account include the needs of editors as well as the needs of the intended audience, the nature of the documents, the publishing technology and the costs associated with this work (Pierazzo 2011, 468).

When transcribing for the purpose of automated collation, editors must decide which level of transcription is aimed for. The modernised level of transcription is not recommended in practice (Robinson and Solopova 1993, 23; Macé et al. 2015, 331). Although normalisation is needed in order to prioritise substantial variants in the visualisation, Robinson and Solopova (1993) have found it advantageous to postpone the introduction of normalised forms: it would prevent ignoring an accidental variant which might be in fact significant, and it increases the speed of transcription since there is no need to choose a normalised form for each word.

3.1. Theory of Transcription

As we have seen in Section 1.4.2, there is no clear consensus about the inclusion of accidentals in collation, such as orthographic differences. Some scholars, such as Willis (1972), have judged the accidentals to be irrelevant for establishing the relationships between witnesses: ‘Accidental variants, i.e. those which are purely matters of spelling or punctuation, are so unpredictable in the pre-1800 period as to have little value as evidence for the descent of texts. Editors, as a rule, do not even bother to record them’ (Love 1984, 52). Robinson (1991, 85) dismisses accidentals in medieval traditions as non genetic, since they are the product of differing scribal habits. Others have argued that accidentals are not relevant especially in the case of Classical texts (Stussi 1994; Bourgain and Vielliard 2002). Reeve (2011, 364), for instance, agrees with Viré’s decision to ignore spelling differences while analysing Hyginus’ tradition.

Accidental features, such as scribal errors, can nevertheless be of importance to analyse the manuscript tradition. Abbreviations as well can be a source of error, and may therefore be useful in order to understand the relationship between witnesses (West 1973, 27; Macé et al. 2015, 331). A different punctuation may change the meaning of a sentence. Regarding letter forms, Robinson (1989a) created a special font to imitate the shapes of letters in Norse manuscripts, but this was for the purpose of easing the correction of transcription errors against the source document. And accidentals may also be useful to clarify the relationships between closely related witnesses (West 1973, 66; Robinson and Solopova 1993, 24).

In more recent studies in stemmatology, scholars have attempted to assess the actual importance of accidentals, using digital tools to challenge the assumption that accidentals are not relevant (Spencer et al. 2004; Schmid 2004). For instance, Andrews (2014a) has concluded that variation deemed insignificant ‘was surprisingly likely to follow text-genealogical transmission patterns in both artificial text traditions and genuine traditions’. Blake and Thaisen (2004) have shown how the study of spelling differences can help editors to detect if a scribe used more than one exemplar while copying Chaucer’s *Canterbury Tales*, which is relevant to the text’s stemma. Cardelle de Hartmann, Senekovic, and Ziegler (2014) have remarked that accidentals in the tradition of Petrus Alfonsi would confirm the groupings obtained with substantive variants. Lapin (2013) has also experienced with the creation of stemmata for the Mishnah, both with and without orthographic differences, for which phylogenetic tools did produce different results. However, Lapin (2013, 454) concluded that orthographic variation was noise that needed to be filtered out.

3.1. Theory of Transcription

In conclusion, accidentals may not be as relevant to the creation of the stemma as substantive variants, due to the ‘noise’ from scribal habits. However, accidentals can be used at least for confirming groupings from more substantive variants, or clarify relationships of closely related witnesses, since they are likely to follow the same genealogical patterns of a textual tradition.

3.1.1.3 Transcription - Relationship

Finally, transcription can describe the relationship between two documents, one of which is a transcription of the other. Huitfeldt, Marcoux, and Sperberg-McQueen (2008) have provided a detailed model which attempts to formalise this relationship, and answer the question ‘is this document a transcription of that other document?’ Huitfeldt, Marcoux, and Sperberg-McQueen (2008) do not attempt to model the act of transcription, nor the result of transcription, but only the relationship between two documents.

The transcription model was created in the context of scholarly editing and in the creation of digital resources. The premise is the following:

In general, one document (the transcription, T) is said to be a transcription of another document (the exemplar, E) if T was copied out from E with the intent, successfully achieved, of providing a faithful representation of a text as witnessed in E (Huitfeldt, Marcoux, and Sperberg-McQueen 2008, 296).

Thus the text of the transcription must be similar to the text of the exemplar, since the transcription has successfully represented the exemplar’s text. If two documents have a similar enough text, according to the model, one is a transcription of the other. Therefore the model must provide a framework to compare texts and judge their similarity.

Here a few definitions of key terms are necessary to understand the transcription model: document, mark, token, type and reading. A document is a physical object which contain marks, that is ‘perceptible features’ (such as lines drawn in ink, or dots embossed in paper for the braille writing system) which can be interpreted as text. Although ‘text’ is a debated concept (see for instance Caton 2013), in the transcription model a text is a sequence of graphemes such as letters, spaces, punctuation marks, and other symbols.

3.1. Theory of Transcription

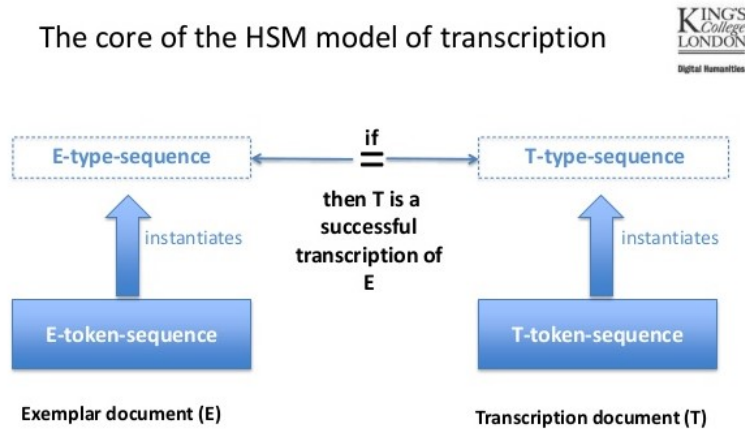


Figure 3.1: Summary of the transcription model (Caton 2014). Retrieved from <https://fr.slideshare.net/PaulCaton/dh2014-slides> (July 21, 2017).

A document is an object that contains at least one mark interpreted as text. Marks are identified as tokens if they can be mapped to a type¹. While a type is an abstraction of a letter, a word, and so on, the token is a concrete physical representation of a type, and therefore unique. On the other hand, marks which cannot be mapped to a type are not tokens. Finally, it is worth noting that the ‘distinction among marks, tokens and types may be applied at various levels: letters, words, sentences, and texts’ (Huitfeldt, Marcoux, and Sperberg-McQueen 2008, 297). Tokens and types are mapped by ‘readings’, which is the interpretation of a human reader. For instance the token *p̄n̄e* can be interpreted as the type *poenae* or *paene*, depending on the reader.

Figure 3.1 shows a representation of the model, which describes the relationship between two documents E (exemplar) and T (transcription). The model seeks to answer the question ‘when is a document T considered a transcription of document E’? A document T is considered a valid transcription of an exemplar E if they are similar enough, i.e., if they have the same sequence of types².

The model is formalised with the alloy language created by Jackson (2006). The alloy files of the model are available from the Markup Language for Complex Documents web server³. Some issues remain to be addressed by the model, such as

¹The distinction between tokens and types was introduced by Peirce (1906, 505-506), as means of distinguishing the abstract word and its written representation.

²In practice this means that a document is similar to itself, although a document can hardly be considered a valid transcription of itself.

³Markup Language for Complex Document (MLCD) is a project led by Claus Huitfeldt. The

3.2. Transcription and Collation: a Common Model?

the formalisation of abbreviations, of material omitted from a transcription such as drawings, or material added to the transcription such as line numbers for instance (Huitfeldt, Marcoux, and Sperberg-McQueen 2008). Despite these limitations, the model is still very useful as a way to formalise the relationship of a transcription to its exemplar .

3.2 Transcription and Collation: a Common Model?

In the previous chapter, we have examined the principal differences of methodology between manual and automated collation (Section 2.4). In particular, the fact that witnesses must be transcribed as a prerequisite to automated collation was highlighted as a major change of method, which was often advanced as an argument against the use of automated collation. However, I will argue in this section that manual collation and transcription are in fact very closely related activities. In order to make this point, I will explore the possibility to adapt the transcription model in order to represent the activity of manual collation as well. The argument for adapting the model is that manual collation is a form of transcription, however selective.

In Chapter 1, we have examined the methodology of manual collation and identified several steps, which include (1) the selection of a base text, (2) the comparison of other witnesses to this base text, and (3) recording the differences in parallel to the comparison. While comparing a witness (the comparison text) to the base text word by word, the collator stops each time there is a significant difference between the witness and the base text, and record this difference by transcribing it into another document.

To summarise, if the base text and the comparison text collated do not share the same reading, then the fact is recorded by transcribing the reading in the collation. On the other hand, if the base text and the manuscript share the same reading, then nothing is actually transcribed; the absence of transcription implies that at this point, the witness collated is considered similar to the base text. In other words, the base text is considered a valid transcription of the comparison text.

project ‘aims to provide a notation, a data structure and a constraint language which as far as possible is compatible with and retains the strengths of XML-based markup, yet solves the problems with representation and processing of complex structures’. See <http://mlcd.blackmesatech.com/mlcd/About.html> (Accessed July 19, 2017). The first file contains the formalisation of tokens, types, and documents <http://mlcd.blackmesatech.com/mlcd/2008/Papers/gs.als> (Accessed July 19, 2017). The second file contains the additional formalisation of readings which map tokens to types

3.2. Transcription and Collation: a Common Model?

It is therefore important to underline the difference between an actual transcription, when a variant reading is recorded, or a ‘virtual’ (silent) transcription, when nothing is recorded because the two texts are considered similar. Many details can be ‘lost in transcription’ during the collation. Features which are not usually collated, such as orthographic differences, are ignored in the comparison text, but can be recorded at the points of textual variation when the exemplar is actually transcribed.

For instance in Håkanson’s edition of Calpurnius Flaccus, in declamation fifty-two, the variant reading *conditio* is recorded in the apparatus for the word *condicione* in the critical text (Håkanson 1978, 39.20). If all witnesses had had either *condicione* or *conditione*, this reading would not have been recorded and therefore the orthographic difference of *c/t* would have been unnoticeable unless a reader decided to check the manuscripts.

On the other hand, not all orthographic differences may thus be recorded in collation. Abbreviations or special characters may not be reported in a critical apparatus. Again, examples can easily be found in Håkanson’s edition: in declamation four, the reading *Aestimate iudices* adopted by Håkanson is a conjecture of Pithoeus, while the manuscripts read *aestima aliud* (Håkanson 1978, 5.1). In fact, Pithoeus’ critical text reads *Aestimate iud.*, which is an interesting interpretation of how the text may have been corrupted in the manuscripts, but this is not reported in Håkanson’s apparatus⁴.

The last step in manual collation, recording the differences, involves the transcription of the variant readings from the comparison text into a document. That document may be the same document that contains the base text, such as the critical edition which serves as a base text (Stählin 1914), or a Microsoft document such as an Excel spreadsheet (Macé et al. 2015). The variants may also be recorded in a different document (for instance Lendle 1968, Willis 1972, De Strycker 1975, Whitaker 1991). See Section 1.4.1 for a description of the notation methods adopted by various scholars when collating.

Since collation is actually in fact a partial transcription of several witnesses, could it be modelled as such, with the support of the transcription model? The next sections will first consider preliminary remarks, and then propose two examples of a first model: far from perfect, this first model rather provides a starting point to bridge the difference between manual and automated collation.

⁴See Pithoeus’ text here: http://www.e-rara.ch/gep_g/content/pageview/1098917 (Accessed July 24, 2017).

3.2. Transcription and Collation: a Common Model?

3.2.1 Preliminary remarks

The first question to address is the purpose of the collation model. In the transcription model, the purpose was to model one aspect of transcription, that is the relationship between two documents but not transcription as an act or a document (see Section 3.1 above). The purpose of the transcription model is to answer this question: given two documents E and T, can it be said that one is a transcription of the other? Similarly, the purpose of a collation model will be to describe the relationships between several documents.

As for transcription, the word ‘collation’ may refer as well to various aspects: collation can be either the act of comparing witnesses and recording the differences, or the result of this act, i.e. the document where the differences are recorded. Some parallels can be drawn between the act of transcription and the act of collation. Transcribing is divided into two stages, decoding an exemplar and re-encoding the exemplar’s textual content in a new document called ‘transcription’ (Robinson and Solopova 1993, 21). Collating may similarly be divided into several stages: decoding a base text, decoding a comparison text, making a decision about the similarity of those two documents. If the decision was negative and the two documents are considered different, a last stage is to transcribe the difference: re-encoding the comparison text in a new document called ‘collation’.

The act of collating is therefore different from transcribing, but it also shares stages in common, the decoding and re-encoding acts are still performed. Since this collation model is based on the transcription model of Huitfeldt, Marcoux, and Sperberg-McQueen (2008), it will not attempt to model the act of collation, but rather the relationships between several documents: given three documents D1, D2, D3, is it possible to determine that one document D1 is a collation of a comparison text D2 against a base text D3?

The model should also take into account the fact that two editors may not produce the same collation from the same witnesses, just as two scholars may not produce the same transcription from the same document. For that purpose, the objects of tokens, types and readings are necessary. Readings are the interpretation of a collator that a token is an instance of a type. For example, two words that differ only in orthography can instantiate the same type for an editor who will not record this difference in the collation, or different types for another editor who will therefore record the difference in the collation.

Here it is also essential to make explicit some assumptions of the model. The

3.2. Transcription and Collation: a Common Model?

transcription model assumes that a transcription is a perfect representation of the original, it does not account for mistakes, scribal corrections, or any sort of modifications that alter the transcription. In the collation model, it is also assumed that the collator does not make any mistakes while collating, and does not miss any difference (the collator records all significant variation). In the transcription model, the comparison between two documents does not require that it be made between two distinct documents, i.e. a document can be considered a valid transcription of itself. What are the implications for a collation model? Should it be possible to consider that a document is a valid collation of itself against itself? It seems contradictory at first glance: if a collation records differences between two documents, and a document is similar to itself, therefore a collation should be empty. On the other hand, we have postulated that when the collator does not record anything, it is because the base text is a valid transcription of the comparison text, and therefore a base text is valid collation of itself. It would seem best as well to not impose additional constraints, and to keep the model as general as possible. However, it is simpler for a first example to treat collation as a separate document from the base text and other witnesses collated, in order to have a clearer understanding of how the various documents relate to each other. Then a second example attempts to remove this condition.

3.2.2 *Collation model*

For the collation model, I have kept the transcription model described in Huitfeldt, Marcoux, and Sperberg-McQueen (2008), extended with new functions for collation. The class of objects from the transcription model are also relevant: documents, tokens, types and readings. Documents are the objects which we are comparing. Base text, comparison text, and collation results are all instances of the class 'Document'.

With respect to the transcription model of Huitfeldt, Marcoux, and Sperberg-McQueen (2008), the collation model innovates in three ways: the first is a correction to the transcription model which was provided by M. Sperberg-McQueen. The second innovation is related to the meaning of reading, and the last one is the addition of two predicates.

3.2.2.1 A Correction to the Transcription Model

For consistency in the transcription model, 'a token is associated with a document if and only if it appears in the sequence of tokens identified by some reading of that document (Huitfeldt, Marcoux, and Sperberg-McQueen 2008, 304). This constraint on tokens is expressed by the following code scrap (number 17, page 305):

3.2. Transcription and Collation: a Common Model?

All $r : \text{Reading} \mid$
this in elems [r.tokenseq]
iff r.doc = d

However, the constraint was too strong, and allowed for tokens to exist while not being mapped to a type, which should not happen. To correct the issue, Sperberg-McQueen has proposed to add the following constraints on tokens elements⁵:

some $r : \text{Reading} \mid$ this in elems[r.tokenseq]
all $r : \text{Reading} \mid$ this in elems[r.tokenseq] implies r.doc = d

The first line states that a given token must be part of a reading's sequence of tokens. The second line states that a token is in a document D only if every reading including the token in its token sequence (tokenseq) is also a reading of the same document (r.doc = d). Finally, a last constraint states that in the reading-to-type mapping of tokens (rtt), a reading can map to a type only if this token maps to the same type in the reading's mapping of token to type (r.tt):

all $r : \text{Reading} \mid$ all $t : \text{Type} \mid$ r->t in rtt iff this->t in r.tt

Otherwise, the collation model takes the same four signatures as were present in the transcription model: Document, Token, Type and Reading. The transcription predicate *t_similar* is still valid, that is two documents are similar if their sequence of Tokens generates the same sequence of Types for two different Readings⁶.

3.2.2.2 The Meaning of 'Reading'

The scope of the term 'reading' must be clarified, as it can be understood in two different ways. In the transcription model, a 'reading' is a reading of the entire document, from start to finish. On the other hand, in the context of textual criticism a reading is commonly understood as 'the form or content of the text in a given place' (Froger 1968, 9). In the common sense of the term reading, a document contains a sequence of 'individual readings'⁷, while in the transcription model there should be only one reading per document. These two views may seem incompatible, but in fact both can be nuanced.

⁵Sperberg-McQueen sent me the correction in a private email (2014), after I pointed out the error.

⁶In the context of logic and formalisation, a predicate is 'a statement proclaiming that certain variables satisfy a property' (Cunningham 2012, 29), so that the statement is either true or false. The predicate *t_similar* takes two documents as argument, and will be evaluated as 'true' if one document can be described as a transcription of the other.

⁷'Individual readings' as opposed to readings of a whole document.

3.2. Transcription and Collation: a Common Model?

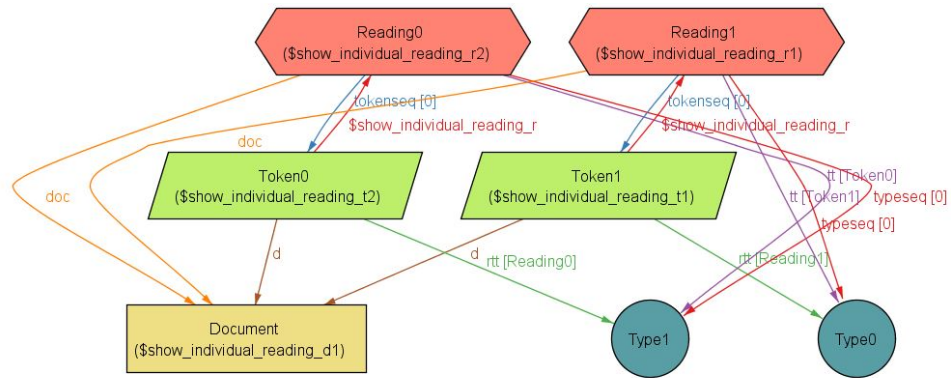


Figure 3.2: An instance of the transcription model with possible individual readings.

Parker (2008, 4) argues that variant readings should be defined as ‘the entire text as it is present in a particular copy’. According to Parker (2008), the reason why variant readings are usually defined as a part in the text (and not the whole text) is for simplification purposes:

Because two copies of a text will have wording in common between them, in practice a variant reading describes the places where the common text ceases, and each has its own form. ‘Variant reading’ is in fact a simple tool for breaking down the differences between two or more copies into manageable units Parker (2008, 4).

It could be argued that readings, and not only variant readings, are broken down into manageable units for practical reasons. The transcription model is based on the concept of similarity of readings, in order to decide if a document T is a transcription of an exemplar E. However, in a collation model, we would need to distinguish between variant and non-variant parts of the text in order to decide if a document is a collation of a witness against a base text. For this we need readings to denote parts of the text, and not the entire text.

In addition, the transcription model may also be flexible. Huitfeldt, Marcoux, and Sperberg-McQueen (2008) intended reading objects as the words of a given witness, from beginning to end. However the Alloy model does not seem to make it explicit that the token sequence of a reading corresponds to the entire token sequence of a document. As a result, reading objects could be used to denote ‘individual reading’ as well as the reading of the whole document.

3.2. Transcription and Collation: a Common Model?

In figure 3.2, a valid instance of the model shows a document with two tokens (Token0 and Token1), and two different readings (Reading0 and Reading1) mapping each token to a different type: Reading0 maps Token0 to Type1; and Reading1 maps Token1 to Type0. This instance can be interpreted in two different ways. The first is to consider that the readings represent the whole document. In that case, Reading0 does not interpret the marks of Token1 as significant, and Reading1 ignores Token0. The second interpretation is that Reading0 and Reading1 are two individual readings in the document. Reading0 is a reading of the first token (for instance the first word), and Reading1 is a reading of the second token.

Because of this flexibility of interpretation, the transcription model seems to be suitable to expand for modelling collation. Thus the collation model is based on the assumption that ‘reading’ means ‘individual reading’. As argued above, the distinction between variant readings and non-variant readings is necessary for the collation model.

3.2.2.3 New Predicates

To model collation, a predicate is needed in order to state that two Readings are similar, instead of whole documents. Here is a predicate *r_similar* (‘reading similarity’), that is evaluated as true if two readings’ sequence of Token maps to the same sequence of Type:

```
pred r_similar (r1, r2: Reading) {  
  r1.typeseq = r2.typeseq  
}
```

In the collation model, a collation C, a base text B, and an exemplar E are three separate documents. There are two possible situations. Firstly, the base text B and the exemplar E have a similar reading: the predicate *r_similar* is evaluated as true. In that case there is no variant to record, and the collation C is an empty document. The base text B is a valid transcription of the exemplar E: both B and E are *t_similar* (see figure 3.3).

On the other hand, the base text B and the exemplar E may have different readings, that is readings which do not generate the same sequence of type (predicate *r_similar* is evaluated as false). In that second case, a variant is recorded in the collation C, which has the same reading as the exemplar E. It implies that the collation C is a valid transcription of the exemplar E: both documents C and E are *t_similar* (see figure 3.4).

3.2. Transcription and Collation: a Common Model?

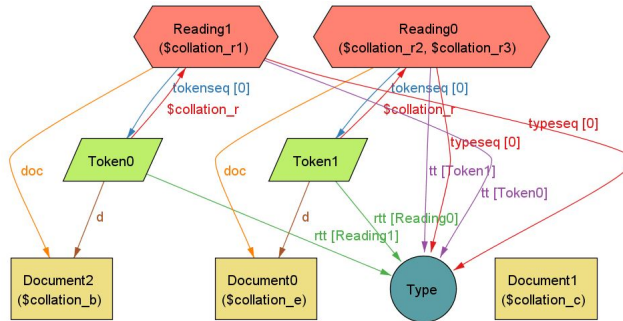


Figure 3.3: The exemplar and the base text have a similar reading.

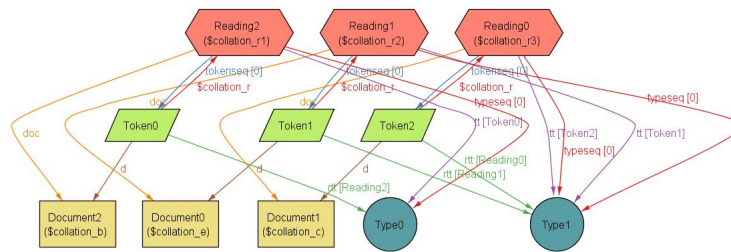


Figure 3.4: The base text and the exemplar have a variant reading which is recorded in the collation.

The predicate for collation is the following:

```

1 | {pred collation (disj b, e, c: Document) {
2 |     some r1, r2, r3: Reading | {
3 |         //reading 1 is in the base text
4 |         r1.doc=b
5 |         //reading 2 is in the exemplar
6 |         r2.doc=e
7 |         not r\_similar[r1, r2] implies
8 |             {r3.doc=c and r\_similar[r2, r3] and t\_similar[c, e]}
9 |         else no r: Reading | r.doc=c and t\_similar[b, e]
10 |     }
11 | }}

```

This simple example illustrates the relationship between the three documents of base text, exemplar and collation, and how a transcription and a collation can be related. However, there are many limitations to this basic model. This is a model for collating only one reading, of one exemplar. What happens when more witnesses are collated, and other variant readings are added to the collation document? How to deal with additions and omissions, instead of only substitutions? Is this an

3.2. Transcription and Collation: a Common Model?

accurate representation of collation? For instance, let us consider an example of a base text B, and an exemplar E:

B: A man petted a small dog

E: A man ate a hot dog

For each pair of corresponding tokens, either B and E are similar and nothing is written in the collation, or there is a difference and the reading of E is transcribed in the collation. The collation, according to the model would look like this:

C: 0 0 ate 0 hot 0

The zeros would serve here as an indication that at this point in C there is nothing, and as an indication of how many readings have been compared. Adding more witnesses to this collation would require some form of alignment, and those zero tokens are very similar to the ‘gaps’ added by collation algorithms in order to provide such an alignment (see Section 2.4.4.3).

But is this a good representation of what scholars actually do? It seems that it is not entirely satisfying. Let us consider a discussion from the 3rd of September 2015 on the Digital Classicist mailing list, in which researchers debates which tools to use to support the collation of manuscripts:

I saw CollateX and also Juxta (and TUSTEP) and they seem to be great where you have transcribed manuscripts. But traditional collation doesn't transcribe entire MSS but only notes the differences. If I can put it this way, those collation packages operate on a ‘diff’ basis, whereas traditional collation works on a ‘patch’ basis. [...] So I'm wondering if there is anything more in the line of supporting traditional textual criticism (Ruffell 2015).

This scholar is aware that the difference between transcription and collation is about the scope: a transcription of an entire manuscript versus the transcription of only variant readings. Nevertheless, they did not feel that transcription and alignment of witnesses were supporting traditional textual criticism. Instead, they were considering using a SQL database to analyse the collation and the manuscripts relationships: “I'm currently thinking SQL data[b]ase of choice and custom queries as needed” (Ruffell 2015).

Indeed a database model of collation may correspond to manually prepared colla-

3.2. Transcription and Collation: a Common Model?

tions examined in Section 1.4.1, such as the card system of Willis (1972) or Whitaker's register of variants. From this point of view, the database of variant is enough, in conjunction with a base text, to generate an acceptable transcription of each witness. Therefore, it maybe it is more appropriate to think of manual collation as a database and to model it as 'set of entries' instead of as a sequence of tokens and 'gaps'. A further development of the model could include:

1. A collation contains a set of entries.
2. Each entry points at one location of the base text.
3. Each entry gives one or more alternate reading(s) for that location and an indication of the witness in which that reading appears.

The manual collations examined in Section 1.4.1.4 mostly follow a similar structure. For instance, the cards of Willis' collation correspond to the collation entries. Each card points at a location of the base text, which is expressed by a number (page and line, or verse number) and the base text reading. Each card also lists the alternative readings, followed by the siglum of the witness where they appear (Willis 1972, 33). There are no cards for the readings of the base text which have no variant in other witnesses. Similarly, the tables in Lendle (1968) function like a set of collation entries: each entry contains an identification number, the base text reading and the variant readings. There is always an indication of the witness in which the readings appear, thanks to the crosses system (see p. 36 above).

3.2.3 Conclusion

Although it is not perfect, the alloy model of collation can help to understand how manual and automated collation are related through transcription. While transcription happens during manual collation, it must be done entirely before automated collation. Transcription of witnesses is only partial for manual collation, while it should be complete for automated collation. As a consequence, if it is acceptable to transcribe only variant readings during manual collation, then it must also be a valid solution to modify a digital base text in order to adapt it for each witness and collate those transcriptions with digital tools, as proposed by Robinson (1989a) and Andrews (2012) (Section 2.4.1, p. 68 above).

However, the discussion above may have revealed a more subtle, conceptual difference between how scholars think about collation (as a kind of database), and the

3.3. Transcription Issues Related to Automated Collation

alignment of witnesses with algorithms. As a result, scholars may avoid automated collation because they do not see how it can support their work of textual criticism. In Chapter 8, I will argue that the results of automated collation can perfectly support scholarly work, with the example of Calpurnius Flaccus.

For the rest of this chapter, I will examine the implications of the change of scope in transcription that takes place when moving from manual collation to automated collation. The next section reviews some issues specifically related to transcription prepared for the purpose of automated collation and how various projects have dealt with those issues.

3.3 Transcription Issues Related to Automated Collation

In the context of automated collation, a scholar preparing transcriptions needs to take into account how the texts will be processed by the collation tool: since a collation tool needs as input the transcription of each witness that must be collated, what makes a witness? How is the tool going to deal with corrections and modifications by different scribes in the same document? How is the text going to be tokenised, and is the tokenisation satisfying? Does the editor need to provide a pre-tokenised text? If so, does the transcription allow for transformation into the desired pre-tokenised format?

The purpose of the collation tool also needs to be taken into account: is the collation result obtained with the tool meant as a visualisation for readers of the edition, or is it a result that the editor will work with in order to produce a critical edition? In that case, what information will be necessary to the editor while working with the collation results? What level of detail should be encoded in the transcriptions? And what will happen to the details encoded in transcription, once the transcription data is transformed to a format accepted by a collation tool? In which format should the transcriptions be prepared? These are examples of specific issues related to transcription in the context of automated collation, which will be discussed in this section with illustrations from projects which have adopted automated collation in their workflow.

3.3.1 *Transcription Format*

In the previous chapter, we have already discussed the various input formats accepted by automated collation tools (see Section 2.6.2). The two most common formats were plain text and XML. Although some tools accept other formats, such as HTML, these are usually transformed into plain text for collation.

3.3. Transcription Issues Related to Automated Collation

Plain text is a very limited format to record a transcription. As it will be discussed in Chapter 8, the collation must include paratextual elements beside the text, such as folio numbers or comments (see also Whittaker 1991). Plain text is not a convenient format to record those elements, since they will appear as variant readings in the collation results, and it will significantly increase the difficulty in visualising real textual variants. In addition, plain text transcription is difficult to adapt for other purposes, such as preparing different input formats for other tools, publication as a diplomatic edition, or displaying the text to different audiences with different interests or qualifications (Zeevaert 2015). On the other hand, an XML input is more flexible: it will allow for the encoding of paratextual elements, but can also make collation easier, by letting scholars choose the exact elements to be collated (see for instance the usage of Juxta, in Section 7.2). However, XML is such a flexible format that it is preferable to follow a standard markup scheme in order to facilitate the interchange of data.

The Text Encoding Initiative (TEI) has become the standard XML format to record transcriptions in the Humanities. It is a very flexible format: the same textual features can be recorded in many different ways (Pierazzo 2016). The TEI provides a rich vocabulary to describe the meaning of the text instead of its appearance (Burnard 2014a). It provides a tool to add meaningful information derived from the appearance of the text (for instance the fact that a segment text is in bold and centred is interpreted as a title). The TEI is not dependant on a specific platform or software environment, and it was designed by the scholarly community with extensive guidelines and instructions, which makes it an ideal encoding format (Burnard 2014a).

3.3.2 *Witnesses*

A witness is usually defined as a written document, either handwritten or printed, which bears a textual copy of a work (Stussi 1994, 89)⁸. This definition does not give any indication regarding the different hands which may have influenced one particular document. It is often the case that a text in a medieval manuscript has not been written by a single scribe. On the other hand, it is a very common situation to find that one scribe is responsible for the major part of the text, and that this same scribe or other scribes made corrections to the text, added variants in the margin, and so on. It is also possible that one scribe started the transcription, which was continued later by a different scribe, possibly using a different exemplar. Scribes cannot always be identified, and the more general term of 'hand' is used to

⁸See more definitions in the Lexicon of Scholarly Editing, s.v. witness: <http://uahost.uantwerpen.be/lse/index.php/lexicon/witness/> (Accessed July 27, 2017).

3.3. Transcription Issues Related to Automated Collation

refer to the various agents involved in the text of one document. Are those different hands representing different witnesses of the text? In modern manuscripts which show the genesis of a text, there may be only one hand (the author's hand), however the question of what constitutes a witness is still valid. Several layers of revisions can be identified, as the author worked on the draft and modified the text. Should these different layers represent different witnesses?

At a conceptual level, it seems that different hands are not considered as different witnesses, although the corrections of a second hand in a manuscript can sometimes be traced back to another witness: for instance in Calpurnius Flaccus, Håkanson identified a relationship between the second hand of the *codex Bernensis* 149 (N) and another manuscript, the *codex Monacensis Latinus* 309 (B) (see Chapter 6 and Section 8.5). Witnesses are defined as documents, and a hand is not a document. At a practical level, however, it is necessary to separate each hand as an individual witness for automated collation, in order to visualise the corrections of different hands as variant readings of the first hand. Therefore it may be more accurate to say that we collate 'texts' rather than witnesses.

As we have seen above, the result of a manual collation can be regarded as a set of entries that record variant readings, and each entry is associated to the corresponding reading of a base text. In that situation, it is relatively easy to include in the entries reading from different hands without having to decide if they represent a new witness. It is also easy to include editorial conjectures or emendations, for instance, even when these are proposed in a scholarly article (see also Section 2.4.1 for the problem of witnesses that do not represent a complete instance of the text).

With automated collation, on the contrary, each instance of the text to compare must be transcribed in full, but this does not prevent the collation of lacunose text. To overcome the issue of editorial conjectures, the Digital Mishnah editor prepared special transcription documents to collect readings from secondary sources which do not contain the whole text of the Mishnah (Lapin 2013, 449). Lacunose witnesses of the New Testament are collated as well⁹.

There are few collation tools that let users visualise properly the different hands or textual layers, if those are not encoded as separate witnesses. In Juxta for instance it is possible to switch between different hands encoded in XML thanks to the 'Edit

⁹The collation Editor has some guidelines on how to prepare lacunose texts for collation: https://github.com/itsee-birmingham/collation_editor (Accessed October 21, 2018).

3.3. Transcription Issues Related to Automated Collation

Document Parsing Template' feature (see also Section 7.2)¹⁰. This feature in Juxta lets users choose the XML tags from which the plain text is extracted for collation, but there are limitations and the result will be different depending on the encoding adopted.

Let us consider a simple example from a short extract of Catullus from manuscript Bodmer 47, folio 1v: the first word of the fourth line has been corrected from *Credo* to *Corde* by a second hand¹¹. In a TEI transcription, the correction could be encoded as such:

```
<subst>
<del hand='#h2'>Credo</del>
<add hand='#h2'>Corde</add>
</subst>
```

The word *Credo* was deleted by the second hand (h2) and substituted with the word *Corde*. In Juxta, the tag selector lets users decide to ignore either the `` tags or the `<add>` tags, so that it is possible to switch between *Credo* and *Corde*, but not to see the two words in parallel as variant readings. In fact, if both `` and `<add>` tags are selected, the two words will appear as two consecutive words in the stream of text (see figure 3.5). In addition, the tag selector cannot be fine tuned to select specific attributes, which makes it impossible to switch between more than two hands or more than two layers of text. On the other hand, if the same extract is encoded with the critical apparatus tags for parallel segmentation in TEI (TEI Consortium eds. 2017c, §12.2.3), Juxta will automatically recognise each hand as a separate witness:

```
<app>
<rdg wit='#Bodmer47-hand1'>Credo</rdg>
<rdg wit='#Bodmer47-hand2'>Corde</rdg>
</app>
```

However, this encoding is actually defining each hands as separate witnesses.

The issue of including many hands as witnesses, while using automated collation, is to visualise the results which may become very large and contain large blocks of identical text (for instance if a later hand has made a very small amount of

¹⁰In the Juxta desktop application, this feature is available in the menu Edit>Edit Document Parsing Template. In Juxta Commons, the same feature is available when an XML witness is selected in the middle column in the first half of the screen, and by selecting 'XML View' under the witness name in the bottom half of the screen.

¹¹See the digital facsimile on E-codices: <http://www.e-codices.unifr.ch/fr/fmb/cb-0047/1v/0/Sequence-807> (Accessed July 27, 2017).

3.3. Transcription Issues Related to Automated Collation

The screenshot displays a software interface for collating TEI XML files. On the left, a 'Witness List' panel shows several files: TEI-O2.xml, TEI-O1.xml, Catullus-TEI-Bod47.xml (marked as the base), TEI-G2.xml, and TEI-G1.xml. Each entry includes a 'Difference from base' progress bar. On the right, the 'Catullus-TEI-Bod47.xml' text view shows a list of lines (L17 to L19). Line L4 is highlighted with a red box and contains the text 'CredoCorde ut quuntum gravis acquiescat ardor'. Several words in this line are highlighted in blue, indicating corrections: 'desyderio' (replacing 'Credo'), 'libet' (replacing 'Corde'), 'quuntum' (replacing 'ut'), and 'acquiescat' (replacing 'ardor'). Other lines also show corrections, such as 'mihi' in L7 and 'zonam' in L9.

Figure 3.5: Deletion and addition appearing both in the text (Juxta Commons). Retrieved from my personal account (July 27, 2017) .

corrections to the text of the first hand). However, it may be useful to visualise among the witnesses a hand which has made many corrections, or which can be related to another manuscript. Therefore, the transcription encoding should be prepared with those issues in mind. The transcriber needs to decide what must be considered a witness for collation purposes: should each hand become a separate witness? How will it be encoded: as a fragmentary witness, or by completing the gaps with the text from the first hand, at the risk of creating a pseudo-witness that never existed? This issue was also raised for other incomplete witnesses, such as *scholia* or editorial conjectures (see Section 2.4.1). And then, this must be balanced with the need for a convenient visualisation. If only one correction in the entire text can be attributed to a particular hand, is it worth creating a new witness that may add confusion to the results? The transcriber should also be aware that the encoding of scribal corrections with different tags in TEI P5 may affect the visualisation of the collation results, as demonstrated with Juxta.

3.3.3 Tokens

When the transcriptions of witnesses are passed to a collation tool in order to be compared, the transcriptions are transformed as they are processed through each step of the Gothenburg model (Section 2.4.4). Even the simplest plain text transcriptions will be divided in a series of tokens, which will be eventually normalised, and finally will be returned in a completely different output format from the original transcription input. Here again, it is very important to understand the implications of these transformations, in order to prepare adequate transcriptions.

3.3. Transcription Issues Related to Automated Collation

3.3.3.1 Tokenisation Process

The first issue concerning tokenisation is how the process is performed by the collation tool. What characters are used to delimit words boundaries: whitespace characters, punctuation marks? If this is not satisfying, what are the options available to make sure that the witnesses will be tokenised correctly according to the editor's need? There are two approaches to influence the tokenisation process.

The first approach is to interact directly with the collation tool and to modify the tokeniser function. For instance, the participants of the Code and Collation workshop in Amsterdam learnt to modify CollateX's tokeniser function. The default behavior of CollateX is to divide the text into tokens at whitespace characters and to treat punctuation marks as separate tokens. However, this is not always the best solution: the contraction *Adam* (for Amsterdam) would become three different tokens (A / ' / dam). Therefore, the tokenisation was modified with the help of regular expressions so that only final punctuation marks would become tokens, but not the punctuation marks followed by other letters. As a result, the contraction *Adam* would become one single token. This approach may be efficient, but it is not always possible to apply: the collation tools do not always let users modify the tokenising process, and in some cases users need to have sufficient coding knowledge in order to be able to do it.

The second approach is to mark directly the word boundaries in the transcription, for instance with the tags `<w>` or `<seg>` available in the TEI. This approach is particularly useful in the case when the same transcription generates different outputs. For instance if there are tags for abbreviations or for choices between an original form and a regularised one for the same word: the transcription can be used to generate both the original form and the regularised form for the same token, so that the same word needs to be processed twice in parallel. The same word may also be processed twice (or more) if it contains additions or deletions, so that different witnesses can be extracted from the same word for different hands or different layers of text. In that case, it may be useful, or even necessary to mark up the token's boundaries directly in the transcription.

3.3.3.2 Loss of Context

The second issue of tokenisation is the loss of context and information that arises when markup is removed or ignored during collation. While collation tools may accept various input formats such as XML, HTML, JSON, Microsoft Word or PDF documents, and so on (see Section 2.6.2), these input documents will be processed

3.3. Transcription Issues Related to Automated Collation

in order to extract plain text tokens. After the witnesses have been processed to extract plain text tokens, the information that had been carefully encoded in transcriptions is no longer connected to the collation results. When preparing transcriptions, editors may thus need to consider what elements they will need to access in addition to the textual tokens, and how to access those elements again after the collation has been performed. Several solutions have been applied to work around this issue:

Juxta In the Juxta Commons interface, the link between transcription, witnesses and collation is visible in the top half of the screen. The witnesses present in a collation set, and the transcription from which the witnesses are extracted are all highlighted in yellow. It is possible to visualise the page breaks and line numbers if those have been encoded in transcription. However, when examining the result of collation, there is no way in Juxta Commons to select one specific token and see it in its transcription. The user needs to go back to the right transcription file and search for that particular token. In Juxta Desktop interface, on the other hand, it is possible to see the transcription by clicking on a token (see also Section 7.2).

JSON format Another way to create a link between the collation result and the transcription is to use a system of identification. The projects which have implemented a system of identification have done so with the CollateX JSON input. For instance, in the Digital Mishnah project, each token receives a unique identifier when the transcription is transformed into JSON, so that the identifier is still available in the collation results and can be used to trace a particular token back to its exact location in the transcription file. The JSON input of the IGNTF project also has a detailed system of identification for each token, with properties such as ‘index’, ‘siglum’, ‘verse’ and ‘reading’.

It is also possible to take advantage of CollateX’s JSON input to keep some elements of markup associated with a token. For instance, Birnbaum (2012) has explained how to track XML markup information during collation¹². In this tutorial, Birnbaum describes the Python code which transforms an XML transcription into CollateX’s JSON input so that the internal markup of a word does not affect how it is collated: for instance, both words ‘maint’ and mai<abbrev>n</abbrev>t should be matching when only the string of characters are compared and markup is

¹²The material was prepared for a workshop on CollateX at the Digital Humanities conference 2015 in Sydney. It is available from Birnbaum website: <http://collatex.obdurodon.org/xml-json-conversion.xhtml> (Accessed July 30, 2017).

3.3. Transcription Issues Related to Automated Collation

ignored¹³. The fact that one version contains an abbreviation should not influence the final alignment. Therefore the transcription generates two properties for a token with markup: a property with the original token that include markup as well, and a normalised property where markup has been stripped. As a result, the XML markup is still present in the collation output, and can be used to display the tokens according to an editorial standard: the abbreviation can be displayed in italics, ‘*maint*’, or in parenthesis, ‘mai(n)t’.

A similar workaround is to include the XML markup elements directly into CollateX JSON format. The elements that the editor wishes to retain are transformed into token properties: for instance, the page numbers and line numbers encoded in the elements <pb> and <lb> may be transformed into a location property (see Chapters 4 and 8). It should be noted here that the decision to keep or to ignore a set of markup elements is a critical decision, and that it should be kept in mind when analysing the collation results. As Elli Bleeker has argued in her dissertation, the preprocessing from XML to JSON involves an additional layer of interpretation (Bleeker 2017, 147). In her thesis, Bleeker explores different options to retain markup for draft manuscripts which contain many layers of revisions and are therefore heavily encoded, such as standoff properties in Multi-Version Documents (Schmidt 2009) and ‘Native XML Collation’.

Standoff Markup Schmidt (2009) has proposed the Multi-Version Document format to deal with automated collation (see Section 2.4.3). In principle, the MVD ‘encodes all the complex overlapping structures of a set of versions, [so that] the markup of an individual version can be much simpler’ (Schmidt 2010, 351). This simple markup is then transformed into standoff properties to link a fragment of plain text to the corresponding attribute: for instance, the text encoded within <l> elements would have a standoff property ‘line’ (Schmidt and Eggert 2015). However, Schmidt was more concerned with issues such as overlapping hierarchies in encoding than the problem of a loss of context that arises when transcription markup is ignored during collation. In his view, markup should be as simple as possible, and the identification of relationships between versions (‘what is a variant of what’) should be computed automatically (Schmidt 2010). In fact, the Charles Harpur Critical Archive, which make use of the MVD format, does not seem to provide examples of how these standoff properties may be used, for instance in collation visualisation. Admittedly, only few texts do have more than one version (Schmidt, private correspondence). And in tables visualisation, while some witnesses are

¹³This encoding example is from the electronic edition of *Partonopeus de Blois* (Eley et al. 2005). The XML tag <abbrev> is not part of the TEI set of elements.

divided into several layers, it is not clear to what exactly those layers correspond to (see Section 8.2 for an example of MVD collation table). For Bleeker (2017), the MVD format is not sufficient to keep track of markup context for complex genetic editions, where transcriptions encode valuable editorial information necessary to create an informative collation output (Bleeker 2017, 108).

Native XML Collation Bleeker (2017, 130-145) proposes a new approach to collation that would not necessitate to remove any markup encoding. Still at an experimental stage, the approach called ‘Native XML collation’ was proposed in order to satisfy the need of genetic textual criticism, for which there are many layers of text in a single manuscript. These layers are carefully analysed during transcription, in order to retrace the creative process of the writer. As such, those transcriptions contain essential information for the editor, who must be able to access when studying the collation results. The idea behind Native XML Collation is to collate entire XML transcription documents, including the markup added by editors.

The technique has two advantages: it reduces the amount of processing and interpretation that the editor must perform when removing markup from transcription, and it takes into account all the editorial analysis expressed in markup encoding, such as the organisation of the consecutive layers of variation present into a single witness. In practice, Native XML collation divides the transcription between text and markup tokens (see for instance Section 2.4.4.1 or Chapter 4 on the topic of tokenisation). The collation result is transformed back into XML with added markup to indicate differences. The distinction between the original transcription markup and the new collation markup is expressed with the help of a namespace¹⁴. At the moment, the Native XML Collation is a promising approach, but still in a prototype format, and users require sufficient XML skills to be able to interpret the collation output (Bleeker 2017, 141)¹⁵.

3.4 Conclusion

Until Native XML Collation becomes a fully robust solution to collation, a means of transforming transcriptions into a valid collation format will still be required. Therefore, editors should be aware of those transformations applied to their tran-

¹⁴A namespace is a way of labelling a group of elements’ (Burnard 2014b). In this case, the group of new XML markup are labelled with the prefix ‘CX’, which represent the result of collation with CollateX.

¹⁵The prototype is available on Github: https://github.com/bleekere/xml_collation (Accessed July 30, 2017).

scription, and how those transformations might influence the way they decide to encode the transcriptions so that the collation results are satisfying. In particular, editors need to decide which elements they wish to keep during collation so that they are still available in the results. Editors need to consider as well how to encode those elements so that they can transform the transcriptions into collation format as easily as possible.

The issues of witnesses and tokenisation imply that decisions need to be made *a priori* at the transcription stage, which will directly impact on how the collation is performed, and the quality of the collation results for further analysis. In the next chapter, the question of tokens and readings are considered more in detail, based on the examples of tokens which include more properties: I will propose a model to represent textual readings, and I will show how it can be implemented with the JSON input of CollateX.

THIS chapter presents the material of an article that I submitted following a presentation at the symposium ‘Versioning Cultural Objects: Concepts, Structures, and Expressions’ organised by Roman Bleier and Vinayak Gupta at Maynooth University, in December 2016. The article is titled ‘Towards a Model of (Variant) Readings’ and will be published in 2018. The symposium was focused on versions, in the very large sense of cultural objects which are compared and may exhibit differences. The symposium’s organisers were interested in defining versions, from the point of view of various area of studies, such as textual studies and collation, manuscript studies and archiving, but also other fields such as musicology, film studies and oral history, and so forth. Electronic modelling of versions and their representation in a digital format were the two other main concerns of the symposium.

In the context of this symposium, I have considered the modelling of readings, an essential component of textual versions, and the representation of readings in a digital format. Although the term ‘version’ may describe a major rewriting of a work, possibly by the author themselves, here I refer to version as one instance of a text that is compared during collation. As we have seen previously (Chapter 3), it may not be entirely accurate to say that witnesses are collated, since it is often the case that one single witness contains more than one textual version (either because of the intervention of different hands, or a succession of revision layers by the author).

Each version to be collated contains readings, i.e., the particular word or words found at a given point in the text. A version is determined, amongst other characteristics, by the *differences* in the words found in the text, or variant readings. Variant readings are important since they provide valuable information regarding how versions are related to each other and how the text evolved through transmission. We have seen in Chapter 1 that variant readings need to be precisely defined (see p. 42).

A	The	bright	star	shines	upon	the	world.
B	The		sun	shines	upon	the	world.
C	The	bright	stars	shone	upon	the	world.
D	The	bright	sunne	shines	upon	the	worlde.

Figure 4.1: Readings.

This chapter will focus on the modelling of ‘readings’, arguing that formalising this concept is necessary in order to define, and model, the concept of ‘variant readings’. We will show how reading was a technical term that was used quite consistently through the ages, until it was defined with precision. Then we will establish the basis for a model by selecting important features of textual readings according to the previously examined definitions. These features, such as the textual content (or absence thereof), its size, and location in the text, will be discussed, raising various issues. This chapter will also address digital representation of a reading by focusing on one implementation: the JSON data format used in conjunction with collation programs such as CollateX (see Section 2.6.2). As we will see, the concept of a variant reading may depend on the tradition of the text in consideration, and a variant in Homeric epic is different from a variant in a medieval tradition. The concept of variant is also dependent on the purpose of the comparison: a scholar attempting to reconstruct a stemma or a linguist may need to examine different variants. Therefore, a model of a reading should make it possible to distinguish different sets of variants depending on the context, and we will examine how the JSON implementation makes it possible with a few examples.

Let us consider the example of figure 4.1, where four versions of a sentence are aligned. When comparing the sentences of A, B, C, and D, some readings can be considered equivalent in all four sentences, such as *The* or *upon*; other readings are different and change the meaning of the sentence: the absence of the adjective *bright* in sentence B, the triplet *star/sun/stars*, and the verbs with different tense (*shines* and *shone*). Finally, some readings are different, but may not alter the sense of the sentence (such as *world* and *worlde* or *sun* and *sunne*). Readings are thus divided between equivalent readings and different readings, and among the different readings a certain number may be considered variant readings (see figure 4.2). Other approaches reject altogether the concept of variant because it implies a deviation from an invariant text (see below p. 143).

In the short collation extract of figure 4.1, there are four places where differences

appear in the text. However, not all differences between the readings are necessarily considered variant readings in any possible context. Scholarly opinions on this point range widely: from the view that every difference is a variant (Andrews 2012), to considering only a limited number of ‘significant’ differences to be variants, for instance, in the context of New Testament criticism, and therefore it is not enough to define a variant simply as a difference:

The common or surface assumption is that any textual reading that differs in any way from another reading in the same unit of text is a ‘textual variant’, but this simplistic definition will not suffice. Actually, in NT textual criticism the term ‘textual variant’ really means - and must mean - ‘significant’ or ‘meaningful textual variant’ (Epp 1993, 48).

In fact, the concept of the variant has evolved with time and according to several theories. Since the nineteenth century, many scholars contributed to the development of a method for the establishment of genealogical relationships between manuscripts: the so-called Lachmann method. Maas (1958) in particular focused on a specific category of differences: shared errors, or indicative errors, can be used as a guide in order to assess the witnesses of the text and determine their relationships into a *stemma codicum*, or genealogical tree of textual witnesses. Greg (1950) separated variant readings between accidental and substantial, following the idea that some differences (substantials) have more importance than others (accidentals):

[W]e need to draw a distinction between the significant, or as I shall call them “substantive”, readings of the text, those namely that affect the author’s meaning or the essence of his expression, and others, such in general as spelling, punctuation, word-division, and the like, affecting mainly its formal presentation, which may be regarded as the accidents, or as I shall call them “accidentals”, of the text’ (Greg 1950, 21).

In the twenty-first century, scholars started to compare textual variants to DNA mutations and applied concepts from evolutionary biology and phylogenetics to textual criticism (Barbrook et al. 1998; Salemans 2000; Heikkilä 2014). Lastly, in opposition to the distinction between accidental and substantial variants, Andrews (2012) suggested a big data approach where every difference is a variant.

With the introduction of Lachmann's method, shared errors became the object of scholarly attention, and much work was done on the description and classification of the kind of errors committed by scribes who were copying manuscripts by hand. The cause of the error, as well as its conscious or unconscious character, is generally taken into account. Since the conscious modifications of scribal corrections were often attempts at improving or restoring the text, the terms *innovations* and *secondary readings* are frequently preferred to errors. One of the most comprehensive review of errors and innovations was published by Havet (1911), but other scholars have proposed variant typologies (Petti 1977; Love 1984; Reynolds and Wilson 1991). These typologies often divide errors into four types: additions, omissions, substitutions and transpositions (Petti 1977). When the scribe is consciously modifying the text, Petti (1977, 28-29) refers to scribal corrections as insertions, deletions and alterations instead of additions, omissions and substitutions.

In parallel, many fields of study have offered their own definitions for variants according to their needs. From oral traditions such as Homeric epic to early printing, from medieval traditions to genetic criticism, from linguistics to phylogenetics, variants take many forms depending on the context: multiformity (Nagy 2010), early or late states (Dane 2003), variants at the sentence level (Cerquiglini 1989), open or alternative variants (Gadda 1983), type-2 variants (Salemans 2000), and so on.

The task of proposing a model for variant readings, which would be suitable in any of the possible contexts, seems at best challenging, if not impossible. Rather than dealing directly with variants, I will focus on modelling readings, especially textual readings. Not all readings are *variant* readings, but variants are always readings which differ in some respect from one another (see figure 4.2). Once readings have been modelled, variant readings could be more easily modelled as a set of readings, with various criteria according to each discipline (V1, V2, V3). However, modelling those subsets will not be in the scope of this dissertation. In order to propose a model for readings, we will first review the origins and usage of the term as well as its definitions in Section 4.1. The analysis of definitions will provide a first outline for a model, which will be discussed in Section 4.2.

4.1 Readings in Context

Reading is a technical term that has long been used in the context of textual criticism and philology. It was already attested with Alexandrian critics: terminology included *graphe* (what is written), and *anagnosis* (what is read, a reading). The

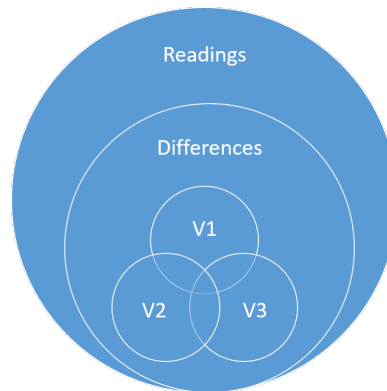


Figure 4.2: Readings, differences and variants.

Latin equivalents are *scriptura* and the most common *lectio* (Montanari 2015, 26). The terms used by scholars of Antiquity imply a distinction between the words that are actually written on the page as opposed to the interpretation of the text. In English as well, a reading implies a form of interpretation; it could be read in more than one way. Here are a couple of examples where the words *scriptura* and *lectio* are used to qualify textual variation:

*Obolus, id est, virgula iacens, adponitur in verbis vel sententiis superflue iteratis, sive in his locis, ubi **lectio** aliqua **falsitate** notata est, ut quasi sagitta iugulet supervacua atque falsa confodiat.* Isid. 1.21.3¹.

The obelus, that is, a horizontal stroke, is placed next to words or sentences repeated unnecessarily, or by places where some **passage** is marked as **false**, so that like an arrow it slays the superfluous and pierces the false (Barney et al. 2006).

*Et idcirco inportunissime inquit fecerunt, qui in plerisque Sallusti exemplaribus **scripturam** istam **sincerissimam** corruperunt.* Gell. 20.6.14².

'And therefore,' said he, 'those have acted most arbitrarily who in many copies of Sallust have corrupted a **thoroughly sound reading**' (Rolfe 1927).

Here the nouns *scriptura* and *lectio* have been emphasised, as well as the terms

¹Isidore, Bishop of Seville (c.560–636), is the author of the *Etymologies*, an etymological encyclopedia compiled in the early 7th century.

²Aulus Gellius (c. 125 – after 180 AD) is a grammarian and author of the *Attic Nights*, a collection of notes on many topics such as grammar, philosophy, or history.

which qualify them. As it is shown, during Antiquity, there was a strong focus on whether a reading is corrupt or sound. When producing a new literary book, Hellenistic scholars used to correct a single copy of a work, instead of comparing as many copies as possible as modern editors do. This practice led Hellenistic scholars to become correctors of a specific work, and some experts compared them to editors (Montanari 2015). Therefore, the need to distinguish between authentic and spurious readings arose, which may have motivated the dichotomy between sound versus corrupt readings, true versus false. The concept of variant reading, however, appeared much later during the Renaissance.

In the Renaissance, Humanist scholars who were rediscovering and editing classical texts of Latin and Greek literature started to deploy technical terms that would become the base of the language of textual criticism. Silvia Rizzo's *Lessico Filologico degli Umanisti* 1973 provides invaluable information about the vocabulary in use amongst famous Humanists in the fourteenth and fifteenth centuries. By analysing their correspondence and publications, Rizzo was able to extract global definitions and explain what they meant when they used a given word. During the Renaissance, as Rizzo (1973, 209-213) shows, *lectio* and *scriptura* continued to be used as synonyms in much the same way as in Antiquity, for a passage of a text that can be read in a manuscript or an edition. Renaissance scholars would apply the terms to readings from manuscripts as well as conjectures by other Humanists, and would mostly describe those readings as either correct (*recta, sincera*) or incorrect (*corrupta, mendosa*) according to their judgement.

At the same time, the concept of 'variant reading' started to be used more precisely with *varietas* (diversity) and in expressions where *lectio* or *scriptura* were used in connection with the adjective *varius*. Lorenzo Valla and Girolamo Avanzi have both used *varia lectio* and *varia scriptura* to describe a portion of text with different possible readings, as reported by Rizzo (1973, 213)³. Valla was accused by Poggio Bracciolini of having presumptuously corrected a verse from Sallustius' first Elegy. Valla replied to Poggio that he did not emend Sallustius but merely chose one reading in a passage that varies (*varia scriptura*), even though the reading was attested only in very few manuscripts⁴. Another scholar, Avanzi, was asked for his

³Lorenzo Valla (1407 – 1457) and Girolamo Avanzi (1493 – 15..) were Italian Humanists of the fifteenth and sixteenth century.

⁴*Nam quomodo videri possum emendare Sallustium, qui, incertum est, an sic scriptum reliquerit, ut me tu ais emendare voluisse? Ego tantum ex varia scriptura, quid mihi satis videatur, pronuncio. At cur praeponis, inquires, illam scripturam, quae in paucioribus codicibus est? Praepono, non ut Sallustius emendem, sed ut admoneam sequendum, quod plurimorum confirmat autoritas.* (Valla 1540, 263). The discussion can be found in Valla's *Antidoti in Pogium*, book I, in the section on Sallustius.

opinion on a difficult passage from Catullus I, 9. He offers no solution of his own to emend the corrupted text, but he sends to his correspondent a list of conjectures (*varia lectio*) proposed by others⁵.

The usage of *lectio* and *scriptura* illustrates two contrasting approaches on readings and variant readings. Usually, a reading becomes a variant only when compared to another reading (Froger 1968, 80); variant also implies a deviation from a norm (Colwell and Tune 1964, 253)⁶. On the other hand, a variant can be one among multiple possible alternatives, in a place where at least two witnesses disagree as to what the text is. Consequently, Colwell and Tune (1964) decided to refer not to variants, but to variation-units. This approach is shared by genetic criticism, which reject the existence of an invariant final text, against which variant readings are compared (de Biasi 2000).

In the twentieth century, formal definitions of reading can be found for instance in editing manuals, dictionaries or lexicons⁷. Stussi (1994) defines a reading as ‘a passage from a transmitted text as it appears in a given witness’⁸. In 1968, Froger describes one of the first collation software and gives a very precise definition for a reading:

The form or content of the text in a given place is a ‘reading’, that is to say what we read at this location. Any manuscript, for instance the original, can be considered regarding its content as a collection or set of readings, which are the text elements at various levels: chapter, paragraph, sentence, word, syllable, letter, and even punctuation or accents (Froger 1968, 9)⁹.

⁵*non meam, sed **variam lectionem** accipies illius versus in primo carmine Catulli* (Avanzi 1495, f. a5v).

⁶Colwell and Tune explain that the ‘norm’ against which variant readings are compared may be different depending on editors: ‘So what is commonly done in practice? Some particular text is chosen — often at random — for the norm. Either we use a printed text such as the Textus Receptus, sometimes an edition by Tischendorf, Westcott-Hort, or Nestle; or, we may use the text of a particular MS whose textual affinities are already known, e. g., Vaticanus or Alexandrinus’ (Colwell and Tune 1964, 253).

⁷A series of definitions can be found in the Lexicon of Scholarly Editing: <http://uahost.uantwerpen.be/lse/index.php/lexicon/reading-variant/> (Accessed October 31, 2016).

⁸Con lezione di un determinato testimone si designa un passo del testo tramandato così come compare in tale testimone (Stussi 1994, 89).

⁹La forme ou teneur du texte en un lieu donné est une ‘leçon’, c’est-à-dire ce qu’on lit à cet endroit. Un manuscrit quelconque, par exemple l’original, peut donc être considéré, quant à sa teneur, comme une collection ou un ensemble de leçons, qui sont les éléments du texte à différentes échelles : celle du chapitre, du paragraphe, de la phrase, du mot, de la syllabe, de la lettre, et même

This definition adds more precision: a reading is a textual element ('what is read'), and it can be of various scope, from the smallest punctuation marks to whole chapters. How can these definitions of a reading lead to a first example of a reading model?

4.2 Modelling a Reading

The purpose of data modelling in the Humanities is to describe and structure information about real-world or digital objects in a formal way, and so that this information becomes computable (Flanders and Jannidis 2016, 229-230), and that it can be manipulated and queried with the help of a computer in order to answer questions. Ultimately, the purpose of modelling readings is to help determine if two given readings may be considered variant readings in a specific context. Flanders and Jannidis (2016, 234) suggest modelling textual variants in a scholarly edition by classifying variants according to some scheme, such as accidental versus substantial, or orthographical versus lexical, which corresponds to a consensus within the community.

As we have seen, however, variants can represent something very different depending on the approach (stemmatics, linguistics, etc.) and textual traditions (oral, medieval, early printing, and so on); therefore readings need to be modelled independently of their function in textual criticism, but with enough information to decide what is a variant in those distinct contexts. It may be helpful to consider the distinction between readings and variants in the framework of Sahle's wheel of text model (Sahle 2013, 45-49). According to Sahle's model, there are several perspectives on the text which are closely related. Different criteria help determine if texts are identical or not according to the perspective adopted. For instance when text is considered as a language, the English and French translations of Calpurnius are considered different (Sussman 1994; Aizpurua 2005). On the other hand, if text is considered as a work, both translations represent the same work and are identical. These perspectives are equal and not in a hierarchical relationship, which is why they are represented as a wheel (figure 4.3).

In Sahle's model, readings can be considered as a part of the text as Document (Text_D), whereas variants are part of the text as Version (Text_V). The text as Version is further divided into subcategories, such as Text_K , a canonical representation of the text which aims at identifying the best (true) text among the different versions. With this framework in mind, the characterisation of readings as authentic or corrupt

du signe de ponctuation ou des accents' (Froger 1968, 9).

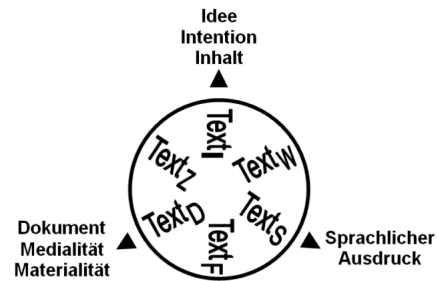


Figure 4.3: Sahle's wheel of text model (Sahle 2013, 47).

does not make a good model for readings, since it represents rather variants than readings. Therefore, the more recent definitions of readings may provide a better starting point to the model than the true/false distinction previously applied to readings.

Models are simplified representations of an object of study, a selection of features among all available (Pierazzo 2015, 44-45). From the overview of the term reading provided in the previous section, in particular the definition of Froger (1968) and Stussi (1994), features which apply to a reading can be inferred, namely that a reading:

- conveys textual content;
- has a precise location in the text (also referred to as *locus*);
- can occur at any level of the text, and thus have various sizes;
- is transmitted by a witness.

4.2.1 Issues

These features need to be discussed in more detail. For instance, is it too restrictive to limit a reading to textual content? What about decorations, mathematical diagrams and other non-textual elements? Historians of Greek, Arabic or Egyptian mathematics have acknowledged the need to collate and critically edit mathematical diagrams instead of simply providing corrected figures to fit modern standards. Raynaud (2014) created a stemma for the *Epistle on the Shape of the Eclipse* by Ibn al-Haytham, a mathematical treatise from the eleventh century, using the mathematical diagrams present in the text. In order to collate diagrams and apply

Lachmann's method of shared errors, Raynaud had to select 'characters' from the diagrams, which could be regarded as an equivalent for readings. This suggests that it is possible to define and model readings for mathematical diagrams. It would be different from textual readings, but as important for the comparison of versions from traditions of mathematical texts. Other types of content could include, and are not limited to, 'visual content' such as decorations, illuminations, or an artist's sketches. Musical compositions need as well to be collated and critically edited, however readings and variants are quite different from textual readings and variants: pitch and metrical values are significant features of a musical note to compare, for instance (Broude 1991).

Let us focus here on readings as textual content. Other issues arise with gaps, and lacunae for instance. Can the absence of text, such as an omission, a so-called lacuna, be considered a 'reading' as well? It would seem that the absence of text is by definition not a reading. It cannot be read in the witness, even if it can often be defined by the other features listed above (the size of the missing text may be difficult to evaluate in some cases). However, a missing reading may be significant for the manuscript tradition: since a missing passage is difficult to restore by conjecture, a lacuna can often be used as a significant error during the stemma construction (Reynolds and Wilson 1991, 213). A lacuna that helps in grouping manuscripts and building the stemma therefore needs to appear in the collation. How should the absence of text be modelled? As a special kind of reading, or separately? In this model, lacunae were included as readings without any content if there is a physical evidence in the document, and lacunae without evidence but which may appear after collation are considered as an absence of reading.

Conjectures — reconstructed readings proposed by scholars, which are not present in any witness — seem to qualify as readings according to the features listed above. However, one may ask if conjectures are indeed transmitted by a witness. Conjectures are obviously constituted of textual content of a certain size, meant to be read at a certain location; can they nevertheless be considered as 'transmitted by a witness', where they are published in a scholarly article instead of an edition? According to Greetham (1994) a conjecture 'is involved only when an editor reconstructs or creates a reading which is not extant in any of the witnesses' (Greetham 1994). A conjecture is thus a new reading, with no prior witness evidence, but with an established origin that can be traced to a particular scholar or scribe. In this sense conjectures are considered as part of the reading model.

The location of a reading in the text is not as easy to formulate as it seems. It would not be enough, for instance, to number each word, since the count would then be different for every witnesses. Even a reference system such as the canonical citations for classical texts can have limitations, when it comes to precision at the word level. Citations such as Pliny nat. 11.4.11 or Vergil ecl. 10.69 refer respectively to the *Natural History* of Pliny the Elder, Book 11, Chapter 4, paragraph 11, or Vergil's *Eclogues* 10, verse 69. The minimal text unit here is the paragraph or the verse, not the word, and at some point in the text, there will be chapters or verses with different word numbers. The location in the text can only be accurately expressed after collation has happened and readings have been aligned with each other. Canonical citations have been formalised in digital formats such as DET (Robinson 2017) or the Canonical Text Services (CTS) Data Model (Crane et al. 2014).

Text can be seen as both a conceptual (immaterial) sequence of words and punctuation from which a reader derives meaning, and as a material sequence of marks on a document. Readings are also made of marks recorded on a physical document, besides being part of the immaterial text, thus a reading has a both a location in the text and a location in the document where it appears. The document location may be rendered with varying degrees of precision: for instance with folio or page number of the witness in which it appears, with an additional line number, or with a very precise set of coordinates for a two dimensional surface on the page¹⁰.

Finally, it is worth asking if different levels of reading (letter, words, sentences and so on) call for different models and how those levels relate to other existing models. For example, how would the letter level relate to the model used by the DigiPal framework Stokes (2012) to describe letters from a palaeographical point of view? How would the sentence level relate to the treebank model (Haug 2015) used to annotate textual corpora? How would the different levels be linked together, if the purpose of the scholar is to collate at different levels? Monella (2014a), for instance, decided to collate a text from the Latin Anthology at three different levels, which are called graphical (letters, punctuation), alphabetic (the abstract representation of a letter in a particular alphabet) and linguistic (word) levels. The different levels may certainly be characterised by additional features of their own. Readings at the word level may also have morphological features, such as lemma or part-of-speech properties or even a phonetic transcription. These linguistic annotations could be useful when comparing readings during collation. For instance, words that do not share gender, number, case, and lemma, could be considered variants. In

¹⁰See for instance the TEI P5 Guidelines chapter 11 for representation of primary sources, in particular the section on digital facsimiles (TEI Consortium eds. 2017b, §11.1).

the case of oral sources, a different pronunciation may be considered a variant. Layout could also be significant in some contexts: the same word written in bold, or italics or in colour could signal a variation. For instance, Caton (2009) argues that a transcription loses information when a word originally written in italics, to denote emphasis, is transcribed into roman font. At the line level in poetry, metrical patterns would be an important feature. At the sentence level, syntactic information about the subject, object, verb and other elements of the sentence may be an important feature. This information could be particularly interesting for the comparison of versions translated in a different language from the original (see Spencer and Howe 2004, 266). In principle, the comparison happens always with readings at the same level: letters are not compared to words, or words to paragraphs.

It is worth noting, however, that even if the word level is used during collation, it may be that in the result, words will be grouped together to form a new reading at a different level than the word level (a variation unit that falls between the word and sentence levels). Considering the sentences from the fictive witnesses in figure 4.1, the groups of words ‘sun shines’, ‘star shines’ and ‘stars shone’ may be considered as one reading only, for the purpose of studying the collation results. When there are many variations close to each other, it may be difficult to decide how to group words into readings, if they should be grouped at all, and the readings may be different according to different editors. One could decide to group words instead as ‘bright star’, ‘sun’ and ‘bright stars’, with the verb as a separate reading.

4.2.2 Model

In summary, the model could be expressed as in figure 4.4: readings can either have content or not. In both cases, the reading has the general properties outlined above, such as the witness in which it is found, a position both in the text and the document, or a level of precision such as the word level. When the content is present, it can be textual content or another type of content such as diagrams or illustrations. The textual content has a second layer of properties: syntax, morphology, phonetic, layout, and so on. Depending on the level of the textual content, features may differ. At the sentence level, it is possible to describe the relationships between words or group of words: ‘The bright star’ is the subject of the verb phrase ‘shines upon you’, a relationship which is more difficult to represent at a word level. The other types of content would have their own properties, such as the characters in diagrams described by Raynaud (2014).

On the other hand, readings without content cannot be described with those

4.2. Modelling a Reading

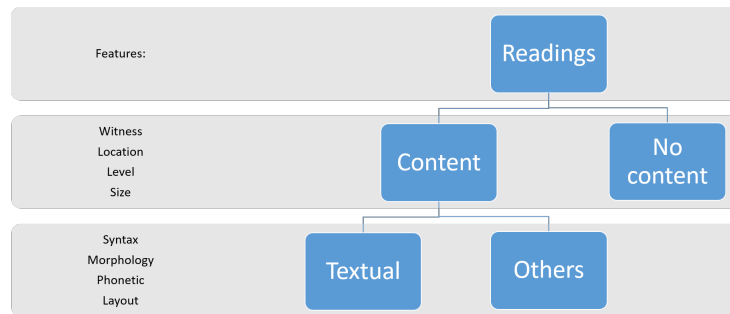


Figure 4.4: Model for readings.

additional features. There are other concerns regarding an absence of content, or lacuna. First, there are different reasons behind the presence of a lacuna. The missing text could have been present in the manuscript but is no longer readable by scholars, due to damage or to missing pages. In other cases, the copyist marked a lacuna explicitly, with a series of dots for instance, because the text was already missing in the witness serving as the exemplar. The scribe may also have left a blank space to be filled later, and which was never completed. In medieval manuscripts, this would happen easily for material such as titles, initials or coloured text, which were added later often by a different person than the copyist of the main text. Even if the text is absent from every witness, the presence of a lacuna can be indicated by inconsistencies in the meaning, for metrical or grammatical reasons, or by an incomplete content (such as a missing plural 's'). In addition, Dillen (2015) has demonstrated the importance of distinguishing between several types of lacunae in Beckett's draft manuscripts, such as authorial lacunae as opposed to editorial ones.

Lastly, the lacuna may not be perceptible, unless the witnesses are collated. The collation result could then expose in a witness the absence of a reading which was present in at least one other witness. In figure 4.4, the absence of 'bright' in witness B would have gone unnoticed unless exposed by the collation against the readings in sentences A and C. The reading may be absent because the scribe did not copy it, whether voluntarily or not, or because it was absent altogether from the exemplar. This kind of lacuna does not belong to the reading model, but only to the variant model: a variant arises either if two readings are considered different, or if a reading is compared against an absence of reading. It is then important to distinguish between the reasons behind a lacuna: is the text present but no longer accessible? Is there a mark indicating that the text was already illegible to the copyist? Or is there no evidence?

4.3. Digital Representation: from Reading to Token

4.2.3 Selecting Sets of Variant Readings

Given two or more readings at a place of variation, the comparison of the reading's features could help to identify in what aspect the readings differ (Monroy et al. 2002; Smith and Lindeborg 2016). This comparison could then lead to decide when those readings become variant readings. Let us consider pairs of readings from the sentences in figure 4.1: comparing the features of *stars* and *star* would show a difference in number, plural and singular, but the lemma would indicate that they represent the same word. It would thus be a grammatical difference. The readings *sun* and *star* have a different lemma, and therefore represent a lexical difference. Two words which share all features (lemma, part of speech and so on) and show no other difference than their original written form would represent an orthographical difference, or graphical difference for languages which have no standardised orthography. In different scholarly contexts, the features of readings could be used to define criteria which are then applied to isolate the relevant set of variant readings¹¹.

Let us consider three different contexts. First, if all differences are considered variants, then readings which display any difference among their features will be considered variants. On the other hand, orthographical differences and other accidentals are often not considered variants while editing a text (Reynolds and Wilson 1991; Love 1984), therefore the distinction between non-orthographical or orthographical differences allows to select the set of readings which represent grammatical or lexical differences, and ignore spelling variants. Finally, it would be possible to examine spelling variants only. Linguists would be able to select only spelling variants only, particularly significant for the study of language evolution (Vierros and Henriksson 2016).

These three contexts will be further examined in Section 4.4 below, using a practical example. The next section will first deal with the representation of a reading in digital format.

4.3 Digital Representation: from Reading to Token

To translate the concept of a reading, as defined by centuries of textual scholarship, into digital representation, it seems there is already a counterpart in computational linguistic terminology: the token. Tokens are commonly used for lexical analysis in computer science, as a sequence of characters with an identified 'meaning' is con-

¹¹These criteria would not necessarily be applied at the time of recording variants, but also after variants are recorded, to identify only the variants relevant to a specific context.

4.3. Digital Representation: from Reading to Token

verted into a token. If manual collation is the comparison of readings, automated collation is the comparison of tokens. Automated collation is the application of computing methods to the comparison of textual witnesses: instead of comparing manually the existing versions of a text, digital transcriptions are collated with the help of an alignment algorithm. The concept of tokens was already introduced in Section 2.4.4, as they are the unit of textual comparison in the Gothenburg model of automated collation. Tokens were also discussed in Chapter 3: tokens are marks on a document that can be interpreted as text. In the collation model, a token becomes a reading when it is read and interpreted by someone, whereas in the Gothenburg model, readings (sequences of characters) are transformed into digital tokens.

The parallel between Froger's reading definition (p. 143 above) and a token is clear. In the Gothenburg model, a text is divided into a list of tokens which are textual units (a sequence of characters) at a chosen level. This is also how Froger described a text, as a collection of readings, which are made of the text's content taken at a particular level. As such, the tokens share the same features as readings: the textual content of a witness, with a precise location in the text determined by its position in the full list of tokens, and at a specific level.

According to Dekker et al. (2015), a token is a textual unit at 'any level of granularity, for instance, on the level of syllables, words, lines, phrases, verses, paragraphs, text nodes in a normalised XML DOM instance, or any other unit suitable to the texts at hand'. The CollateX documentation more explicitly considers a token as a textual unit that ideally carries meaning, thus above the character level (see Section 3.3.3). At letter level, phenomena such as transposition are much more frequent and reduce the efficiency of the alignment algorithm. For this reason, collation is preferably performed at a higher level, rather than at character level. However useful for the collation process, this restriction does not apply in palaeography where letters are the comparison units. Projects such as Monella's 2014 also require analysis at character level. From a theoretical and modelling perspective, it is thus necessary not to make assumptions about the meaning of a token. The transcription model of Huitfeldt, Marcoux, and Sperberg-McQueen (2008) provides a more adapted description for a token, since they do not make a distinction between tokens as characters, as words, or as other levels¹².

In digital format, the most basic form of a token is a simple string of characters, a

¹²See also p. 115. The transcription model is 'agnostic about whether the [tokens] it is concerned with are those at the character level or those at the level of words and lexical items' (Huitfeldt, Marcoux, and Sperberg-McQueen 2008, 298).

4.3. Digital Representation: from Reading to Token

linear sequence of one or more symbols representing letters, but with no linguistic interpretation attached to them. Nevertheless, collation tools usually offer to normalise tokens in order to minimise what is perceived as insignificant variation: typically, normalisation permits the removal of upper case, punctuation or other aspects (such as, for instance, hyphenation or line breaks in Juxta, white space characters in CollateX) from the tokens that will be compared, so that these would not be considered differences: *the* and *The* would be treated as the same word for the purpose of aligning the versions together. However, if this normalised form is not explicitly included in the token, it will not be available in the results of the collation. For example, in the case when accidental differences are not significant, the pair of readings *sun/sunne* and *world/worlde* may be considered as irrelevant differences and thus should be ignored when searching for semantic variants.

However, given only the string of characters it is impossible to discriminate between a significant variant such as *shines/shone* and other variants such as *sun/sunne*. On the other hand, if the reading *sunne* also includes a normalised form *sun*, it is then possible to compare the normalised form of *sunne* and decide that it is equivalent to the reading *sun*. As a consequence, it could be extremely difficult to distinguish between orthographical or non-orthographical differences without normalised forms, when analysing collation results.

4.3.1 Token Format in CollateX

CollateX makes it possible to distinguish between the original token and a normalised form provided by the user thanks to its JSON input format, where tokens can be represented with various properties (see also Section 7.1.3 and Section 8.4.1). These properties include:

t the textual content in its original form.

n a normalised form of the same textual content.

The property **t** is mandatory for CollateX. The normalised form **n**, if provided, is used to align the texts as accurately as possible. In addition to these two properties, users are free to add as many others as they need.

The input format of CollateX is thus a very effective way to represent readings involving textual content. However, readings without content are problematic, since a token must always have at least a property **t** with a positive value. As a result,

4.3. Digital Representation: from Reading to Token

it is not possible to collate empty tokens, which can be a limitation since lacunae are considered readings in this model and need to be represented as tokens as well.

I have represented lacunae present in the text due to damage, or explicitly marked by the copyist, as tokens with the textual content **t** as ‘...’, and the normalised form **n** as ‘lacuna’, a combination that does not appear elsewhere in the witnesses and therefore cannot be confused with another reading (see Section 6.2.3.5). Lacunae which are revealed by the collation, because a portion of text was omitted by a scribe, are not represented by a token. Instead, CollateX inserts empty tokens in the collation to compensate for the absence of text (Dekker et al. 2015).

As CollateX is used in several projects, their encoding choices may provide further ideas about the representation of readings as tokens. As an example, the Collation Editor, a tool prepared for the collation of the Greek New Testament with CollateX, provides a description of the token’s properties¹³. The Collation Editor provides two layers of normalisation and regularisation: the original token is normalised in a first step into *t*, with operations such as setting the words in lower case. Then, the token **t** may be regularised again into **n** according to rules defined by the user, which are provided through a *rule_match* feature.

Besides **t** and **n**, any additional properties can be provided to the token object, and will be ignored during collation. Nevertheless, these additional properties would still be available in the results for further processing. For tokens at the word level, such properties could also include:

Identification Identification refers to a way to identify and locate the token in the document where it appears, with a reference to page and line numbers for instance. The location may also help to situate the token in the text (with a reference system, such as canonical citations for classical texts mentioned above). A unique identifier could also serve to link the collation result to the transcription, where other properties of the token are encoded and could be retrieved. The Collation Editor includes properties such as *index*, *siglum*, *verse* and *reading* in order to provide identification for each token.

Markup XML transcriptions of the witnesses are often used within collation programmes. Since a lot of valuable information is already encoded in the transcrip-

¹³The collation Editor is a tool produced by The Institute for Textual Scholarship and Electronic Editing (ITSEE) at the University of Birmingham. It is an open source tool available on Github: https://github.com/itsee-birmingham/collation_editor (Accessed February 1, 2017).

4.4. Comparing Tokens in Different Contexts

tions, including layout information, several projects have decided to keep the markup in the token properties. It could be exploited during the collation process: for instance a word marked as bold could be considered as different from the same word in italics. This approach was adopted for instance by the editors of Willem Frederik Hermans, a Dutch writer¹⁴. The inclusion of markup could also serve to display and visualise tokens with more precision. The Beckett Digital Manuscripts project, for instance, displays additions and deletions thanks to this markup property¹⁵.

Facsimile A reference to a digital image, for instance in the form of a link, could be helpful to visualise the original reading in context and assess the transcription accuracy (see Section 8.2.2).

Linguistic properties Those properties could be expressed with a standardised format of detailed linguistic annotation, such as part-of-speech, and morphology. Although Crane (2014) argues that morpho-syntactic analysis is one major feature of a digital edition, Monella (2014a, 184) recognises that the additional workload may be an issue for the encoder. The use of semi-automatic annotation methods is a solution to explore in further research. Smith and Lindeborg (2016) propose to use a ‘dictionary form’ to recognise identical lexical readings, and metrical units to compare the rhythm of Iliadic verses. The use of lemma, synonyms and part-of-speech tagging is also planned to be implemented in collation with the tool iAligner (Yousef and Palladino 2016).

Lacunae If lacunae are represented as tokens, a description of the lacuna’s length and reason (such as damage, or missing page) could be added. In the Collation Editor, lacunae are not represented as tokens, but are included in the properties of the preceding token: *Gap_after*, a boolean variable set to true, records the presence of lacuna after a given token. Another property, *Gap_detail*, gives information about the length of the lacuna.

4.4 Comparing Tokens in Different Contexts

So far, I have attempted to define a reading, and to provide a digital representation that can help to satisfy the needs of various scholars. In this last section, I would

¹⁴Bleeker (2017, 122) describes their use of CollateXJSON format to store information about the layout of tokens, such as ‘italic’ or ‘bold’. The digital edition is available here: <http://www.wfhermansvloedigewerken.nl/?lang=en> (Accessed July 31, 2017).

¹⁵See the update from September 17, 2014 here: <http://www.beckettarchive.org/news.jsp> (Accessed October 31, 2016).

4.4. Comparing Tokens in Different Contexts

like to explore how the properties **t** and **n** can be used to select the relevant variants according to the edition, its cultural context, its tradition, or its audience.

As described above in Section 4.2, tokens can be compared to find variant readings according to a specific context. Three possible situations were taken into account: (a) every difference is a variant, (b) only lexical differences are variants, and (c) only non-lexical differences, such as spelling, are variants (see above p. 150). In these situations, the properties **t** and **n** of tokens already make it possible to distinguish variants for these three different contexts. Let us consider again the example of a collated sentence in figure 4.1. The readings would be transformed into tokens with **t** properties, and the two readings *sunne* and *worlde* would be given normalised forms **n** (respectively *sun* and *world*).

In the first situation, all differences are variant readings. Therefore, in each column, the tokens are compared on the basis of their property **t**: in the first column, all tokens have the same property **t**, *The*, and thus there is no variant. In the second column, the absence of *bright* in witness B is a variant, and so on. When each reading has been examined, the following figure 4.5 highlights every variant.

A	The	bright	star	shines	upon	the	world.
B	The		sun	shines	upon	the	world.
C	The	bright	stars	shone	upon	the	world.
D	The	bright	sunne	shines	upon	the	worlde.

Figure 4.5: Every difference is a variant.

In the second scenario, differences which do not change the meaning of the sentence are considered irrelevant. In order to find the relevant variant readings, the tokens must then be compared on their normalised property **n**, so that differences appearing in property **t** are ignored. In our example, this means that the last column will not show a variant, because witnesses C and D will have the word *you* as a normalised form: when comparing this normalised form to the tokens in witnesses A and B, there will be no difference. The two tokens show a spelling difference (in property **t**) but are in fact considered the same reading (they share the same property **n**). Figure 4.6 shows the locations of non-orthographic variants.

Finally, orthographic variants can be isolated when searching for tokens which share the same normalised form **n**, but not the same original form **t**. In our example, there are thus two columns which contain an orthographic variant (see figure 4.7).

4.4. Comparing Tokens in Different Contexts

A	The	bright	star	shines	upon	the	world.
B	The		sun	shines	upon	the	world.
C	The	bright	stars	shone	upon	the	world.
D	The	bright	sunne	shines	upon	the	worlde.

Figure 4.6: Non-orthographic differences are variants.

The table could then be reduced to a list of orthographic variants, so that repeated orthographic variation is shown only once:

- sun (B) - sunne (D)
- world (ABC) - worlde (D)

A	The	bright	star	shines	upon	the	world.
B	The		sun	shines	upon	the	world.
C	The	bright	stars	shone	upon	the	world.
D	The	bright	sunne	shines	upon	the	worlde.

Figure 4.7: Orthographic differences are variants.

These three simple examples are of course generalisations: in reality, the principles of collation may be far more complex. For example, spelling differences may be ignored, except in proper nouns (Love 1984, 52). In some cases, it may be difficult to distinguish between a spelling difference or a morphological one. Some variants could also combine several aspects: for instance *felix* (adjective) and *foeliciter* (adverb) would be a lexical variant, but show also a spelling difference. The search for spelling variants as shown here is limited to spelling variants aligned in a collation table, but spelling differences could also be present elsewhere such as in the same witness (if the scribe has been inconsistent).

In addition, different readers may give diverse interpretations for certain words or sentences, as it is the case with annotated treebanks (Bamman and Crane 2011). The uncertainty and multiple interpretations thus needs to be represented. However, if the tokens contain more detailed information, it may help to bring more precision when deciding which readings are variant readings.

4.5 Conclusion

Different versions of a text are characterised in part by their variant readings. To represent variant readings in digital format, it may then be helpful to precisely define and formalise the concept. What is a variant reading, however, is highly dependent on the tradition in question (oral, medieval, early printing, etc.) and on the scholarly point of view on the text (stemmatics, linguistics, and so on). As a result, the set of differences present in a textual tradition are not all considered significant in every situation: variant readings are only a subset of all the differences, and different contexts call for different sets of variant readings, as we have seen in the last section.

A first step in formalising variant readings may be to model and formalise *readings*, in such a way that later, those readings can be compared efficiently in order to define which readings are variant readings in a given context. The definitions of the term reading thus provided a series of features which could be used to create a model of a reading. Those features raised a few issues regarding their content, their position in the text as well as in the document, and their relationship between different levels of reading (from characters to words, sentences, and so on). The translation of *readings* to *tokens*, with CollateX input format, showed how the use of a simple normalised form could allow to find different sets of variants according to three different contexts. However, as more information is encoded within the features of readings, it could be possible to define variant readings even more precisely. The interest of the model is to represent readings independently of their function in textual criticism, but with enough information so as to decide when a difference becomes a variant. Considering other sorts of content, such as mathematical diagrams, images or music, the model could be extended in the future to allow for collation and comparison of non-textual content as well.

Conclusion

As underlined in Chapter 1, collation is not a mechanical and tedious preliminary to textual criticism, but a genuinely scholarly activity that involves critical decisions. The adoption of automated collation only increases the number of critical decisions that must be consciously taken.

Chapter 1 summarised the definition, purpose and issues of manual collation, while Chapter 2 introduced automated collation and offered an extensive overview of the whole process and its relationship to manual collation. Relying on the recent development of tool criticism, I have prepared a series of criteria which may be useful to compare existing collation tools, as well as understand and assess new tools.

A more tangible result of this theoretical research is that I have added over a hundred definitions to the *Lexicon of Scholarly Editing* (Dillen and Van Hulle 2013–). These definitions have in turn enabled more research: for instance, a paper presented at the ESTS-DiXiT conference in Antwerp (Andrews 2017) examines the definitions of the term ‘collation’ in the *Lexicon*, most of which were added by myself while working on the first part of the thesis.

Chapters 3 and 4 focused on the decisions that must be made before starting a transcription and automated collation: scholars have to clearly identify the witnesses and the textual features that need to be collated (such as scribal corrections, abbreviations, etc.), and whether plain text will be sufficient or if aspects of the context also need to be incorporated into the tokens (such as paratextual elements or editorial comments). The concept of variant readings is central to automated collation, but as noted in Chapter 4, it is difficult to provide a model that is valid for every kind of textual tradition. The solution was to propose a model of readings that allow for a flexible definition of variant, according to the context, and that could be adapted for instance to ignore or include orthographic variation.

PART II

PRACTICE

Introduction

THE purpose of the dissertation is to study the application of automated collation tools to the creation of a digital critical edition in the context of Classical literature. While the first part of this thesis was devoted to theoretical issues, the second part will deal with the practical aspects of automated collation. To this end, a text was chosen as a test case and was collated with different automated collation tools.

In this part of the thesis, I will start in Chapter 5 by describing the material of the textual tradition that was selected as a test case, the *Declamations* of Calpurnius Flaccus. I will review the extant manuscripts and the editors of the text, as well as how the various witnesses are related into a stemma. This introductory chapter will also present the reasons for working with Calpurnius, and describe the workflow that was applied to the text of Calpurnius, from transcription to the analysis of the collation results. Each of the remaining chapters will focus on a particular step in this workflow:

- **Transcription:** Chapter 6 will concentrate on the practical aspects of transcription, such as the choice of a transcription platform, and the encoding of the witnesses with the XML TEI standard.
- **Collation:** Chapter 7 details the automated collation of the *Declamations* with three different tools. This chapter compares the various issues and advantages related to each tool, relying especially on the four criteria for collation tool criticism identified in Chapter 2 (interface, data preparation, collation process and analysis of the results). The outcome of this comparison is the selection of one tool and one collation output, which was adopted for further exploration of visualisations that can help scholars to make use of automated collation in the context of producing a digital critical edition.
- **Visualisation:** in Chapter 8, I will consider existing visualisations, and pro-

pose enhancements to the collation table visualisation. I will also describe a tool that I have created in order to analyse the results of automated collation while applying the traditional Lachmann method for establishing the relationships between witnesses.

The purpose of this part is to put automated collation into practice, considering issues raised in the part on Theory: the division of manuscripts into witnesses, or the transcription and representation of readings as tokens in digital format. How are these issues dealt with in a practical example? What affects the results from the collation tools? The purpose is also to survey what can tools do to collate and visualise the results, and identify which tool is best suited to the task as well as eventual needs from editors to which the current tools would not respond.

The practice of automated collation dates back to over fifty years ago, as we have seen in Chapter 2, and digital scholarly editions have been discussed since the early days of hypertext technology (Franzini, Terras, and Mahony 2016). How many digital critical editions have in fact been produced with the help of automated collation? Catalogues of digital editions such as the ones created by Sahle (2017) and Franzini, Terras, and Mahony (2016) can give us approximative numbers.

For instance, a search through catalogues of digital editions reveals only two editions for Antiquity: the edition of the Greek New Testament, Digital Nestle-Aland, and the Hebrew text of the Digital Mishnah (Franzini, Terras, and Mahony 2016; Sahle 2017). To these can be added a series of seven projects using the Virtual Manuscript Room (VMR) framework¹⁶. It can be noted that in the fields of Classics, digital critical editions are less numerous than in other fields (Dahlström 2000; Monella 2014b; Franzini, Terras, and Mahony 2016).

For the Middle Ages, the *Catalog of Digital Editions* (Franzini, Terras, and Mahony 2016) returns the editions of the *Codex Suprasliensis* (Birnbaum 2012), the *Chronicles* of Matthew of Edessa (Andrews 2009), and Dante's *Commedia* (Shaw 2010). To these can be added the other texts published by Scholarly Digital Editions, a publishing company founded by Peter Robinson and Barbara Bordalejo, as well as the edition of Petrus Alfonsus (Cardelle de Hartmann, Senekovic, and Ziegler 2014) for which part of the text was collated automatically. Although there are more examples for medieval traditions than for classical texts from Antiquity, the digital editions that make use of automated collation represent only a small percentage of

¹⁶<http://vmrcr.org/> (Accessed January 25, 2018).

the total. It is worth noting that most of the examples here make use of CollateX, or its predecessor *Collate* in the case of publications by Scholarly Digital Editions.

To understand why, several explanations can be advanced. The first is to call into question the usefulness of automated collation. For instance, it may be that full transcriptions of witnesses is not adapted to Classical traditions. Instead of starting new critical editions from scratch, Damon (2016) suggests that making use of the work carried out by previous scholars is a more sustainable approach. To digitise old critical edition with OCR technology is the approach of the Open Greek and Latin project, which has resulted for instance in editions of fragmentary texts such as the Digital Athenaeus (Berti, Almas, and Crane 2016).

A lack of user-friendly tools or guidelines may also be a cause, as well as a lack of consensus regarding digital editions (see the General Introduction p. 15). There is no lack of tools to perform automated collation, and new tools are constantly developed. However there is still little information for users to compare and choose amongst all the available options.

The case study of Calpurnius Flaccus is an opportunity to provide a set of guidelines for the application of automated collation and to implement the framework for tool criticism. Through the creation of a visualisation tool for textual editors, this dissertation also demonstrates how automated collation and its results in a flexible digital format can help in the creation of a digital critical edition.

Test Case: the *Declamations* of Calpurnius Flaccus

5

THIS chapter serves as an introduction to the second part of the dissertation: it presents the *Declamations* of Calpurnius Flaccus that was used as a test case, as well as the method which was applied to the text in order to study the potential of automated collation for Classical literary texts.

The first section of this chapter is dedicated to the tradition of the *Declamations* of Calpurnius Flaccus. The section will discuss the text and what is known about the author, the witnesses, and the stemma. It will include a description of the five known manuscripts which bear a version of the *Declamations* as well as the *editio princeps*, and discuss how they are related to each other. The second section will describe the methodology that was adopted for this research: transcription, automated collation, and visualisation of the results.

5.1 The Tradition of Calpurnius Flaccus

5.1.1 *The Text and the Author*

The text that was chosen as case study during this dissertation is the *Declamations* of Calpurnius Flaccus. Declamations were exercises that were taught at the end of the school curriculum in the Greek and Roman world. Declamation was originally a Greek practice, which is attested in papyri dating to the third century BC (Russell 1983, 4), and was adopted in the Roman world as well. Quintilian¹ wrote about the entire education of a good orator in the *Institutiones Oratoriae*, a textbook in twelve volumes describing the two stages of Roman school curriculum: first the students learnt to speak, read and write properly with the *grammaticus* (see for instance Quint. *Inst.* 1. 4. 1 and 1. 9. 1-2), and later they studied with a *rhetor* the principles of rhetoric. The declamation was the last and most difficult exercise practised at

¹Marcus Fabius Quintilianus, Roman rhetorician from the first century (c. 35 - c. 100 AD).

5.1. The Tradition of Calpurnius Flaccus

the rhetor's school (Quint. *Inst.* 2. 4). Declamations could be of the type either deliberative or judiciary. In the deliberative *suasoriae*, the speaker had to give an exhortation to a fictive or historical character, while the *controversiae* were legal speeches for fictitious court cases (Sussman 2013). Given a situation of conflict (the theme) and a set of laws, the students had to play the part of a lawyer and learn to defend both parties. Here is an example of such a controversial situation in Calpurnius, *Decl.* 4, titled 'the parricide who sues for imprisonment':

Law: A person convicted of parricide shall be kept in custody for a year.

Theme: A certain man was convicted of parricide during the ascendancy of a stepmother in the household. His father wants to keep him in custody at home. The son sues to be kept in the state prison (translation by Sussman 1994, 31).

This situation is followed first by the discourse in favour of the son, and then by a second discourse defending the father's point of view. The declaimer had to be able to interpret the same situation from different point of views and different arguments: the point of view adopted is a *color* (see Montefusco 2003), constructed from the law, the elements present in the theme and the personality of the characters involved. The characters portrayed in declamations are everyday members of society: fathers and sons, stepmothers, young women and rapists, rich and poor enemies, deserters and war heroes, tyrants, pirates and so on. Each of these generic characters could have a positive personality attached to them, such as the hero and tyrannicide, or a negative one (for instance stepmother, tyrant, deserter), and some characters were neutral (father, son). The purpose of the declamation exercise is then to build a convincing plea with the help of witty traits, the *sententiae*. Declamations were practised in the school curriculum throughout the Roman Empire (Kaster 2001), but they also evolved in a literary genre of its own, with performances from well-known rhetors for the public entertainment (Sussman 1994; Bloomer 2007; van Mal-Maeder 2007). Four corpora of Roman declamations have been transmitted to us, mostly in the form of excerpts:

- Fragments of 74 *controversiae* in ten books and 7 *suasoriae* in one book, gathered by Seneca the Elder for his sons.
- 19 *declamationes maiores*, falsely ascribed to Quintilian. These declamations are the only ones which preserve entire discourses, hence the qualification of 'major'.
- 145 *declamationes minores*, surviving from an original corpus of 388, also ascribed to Quintilian.

5.1. The Tradition of Calpurnius Flaccus

- 53 excerpts of declamations by Calpurnius Flaccus.

The *Declamations* of Calpurnius Flaccus is a collection of fifty-three *controversiae* extracts: besides the titles, laws and themes, we do not possess complete speeches but only the most noteworthy *sententiae* of the author as selected by the scribe who excerpted the text.

Very little is known about the author Calpurnius Flaccus. He has been sometimes identified as a certain M. Calpurnius, *consul suffectus* in 96 AD, or as the recipient of a letter by Pliny the Younger, *Epistula* 5.2 (see Sussman 1994, 6-7). Both men lived in the first century AD, but stylistic elements of Calpurnius suggest that the *Declamations* were rather written in the second century (Sussman 1994, 6). A posthumous publication by Håkanson in particular seems to indicate a dating to the second half of the 2nd century, based on the analysis of sentence rhythms — an important aspect of rhetorical art (Håkanson 2014, 130)².

The *incipits* of several manuscripts introduce Calpurnius as one of the ten minor rhetors. Those ten rhetors could also have included Antonius Julianus who is associated with the *Minor Declamations*, Seneca and Calpurnius in the correspondence of the Italian Humanist Campano (see Section 5.1.2.1 below). However, Huelsenbeck (2016) is not convinced by the existence of a corpus of ten minor rhetors, but is inclined to think that this number refers rather to the ten books of *Declamations* by Seneca, each introduced by a preface that presents a different declaimer (Huelsenbeck 2016, 366, note 41). Pithoeus also made the link between the mention of ten rhetors in Calpurnius with the ten books of Seneca (see p. 175).

5.1.2 The Manuscripts and the First Edition

The text is transmitted by five manuscripts. The older surviving manuscript is codex Montepessulanus H 126 (A). A lost manuscript (X) appears in the correspondence of Humanist scholars: this lost manuscript is likely the source of two manuscripts from the fifteenth century, codex Monacensis Latinus 309 (B) and codex Chigianus Latinus H VIII 261 (C). The tradition is completed by two other manuscripts from the sixteenth century, codex Monacensis Latinus 316 (M) and Bernensis Latinus 149 (N). Also in the sixteenth century, the French scholar Pierre Pithou [Petrus Pithoeus]³ published the *editio princeps* in 1580, which was reprinted fourteen

²See for instance Cic. *Orat.* 207-226, or Quint. *Inst.* 9. 4. 60-111 for ancient sources about the importance of rhythm in rhetoric.

³French lawyer, historian, humanist and scholar (1539-1596).

5.1. The Tradition of Calpurnius Flaccus

years later in 1594. The following sections provide a brief description of each manuscript and of the *editio princeps*. References to particular locations in the manuscripts are quoted with a folio number, followed by a line number (e.g., f. 15v:9).

5.1.2.1 A (Montepessulanus H 126)

Manuscript A is kept in the *Bibliothèque Universitaire de Médecine* in Montpellier, France. The Codex Montepessulanus is written in a caroline minuscule script on parchment, on a total of 116 folia (185 x 245 mm). It is considered our best witness, although it is severely damaged. Manuscript A is incomplete both at the beginning and at the end, and the text is of the first and last folio is darkened. The text of Calpurnius, in particular, has suffered from these damages: it stops after only one folio (f. 116) at the sixth declamation out of fifty-three. Six to eight folia are estimated to be missing at the end of the codex. The last folio is missing the bottom right corner as well and the ink is faded. In an early attempt to make it more readable, a chemical was spread on the last folio and stained it so that it is now impossible to read in several places.

The manuscript contains the *Declamationes Minores* 244 to 388 (ff. 1-88), followed by the declamations of Seneca the Elder (ff. 89-115) and finally the beginning of Calpurnius Flaccus. The text is occasionally corrected by a second hand (A2), different from the first hand of the scribe who wrote the major part of the text.

Dating from the late ninth century (Cortesi 1994) or the tenth century (Håkanson 1978), manuscript A is the earliest known witness of Calpurnius Flaccus. The manuscript was likely written by Hincmar, archbishop of Reims since 845, and annotated by Heiric of Auxerre (van Büren, private correspondence). It remained unknown until the sixteenth century when Pithoeus received it from Claude Fauchet and published the *editio princeps* of Calpurnius in 1580 (see Section 5.1.2.6 below). The name *P. Pithou* is still visible in the top right corner of folio 116v. In addition, Lehnert (1903) describes two notes which can be read in manuscript A: the first note is dated to the thirteenth century and reads '*Liber Sancti Theodori auferenti anathema sit*'. *Sancti Theodori* likely indicates the abbaye of Saint-Thierry near Reims as the place where the manuscript originated or was once kept. The words *anathema sit* are added by another hand in rasura (see Lehnert 1903, V). Another note in a more recent hand reads *ex libris collegii oratorii Trecensis*, from the books of the College of the Oratorium in Troyes. When Pithoeus died, his library passed to his brother François, who in turn willed the library to the city of Troyes. The College

5.1. The Tradition of Calpurnius Flaccus

of the Oratorium was subsequently created and it inherited the Pithoeus library⁴.

A note describes a lost manuscript similar to Montepessulanus H 126 in a letter from Giovanni Antonio Campano⁵ to Francesco Todeschini Piccolomini, then Cardinal of Siena. The letter, *Censura in Quintiliani Declamationes*, was published by Michele Ferno in 1495 in Venice, along with the other writings of Campano⁶. The date of the letter is estimated to be around 1470, when Campano was working on an edition of Quintilian's *Institutiones Oratoriae*, of which he published the *editio princeps* in 1470. At that time, Piccolomini was sent a manuscript found in Germany containing the *Minor Declamations* ascribed to Quintilian, followed by Seneca and Calpurnius' *Declamations*, and finally excerpts from Antonius Julianus. The presence of Antonius Julianus, a rhetor known to Aulus Gellius, would suggest a similar date for Calpurnius Flaccus, according to Winterbottom (1984, XXI, note 10)⁷.

Here is the extract from the letter, with the complete observations of Campano on this unknown manuscript, and the passage concerning Calpurnius has been highlighted in bold. *Omnia Campani Opera* (1495), *Censura in Quintiliani Declamationes*, ff. LXIIIv-LXIIIr, corrected according to Mencke 1707⁸.

⁴See the notice of the Catalogue Collectif de France (CCFr) about the 'collège de L'Oratoire': http://ccfr.bnf.fr/portailccfr/jsp/index_view_direct_anonymous.jsp?record=rnbcdfonds:FONDS:3060 (Accessed August 18, 2017).

⁵Italian humanist (1429-1477), see his complete biography here: [http://www.treccani.it/enciclopedia/giovanni-antonio-campano_\(Dizionario-Biografico\)/](http://www.treccani.it/enciclopedia/giovanni-antonio-campano_(Dizionario-Biografico)/) (Accessed February 20, 2016). He is called 'Giannatori Campano' by (Sussman 1994, 19, note 64). Since I can find no record of a scholar named Giannatori Campano, I take this to be an error. Giovanni Antonio is sometimes contracted to Giannantonio, which could then have been misspelled by Sussman.

⁶(Lemaire 1825, 9) calls this letter *Campani Praefatio ad Cardinalem Senensem. Censura in Quintiliani Declamationes* and dates it from 1470, as well as another letter from Campano to the Cardinal of Siena, *Ciceronis et Quintiliani Comparatio*. Francesco Todeschini Piccolomini was named a legate of the pope in Germany in 1471, and received the manuscript shortly after according to Sabbadini (1905, 142). In any case, the letter was written before Campano's death in 1477.

⁷Gellius (123 - c. 165 AD) refers to Antonius Julianus in Chapter 15 of the *Noctes Atticae* (Gel. 15.1).

⁸Both pages are available on the Gallica website: <http://gallica.bnf.fr/ark:/12148/bpt6k603528/f128.item> (LXIIIv) and <http://gallica.bnf.fr/ark:/12148/bpt6k603528/f129.item> (LXIIIr) (Accessed February 20, 2016).

5.1. The Tradition of Calpurnius Flaccus

Centum XXXVI Quintiliani declamationes ad te nuper e Germania missas Quintiliani esse sic arbitror ut alterius et temporis et studii putem quam undeviginti quae circumferuntur illius nomine.

5 ...

coniecturam facere ex ipso codice licet: primum quod trecentas sexaginta fuisse ostenditur: quem numerum nescio an alius quam Quintilianus impleverit. Deinde quoniam Quintiliano attribuantur: nec multo post Quintiliani tempora exscriptum tam vetustum Codicem puto: ut errari tam recenti memoria vix potuerit. Etiam quod ante Seneca multos quidem declamasse: verum nullius extare declamationes constat: tum quod subsequuntur in Codice declamationum Senecae decem libri: ut dubitari non possit: nec Senecae quidem esse: quamquam illud quoque accedit quod abest Seneca longissime a tantapuritate. Secundum has Calpurnii Flacci excerptae quaedam paucae perbrevesque sententiae quem ego inter declamatores laudatum nescio an unquam legerim. Argutulus tamen vult videri: et nisi quod brevior est Senecae persimilis: post hunc finis excerptarum suscribitur literis maiusculis: et Anthonii Iuliani nescio cuius ac mox et extemporaneae Quintiliani promittuntur: ut facile appareat subsecuturas fuisse extemporaneas.

4 *circumferuntur*] circumferunt, *Ferno*.
19 *Calpurnii*] Calpurnii, *F*
19–20 *excerptae*] exceptae, *F*
27–28 *extemporaneas*] extemporaneus, *F*

The one hundred and thirty-six Quintilian declamations recently sent to you from Germany are, I believe, from Quintilian, just as I think they are from another time and another 'intellectual activity' than the nineteen [declamations] which circulate under his name.

... [Here follows considerations on the text of Quintilian's declamations.]

Here is what we may infer from this manuscript: first that it appears that there were 360 [declamations]: I do not know if someone other than Quintilian contributed to this total. Next, since they are attributed to Quintilian, I think that such an old manuscript was copied not long after Quintilian's time: working from so fresh a memory, the scribe is unlikely to have made many mistakes. Also that it is apparent that many wrote declamations before Seneca, but no declamations are extant from any of these earlier authors. Then that ten books of Seneca's declamations are following in the manuscript: there can be no doubt about Seneca's authorship here: although admittedly there are elements which are far removed from the great purity of Seneca's usual style. **After those come a small number of very short sententiae selected from Calpurnius Flaccus, whom I don't know if I have ever read praised among the declamators. Nevertheless he presents himself as a clever writer, and he is very similar to Seneca, except shorter. After his work, 'finis excerptarum' is inscribed in capital letters.** The 'Improvisations' of Quintilian and someone called Antonius Julianus are then promised: so it is clear enough that these 'improvisations' originally followed at this point.

The name *Calpurnii* in Campano's letter is corrected to *Calpurnii* by Mencke, and all subsequent editors have assumed without hesitation that this referred to Calpurnius Flaccus. The incipit of manuscript A is shown in figure 5.1, and the name of Calpurnius has a ligature between P and V. In addition it must be noted that in manuscript M, the name of Calpurnius was corrected to 'Calphurnius' by a later hand who added the letter h above the line (see Section 5.1.2.4 below).

The similarity of content between Campano's description of the lost manuscript (X) and manuscript A suggests that they depended on a common archetype. The lost

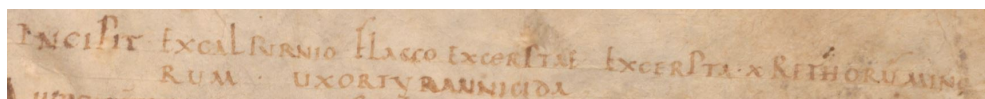


Figure 5.1: The incipit of manuscript A, f. 116r (*Incipit ex Calpurnio Flacco excerptae, excerpta X rethorum minorum*).

5.1. The Tradition of Calpurnius Flaccus

manuscript was likely the source of two fifteenth-century manuscripts, B and C which both have Pseudo-Quintilian's *Minor Declamations* from number 252 to 388, preceding the text of Calpurnius Flaccus. This makes a total of 137 declamations, although declamation 253 is the first to be numbered: the text starts in the middle of declamation 252 (*excutiamus*), which may explain Campano's count of one hundred and thirty-six. According to Sussman, C is directly copied from the lost manuscript and B through one intermediary (Sussman 1994, 19).

5.1.2.2 C (Vaticanus Latinus Chigianus H VIII 261)

The second most valuable witness is Chigianus H VIII 261, kept in the Vatican Library. Manuscript C is written on paper. It contains the *Minor Declamations* ascribed to Quintilian (ff. 1-81v) and is the only manuscript to transmit all fifty-three declamations of Calpurnius Flaccus (ff. 81v-90). The incipit is *Incipiunt ex Calpurnio Flacco excerptę, excerpta x Rhetorum minorum*, and the explicit is *Explicitę ex Calpurnio Flacco excerptę*. Little is known about the manuscript's provenance and history. Two watermarks refer to the years 1465 and 1480 (Cortesi 1994, 84).

There are two portions of text present in manuscript C which are missing in the other manuscripts. First, the text at the end of declamation 31 and beginning of declamation 32 is absent from other manuscripts. As a result, declamations 31 and 32 are combined into only one declamation in B, M and N, as well as in Pithoeus' edition. Second, declamation 45 is altogether absent from B, M, N and Pithoeus. The two lacunae explain why C has a total of fifty-three declamations, as opposed to fifty-one in the other witnesses. Manuscript C is more complete than the other witnesses, which places it closer to the archetype in the stemma.

Abbreviations are numerous, but corrections on the other hand are relatively few, with 28 occurrences. The corrector of C has likely made conjectures of his own according to Håkanson (1973, IV). In a few places, the copyist of C has left dots to mark a place where text is missing (see figure 5.2).

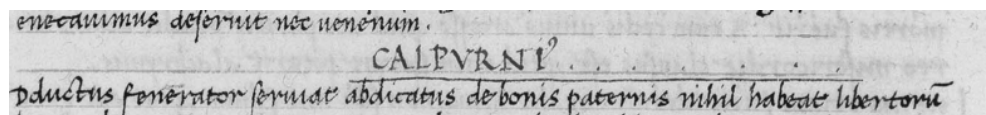
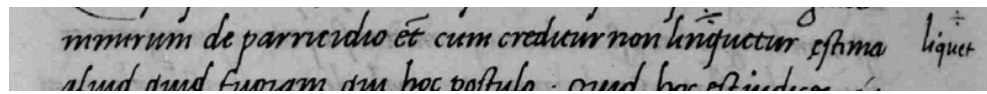
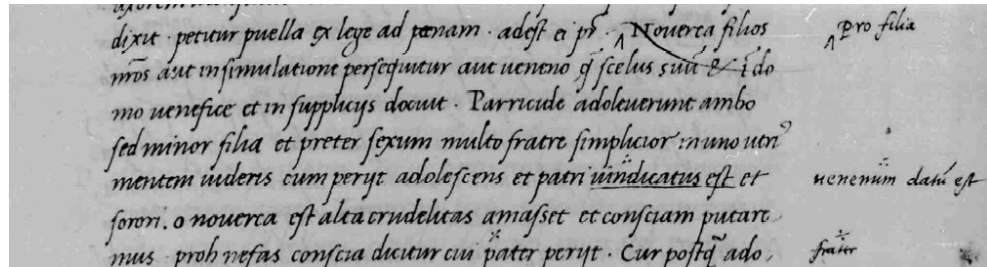


Figure 5.2: Manuscript C, folio 84r, line 38.

5.1. The Tradition of Calpurnius Flaccus

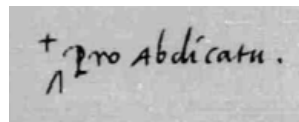


(a) Manuscript B, folio 148v, line 11.

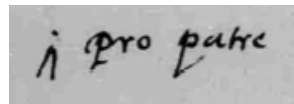


(b) Manuscript B, folio 151r, lines 6-10.

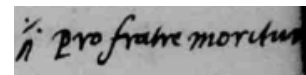
Figure 5.3: Additions in the margin of B, similar to the ones found in N.



(a) Folio 151v, line 16.



(b) Folio 149v, line 21.



(c) Folio 153r, line 23.

Figure 5.4: Additions in manuscript B.

5.1.2.3 B (BSB Clm 309)

Closely related to manuscript C is the Codex Latinus Monacensis 309 held in the *Bayerische Staatsbibliothek* in Munich, Germany. The manuscript is written on paper and contains the *Minor Declamations* 252 to 388 (ff. 1r-146v), Calpurnius Flaccus (ff. 147r-160v) and the Twelve Latin Panegyrics (ff. 161r-324). The incipit of the text is *Incipiunt ex Calpurnio Flacco excerptę. Excerpta Decem Rhetorum minorum*, and the explicit is *Explicitę ex Calpurnio Flacco excerptę*. Manuscript B is dated from the end of the fifteenth century, but was written before it was used as a witness by Taddeo Ugoletti in Parma, for the *editio princeps* of Pseudo-Quintilian's *Minor Declamations* published in 1494 (Cortesi 1994). The text in italic strongly suggests an Italian origin.

Manuscript B is the most heavily corrected, with close to 200 corrections by the first or second hand. The corrections by B2 seem to indicate that they have been added from N according to Håkanson (1978, XII). These annotations are from a more recent hand. The signs for additions in the margins resemble closely the one found in N (see Section 5.1.2.5). Here are some examples of additions in B (see figure 5.3). In some cases, the caret ^ sign for an addition is topped by another sign such as a cross or a dot (see figure 5.4). Manuscript B is also the most abbreviated one, since there are over five hundred abbreviated words.

5.1. The Tradition of Calpurnius Flaccus

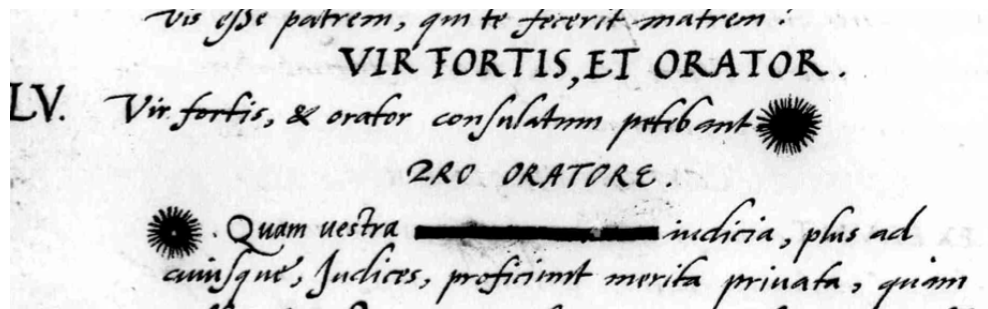


Figure 5.5: Sun-like decorations and a deletion in M, folio 18v, lines 20-25.

5.1.2.4 M (BSB Clm 316)

The codex Monacensis 316 (manuscript M), also in the *Bayerische Staatsbibliothek*, contains only the *Declamations* of Calpurnius (ff. 1-20). A brief description in the *Catalogus codicum latinorum Bibliothecae Regiae Monacensis* (1868, 80) lists only 19 folia. The reason is that the foliation on folio six is missing, and all the following folia have thus been misnumbered. The incipit of the text is *Incipiunt ex Calp^hurnio Flacco excerptae*, with the letter 'h' added later above the name Calpurnio, and the explicit is *Finis. Explicitae ex Calpurnio Flacco excerptae*. The text is written on paper, in an italic script, which could be the sign of an Italian origin. Two bookplates indicate that it was in the library of the Duke of Bavaria Maximilian I, and the first of these bookplates is dated to 1618. There is no other information regarding the history of this manuscript.

The declamations are numbered in manuscript M, and before each discourse there is an indication about whom the declaimer is going to defend or accuse (e.g. *pro oratore*, see figure 5.5). Corrections by the first or second hand are minimal and occur only eighteen times. In some cases, a deleted word is completely blackened and illegible, except for f. 15v:9 where *tenet* can be read beneath the dark ink. In addition, sun-like decorations appear on folio 18v, lines 22 and 24 (see figure 5.5). These decorations could indicate a lacuna or damage of some sort in the copy exemplar, since the text at this point is marked as corrupted by Håkanson (1978, 36).

5.1.2.5 N (codex Bernensis 149)

The codex Bernensis 149 A (manuscript N) is held at the *Burgerbibliothek Bern*, in Switzerland. It contains various alchemical, medical and juridical texts, bound together in one manuscript. Calpurnius Flaccus is item number nine (ff. 244r-259v), and is preceded by the second of the *Major Declamations* ascribed to Quintilian.

5.1. The Tradition of Calpurnius Flaccus

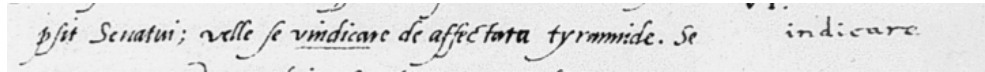


Figure 5.6: Correction by a second hand in manuscript N, folio 246r, line 13.

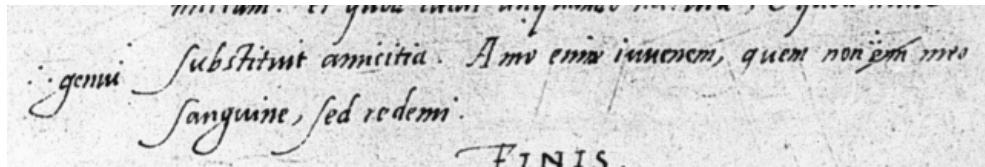


Figure 5.7: Correction likely added by N1 in manuscript N, folio 259v, line 24.

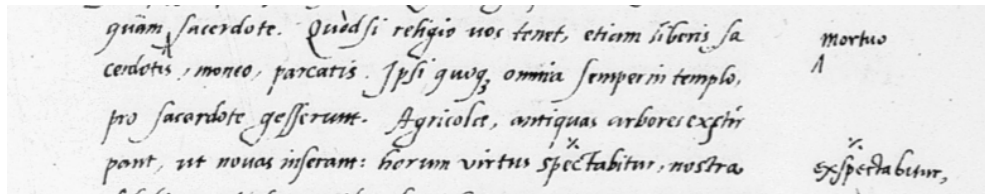


Figure 5.8: Corrections possibly by a third hand in manuscript N, folio 254r, lines 19-22.

The incipit of N is *Incipiunt ex Calpurnio Flacco excerptae*, and the explicit is *Finis*. The text is written on paper, in an italics that could imply an Italian origin. The manuscript is dated to the sixteenth century, but not with more precision. Previously, manuscript N belonged to the library of Jacques Bongars (1554-1612), a French diplomat and Humanist. Through marriage and inheritance, his collection of manuscripts and printed editions came to the Burgerbibliothek of Bern in 1632.

The declamations are also numbered in manuscript N, and the indications *pro* or *contra* are included as well. N contains 32 corrections by either the first or second hand. Håkanson (1978, V) notes that interventions from the second hand are 'rare'. In fact, the only correction that Håkanson (1978, 6) attributes to a second hand N2 in his apparatus is *indicare* in f. 246r:13, a correction which is very clearly by a different hand from the main text (see figure 5.6). The other marginal corrections in N may have been added by the same scribe who copied the text N1 (figure 5.7) or possibly by a third hand, although Håkanson attributes all those corrections to N1 as well (see for instance figure 5.8). Håkanson (1978, IX-X) attributed those corrections in N's margin partly to conjecture and partly to a source outside the known manuscript tradition, which is marked as Y in the stemma p. 177 below.

In addition, it is worth noting that the addition marks are similar to those used in manuscript B, and the hands look similar in both manuscripts. See the shapes of &, b, e, p, s or x, for instance, in figure 5.9.

5.1. The Tradition of Calpurnius Flaccus

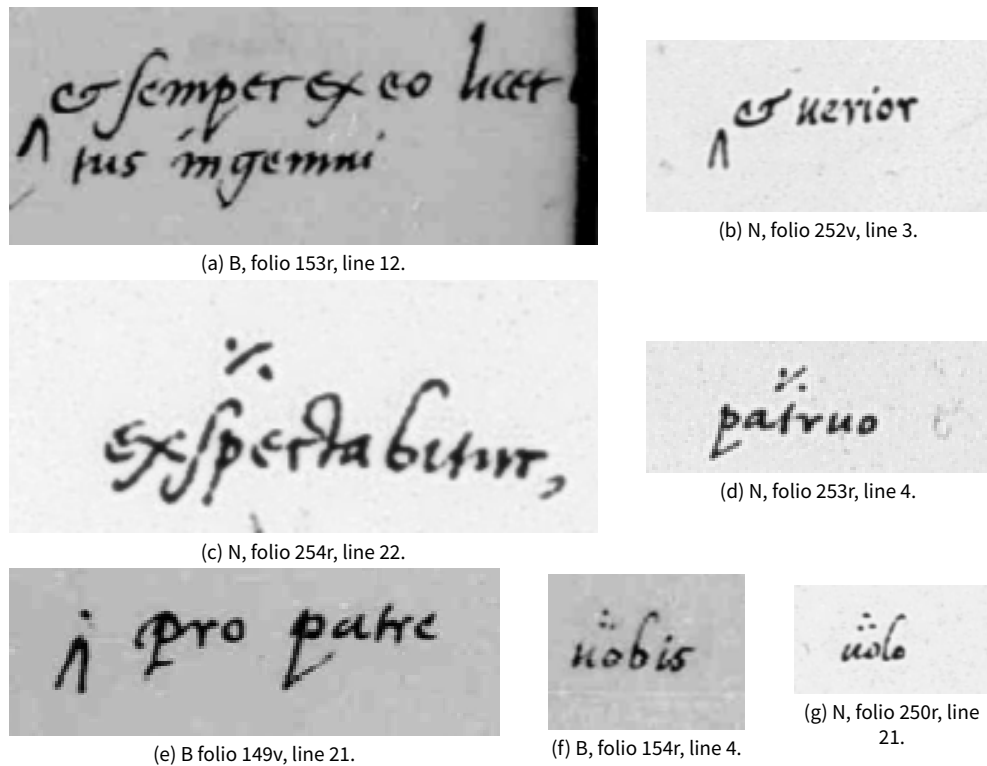


Figure 5.9: Comparison of the corrections appearing in the margin of manuscripts B and N.

Both M and N are often considered very close manuscripts (Håkanson 1978, V; Winterbottom 2017). The two codices have in common declamation numbers, and *pro/contra* indications, as noted above. In addition, they share the presence of diples (») in the left-hand margins. Diples are citation marks which signalled a quotation borrowed from another text, often though not exclusively a biblical text (McGurk 1961). In Greek literature, it had also been used as a generic ‘nota bene’ which may refer to a commentary (Schironi 2012). In the text of Calpurnius, there are 34 passages marked with diples. However, these passages do not seem to represent quotations from the Bible, or any other Latin text that can be searched in the database LLT-A (Library of Latin Texts). They do not distinguish either direct or indirect discourse, that can be isolated from the rest of the text. Instead, the most likely explanation is that a reader marked *sententiae* of personal interest in a manuscript, and that the diples were present in the exemplar of M and N.

All the diples mark the same passages both in M and N, with the exception of two passages absent in N. Despite a few unclear situations, in most cases the diples highlight a coherent and complete *sententia* corroborated by text common to M and N, such as:

5.1. The Tradition of Calpurnius Flaccus

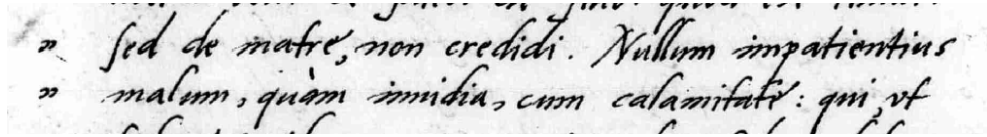


Figure 5.10: Example of diples in the margins of manuscript M, folio 5v, line 18.

- *inquieta res est homo cui iam in deterius nihil superest* (a man to whom nothing worse can happen is a restless thing), in Declamation 6.
- *nullum impatientius malum quam invidia cum calamitate* (there is no ill more unbearable than indignation combined with disaster), in Declamation 10.
- *dolere integre non potest qui urgetur irasci* (one cannot grieve properly, when forced to be angry), in Declamation 11.

The presence of diples, at the same points in the text, reinforces the proximity of the two manuscripts MN in the stemma. It is possible that M and N were copied from the same exemplar.

5.1.2.6 P1594 (editio princeps by Pithoeus)

Pithoeus published the first edition of Calpurnius Flaccus in 1580, in Paris, printed by Robert Estienne in the workshop of Mamert Patisson. The editio princeps was preceded by the *Minor Declamations* and followed by the *Dialogus de Oratoribus*. It was then reprinted in 1594 in Geneva by Jérôme Commelin [Ieronymus Commelinus], a German printer based in Heidelberg. The reprinted edition was used here in the collation because it was already digitised when I started transcribing and more easily available. Nevertheless, the two versions of 1580 and 1594 have been compared and three important differences, other than abbreviations, were noted: in Declamation 21, *possem* was replaced with *posse* Pithoeus (1594, 400:16), in Declamation 23, *es* was replaced by *est* (Pithoeus 1594, 403:9), and in Declamation 34, *medius* was replaced with *melius* (Pithoeus 1594, 409:27). Now the 1580 edition is available online as well⁹.

In his apparatus, Pithoeus explains that he based his text on a very ancient but damaged manuscript which he received from Claude Fauchet¹⁰, and that he sup-

⁹http://digital.onb.ac.at/OnbViewer/viewer.faces?doc=ABO_%2BZ18038450X (Accessed August 10, 2017). The text of Calpurnius starts on page 383 in the book, which is page 417 of 520 in the digital facsimile.

¹⁰French magistrate and historian (1530-1602).

5.1. The Tradition of Calpurnius Flaccus

plemented it with a more recent Italian manuscript. This note comments on the subscription at the end of the Minor Declamations¹¹:

*Sic vetustissimum et optimum exemplar.
S.T.R. quod Claudii Falceti V.C. et doctiss. beneficio habuimus. Sequebantur vero in eodem exemplari declamationum Annaei Senecae libri
5 decem [...].*

*Post has incipiebant EX CALPURNIO FLACCO
EXCERPTAE X. RHETORUM MINORUM, quo ordine superior nostra coniectura de hac inscriptione non nihil confirmatur. Sed nec illud dis-
10 simulare volumus in illo optimo et vetustissimo codice Calpurnii Flacci vix superuisse quartam partem: reliqua nos habuisse ex Italico exemplari non adeo vetusto. Quis autem hic Calpurnius fuerit, alii fortasse dicturi sunt: mihi quidem
15 nondum scire contigit.*

Thus [was] the ancient and excellent manuscript [from Saint-Thierry] that we obtained thanks to Claude Fauchet, a very distinguished and erudite man. In the same exemplar follow ten books of Annaeus Seneca's declamations [...].

After those begin the extracts of Calpurnius Flaccus, of the ten Lesser Rhetors, and as for the position of these extracts in the manuscript, our conjecture on this topic above is somewhat confirmed by this inscription. However, we do not wish to conceal the fact that barely a quarter of Calpurnius Flaccus is recorded in this excellent and ancient manuscript: the rest we have taken from an Italian exemplar which is not as old. Who this Calpurnius was, maybe others will tell: as for myself, I do not happen to know yet.

The abbreviation S.T.R. may be expanded to 'Sancti Theodorici Remensis', the abbey of Saint-Thierry located near the city of Reims in France. Indeed there is a note in manuscript A which mentions Saint-Thierry (see the description of A above). In the dedication, Pithoeus conjectures that the extracts of Calpurnius Flaccus serve as a comparison with Seneca's ten books of Minor Declamations, which are placed just before Calpurnius in the manuscript, and which are each introduced by a portrait of a declaimer.

From this description, manuscript A is clearly identified as the *vetustissimum et optimum exemplar*. First of all, the name of Pithoeus is inscribed in manuscript A (see above). The contents of the manuscript listed by Pithoeus correspond to the contents of manuscript A. The damage described by Pithoeus corresponds as well to the state of the codex Montepessulanus, and a few readings quoted by Pithoeus corroborate the identification, such as *hispaniae* (folio 116r:14), *miraris* (116r:18), *livor* (116r:25), *crede* (116r:32), or the title of declamation four, *parricida carcerem petens* (only the first two words are still visible on folio 116r:37). However, it seems that Pithoeus is misquoting the inscription of A in the note above since he omitted the second *excerpta*, before the *X rhetorum minorum* (see the *incipit* of A in Section 5.1.2.1 above).

The *editio princeps* of Pithoeus is therefore based in part on manuscript A, but since it is damaged and incomplete, Pithoeus had to rely mostly on another manuscript

¹¹The note is available here : http://www.e-rara.ch/gep_g/content/pageview/1099004 (Accessed January 5, 2016).

5.1. The Tradition of Calpurnius Flaccus

which he referred to as the 'Italian exemplar'. Pithoeus does not give much detail about this codex, but the only manuscript that we know for certain to have been in Italy is manuscript C, now held in the Vatican Library. However, both M and N are written in italics, an indication of a potential Italian origin. In addition, Pithoeus' edition shares in common with M and N the numbering of declamations, and the *pro/contra* indications at the beginning of each discourse. Håkanson argues that N was the Italian manuscript (see p. 177 below).

5.1.3 *The Editors and the Stemma*

After Pithoeus, the *Declamations* of Calpurnius Flaccus were edited again several times, and emendations were proposed by various scholars. Some of these old editions are digitised and available online. Here is a list of Calpurnius' editors:

1665 Gronovius.

1698 U. Obrecht.

1720 Burman. Lemaire (1824) has also published an edition with Burman's text, reporting notes from Gronovius, Schulting and other scholars.

1903 Lehnert.

1978 Håkanson, who quotes also emendations by Dessauer and Klotz (see the preface, page XIV).

Håkanson's edition is still the standard reference for Calpurnius Flaccus (Sussman 1994, 22; Winterbottom 2017). Other suggested emendations to the text can be found in articles by Håkanson himself (1972; 1974; 1976), in Jones (1985), Watt (1996), Winterbottom (1999), and Balbo (2012). In addition, two translations have been published in English with a commentary (Sussman 1994) and in French (Aizpurua 2005). Both translations are presented in parallel to a Latin text without critical apparatus, and they rely heavily on Håkanson's edition although with slight differences (Sussman 1994, 21-22; Aizpurua 2005, 25-26). A new critical edition is in preparation for the French series of the *Belles-Lettres* by Andrea Balbo and Catherine Schneider. For more on the editors of Calpurnius Flaccus, an in-depth discussion is provided by Winterbottom (2017).

Lehnert's edition is the first to make use of all known manuscripts (Håkanson 1978, XIV). Lehnert's edition also took advantage of the progress of textual criticism in the

5.1. The Tradition of Calpurnius Flaccus

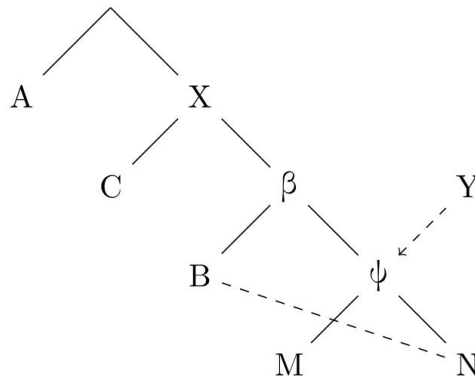


Figure 5.11: Calpurnius Flaccus stemma (Håkanson 1978, X).

nineteenth century, with the adoption of the Lachmannian method: he proposed the first stemma for Calpurnius Flaccus (Lehnert 1903, X). This stemma was then confirmed by Håkanson (1978, V), who did not make major changes to Lehnert's hypothesis. The only notable addition is the hypothetical witness Y, which may have introduced interpolations into the hyparchetype of M and N (see Håkanson's stemma in figure 5.11).

In his preface to the *Declamations*, Håkanson describes in detail the reasoning process behind the stemma construction. He gives a practical example of the application of Lachmann's method to a Latin literary text. Here is a summary of how Håkanson established his stemma. Håkanson shows first that all five manuscripts descend from a common archetype, since they share a few errors (Håkanson 1978, VI). He postulates then that A and X, the lost hyparchetype of BCMN, form two distinct branches of the stemma. However, the text of Calpurnius is too short in manuscript A to prove this point. Instead, Håkanson relies on other texts transmitted in the manuscripts A, B, and C.

Next, Håkanson proceeds to analyse relationships between BCMN: BMN have some errors in common which are absent from C. Therefore, the stemma is divided again in two branches stemming from X, with C on one side and BMN on the other side (Håkanson 1978, VII-VIII). Manuscripts M and N are separated from B by errors that they have in common (Håkanson 1978, VIII), and interpolations which have been introduced in their exemplar by an unknown witness Y (Håkanson 1978, IX). Furthermore, there was an exchange of readings between manuscripts B and N. A few readings from N have been added in the margins of B by a second hand (B2): *vel* (22.4), *remittitur* (24.11) and *in vita liberis* (25.17) (see Håkanson 1978, XII).

5.2. Method: Automated Collation Applied to a Classical Text

Håkanson examines how the *editio princeps* is related to the manuscripts. Håkanson argues for N to be the Italian manuscript mentioned by Pithoeus, because readings unique to N were adopted by Pithoeus:

- 7.19 *inquit erant*
- 19.4 *scio me*
- 20.11 *quacumque*
- 22.4 *vel*
- 22.20 *es*
- 24.11 *dicit* and *remittitur*
- 25.17 *in vita liberis*
- 31.2 *pauper*

Furthermore, Jacques Bongars, the last owner of manuscript N before its acquisition by the Bern Burgerbibliothek, was in close contact with Pithoeus (Banderier 2009, 397). Bongars could have shared the manuscript with Pithoeus. The relationship between Pithoeus' edition and the other manuscripts will be further examined in Chapter 8, which introduces a visualisation tool for the collation results, and shows how the conclusions of Håkanson can be reproduced.

5.2 Method: Automated Collation Applied to a Classical Text

This section describes the methodology that was applied to the tradition of Calpurnius Flaccus in order to study automated collation. The first step was in fact to choose a text for the application of automated collation, and there were several reasons for choosing the *Declamations* of Calpurnius Flaccus as a case study. First, it is a relatively limited manuscript tradition, with only five manuscripts: the transcription of the entire tradition represents a manageable amount of work for this dissertation, while leaving sufficient time to perform collation with various tools and to investigate visualisation options.

A second reason is that the texts of declamations have long been ignored in the context of literary studies, as they were considered as little more than schoolboy work instead of true literary productions (for instance Bloomer 1997). However, since

5.2. Method: Automated Collation Applied to a Classical Text

the 2000s, there has been a growing interest in Roman declamations (Gundersen 2003; van Mal-Maeder 2007; Bernstein 2013; Dinter, Guérin, and Martinho 2017), and a new critical edition of Calpurnius is currently underway.

Finally, the stemma of Calpurnius is rather well defined, and both Lehnert and Håkanson give a detailed account of the process of creating the stemma, with lists of variant readings to support their analysis. For this reason, the text of Calpurnius makes a good candidate to compare the results of traditional versus digital methodology, and to understand how the output of an automated collation tool can help editors.

The various steps, from the transcription, to automated collation and visualisation of the results, will be briefly described here. The next chapters will provide more details for each aspect of the method.

5.2.1 Transcription

As we have seen in Chapter 3, the witnesses need to be transcribed in order to be collated with a digital method. Decisions were made regarding which witnesses to transcribe, and how to represent second hands. I have transcribed all five manuscripts, as well as the *editio princeps* of Pithoeus and the critical edition of Håkanson. Håkanson's text is still the best critical edition available, complete with a comprehensive critical apparatus, and his edition serves as a reference point. The reason to include the edition of Pithoeus is that its relationship with the manuscripts had become a subject of enquiry during this dissertation and I wanted to compare his text to the other manuscripts. The other editions either are not critical editions complete with an apparatus (Sussman 1994; Aizpurua 2005), or are outdated (Lehnert 1903) and therefore do not present the text as accepted by the scholarly community.

It is worth noting the particularities of Calpurnius Flaccus: this is a small manuscript tradition, with few transpositions limited to at most three or four words, and a limited amount of orthographic variation (certainly less than in a medieval tradition, for instance). There is only one fragmentary witness. For each manuscript, the text of the first hand and the corrections by a second hand were represented as two different witnesses. However, as we have seen above, there are some cases when it could be argued that a third hand may have altered the text. I did not create separate witnesses for those third hands or for corrections by the first hand itself, to limit the number of artificially created witnesses with very little difference between them. I have preferred the use of notes to indicate such readings in the collation

5.2. Method: Automated Collation Applied to a Classical Text

results. Although the creation of witnesses for each layer of correction makes more sense from a strictly methodological point of view, this was a practical decision to restrict redundant witnesses that would otherwise make the collation results more confusing (see Section 3.3.2).

The transcriptions have been done in the XML format, following the Text Encoding Initiative guidelines, P5 version (see Chapter 3). This choice was motivated not only because the TEI has become the standard format for manuscript transcription in the Humanities, but also because this would make it easier to produce different input formats, such as plain text or JSON, needed in order to experiment with the various collation tools.

5.2.2 *Automated Collation*

In total, there were ten witnesses in the final collation: eight witnesses for the first and second hands in the four manuscripts BCMN, and the two editions of Pithoeus and Håkanson. The damaged and lacunose manuscript A, which was transcribed and collated during the experiment, was ultimately left out of the final collation. This is not because it was impossible or too difficult to collate this manuscript, but rather because there were specific issues related to the visualisation of this manuscript which I have not had the opportunity to deal with (see Section 8.6.4).

The transcriptions were used as input to different collation tools. In Chapter 7, I describe in detail the use of three collation tools: CollateX, Juxta and the Classical Text Editor. There are many collation tools, as we have seen in Chapter 2, and although many do not exist anymore, about a dozen tools are still available or under active development (see the complete list in Appendix A.1). While I tried many of those, I decided to focus on the well-established tools that I considered would be most useful for editors, either because of the input and output formats (CollateX), the algorithm's efficiency (CollateX, Juxta), the user-friendly interface (Juxta, CTE) or the potential for creating directly a digital critical edition (CTE). I renounced other tools such as TUSTEP or Compare for practical reasons (see Chapter 7), and to tools such as LERA and Lakomp, iAligner, and eComparatio, because they were still in early or experimental stage. In Chapter 7, I will also explain why CollateX was ultimately the preferred tool for this dissertation.

5.2.3 *Visualisation of Collation Results*

There are several ways to process collation results: for instance to create a stemma, or a digital critical edition, or to visualise the results in a way that help scholars to

5.2. Method: Automated Collation Applied to a Classical Text

work with the collation results. In this dissertation, visualisation of the collation results has become a major area of interest. The options to not only visualise collation results, but also interact with those in order to edit the text, are still limited. Focusing on editors' needs, I have created a tool to visualise and process collation results, which will be described in detail in Chapter 8.

THIS chapter describes the TEI encoding applied to the transcriptions of Calpurnius Flaccus manuscripts and editions. A first section briefly reviews some of the transcription platforms available at the time of transcription and discusses their advantages and inconvenients. I have chosen to do the transcriptions within the oXygen editor and TextGrid Lab, and the reason behind that choice is explained in this section. The next section focuses on the exact encoding that I have applied to the manuscripts of Calpurnius Flaccus. The description first follows the structure of the documents (organisation in folios, lines, words). Then I will focus on the content, such as the issues of encoding scribal corrections, damages, special characters, and so on.

6.1 Transcription platforms

6.1.1 *TextGridLab and oXygen*

As my personal computer is a laptop with a small screen, it was important for me to work on a transcription platform with a form of cloud storage, so that I could access the transcription files from other computers with a larger screen and keep all the files synchronised. I also needed to have an environment which would facilitate encoding complex documents in TEI XML, such as oXygen. For these reasons I have transcribed mostly within the TextGrid framework, and the oXygen XML Editor. TextGrid is an open-source tool that supports scholarly research and especially digital editing¹. Notably, TextGrid has a Laboratory component TextGridLab that incorporates an XML editor with useful functionalities such as associating a schema to a document and validating against that schema, as well as debugging². TextGridLab is also linked to a digital repository, TextGridRep, where the transcriptions as well as manuscript facsimiles are stored and can be accessed

¹<https://textgrid.de/> (Accessed August 22, 2017).

²The XML Editor features are described here : <https://wiki.de.dariah.eu/display/TextGrid/Features+of+the+XML+Editor> (Accessed August 22, 2017).

6.1. Transcription platforms

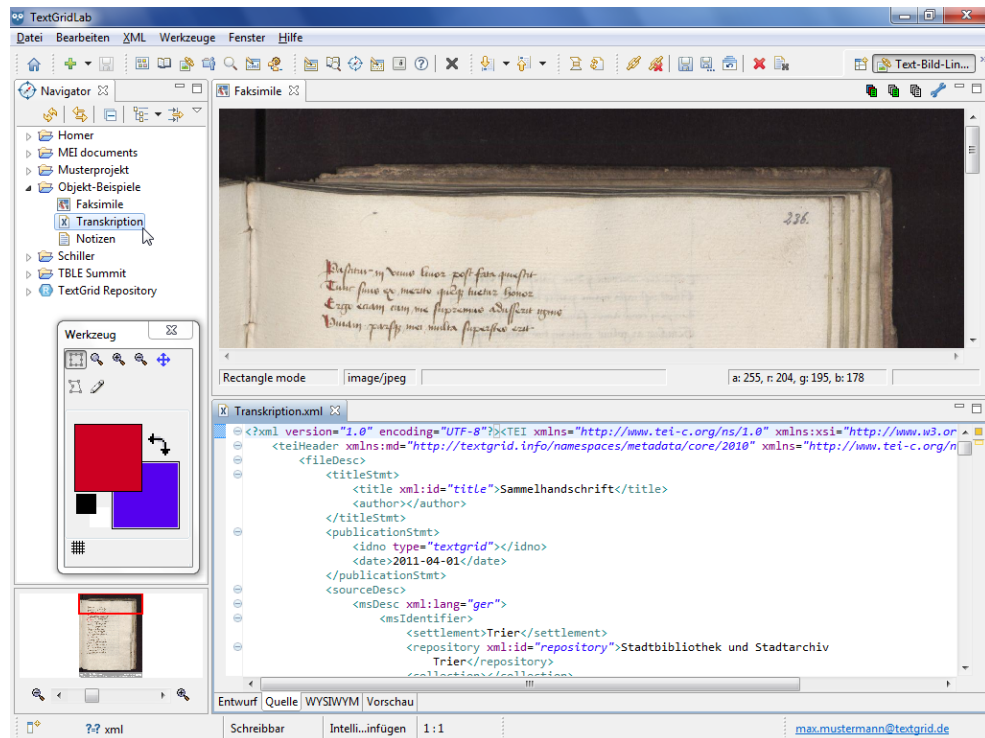


Figure 6.1: TextGridLab working environment. Retrieved from <https://textgrid.de/arbeiten-mit-verknuepfungen> (August 23, 2017).

from different computers³. In addition, TextGridLab provides a text-image linking facility. Since 2015, the technological components of TextGrid have migrated to DARIAH-DE. TextGrid integrates a version of CollateX, but with limited options: currently only plain text can be collated⁴.

6.1.2 Transcription Editor

The Transcription Editor was developed for the Workspace of Collaborative Editing, a joint project between the IGNTF project and the *Institut für Neutestamentliche Textforschung* in Münster (INTF) which also includes the Collation Editor (Houghton and Smith 2016). The Transcription Editor may have been a very practical solution, with an intuitive online interface and the possibility to make use of a large set of TEI elements. However, I did not know about it until the Digital Humanities conference in the summer of 2014, when I was already well advanced in my transcription work. Therefore I did not use the Transcription Editor for the transcription of Cal-

³<https://textgridrep.org/> (Accessed August 22, 2017).

⁴See <https://wiki.de.dariah.eu/display/TextGrid/Collating+Texts> (Accessed August 23, 2017). There is also a normalisation tool.

6.1. Transcription platforms

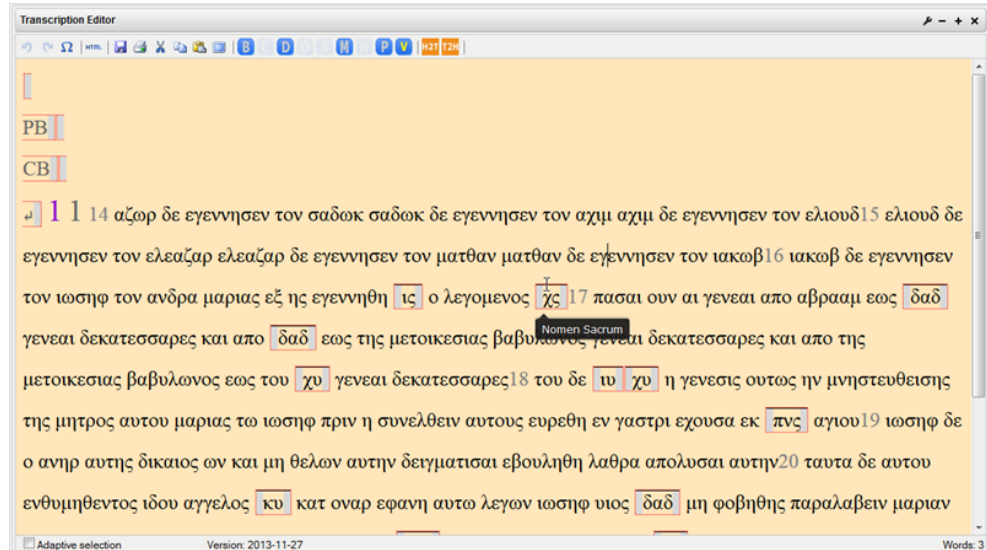


Figure 6.2: The Transcription Editor for the IGTP project. Retrieved from <http://www.birmingham.ac.uk/research/activity/itsee/projects/workspace.aspx> (August 22, 2017).

purnius, but I would certainly choose this tool if I had to start a new transcription and collation project. Although both TextGrid and the Transcription Editor can be used in conjunction with CollateX, the Collation Editor of the Workplace for Collaborative Editing offers more options to change the algorithm's parameters, to correct the collation results and to visualise them. For that reason it is a more attractive solution than TextGrid.

6.1.3 Transcription for Paleographical and Editorial Notation (T-PEN)

I have also experimented with the tool T-PEN, a project developed by the Center for Digital Theology at Saint Louis University (SLU)⁵. T-PEN divides the manuscript images into columns and lines, and then lets users do the transcription line-by-line. I experienced several issues with the first manuscript transcribed, the Montepesulanus H 126. The first issue was that the lines in manuscript A were not straight, which made the line division difficult. There may be solutions to this problem for experienced scholars. In a blog post about her use of T-PEN for transcription, Andrews (2015) explains that she has used the tool ImageMagick to de-skew the manuscript images⁶. However at the time of transcription, it did not seem worth spending too long on this issue given that it was only one among others. Another issue was a difficulty to zoom enough on the image in order to see the difficult passages in manuscript A. In addition, T-PEN does not facilitate TEI encoding,

⁵<http://t-pen.org/TPEN/> (Accessed August 22, 2017).

⁶<https://www.imagemagick.org/script/index.php> (Accessed August 22, 2017).

6.1. Transcription platforms

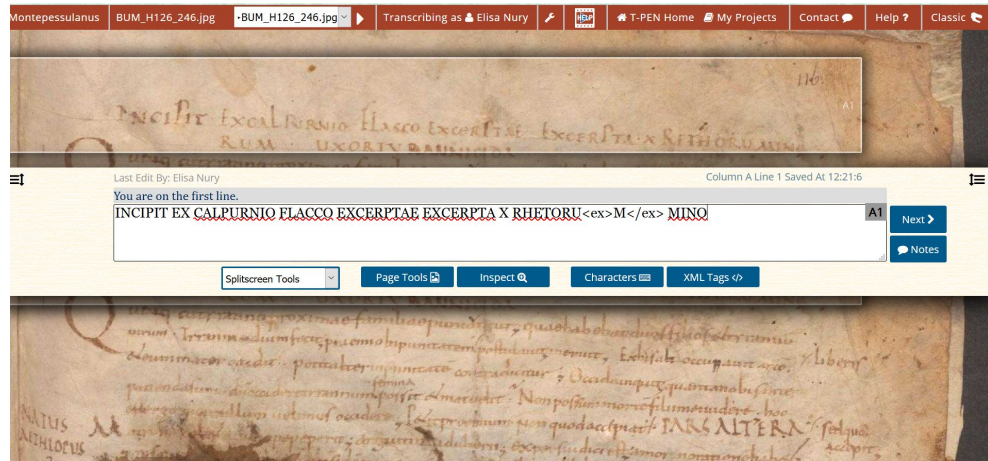


Figure 6.3: T-Pen transcription interface. Retrieved from my personal account (August 23, 2017).

although users can manually write XML tags in their transcription. Finally, there is no TEI XML export option. The available options, such as HTML or XML/Plaintext, must be updated to in order to create a valid TEI file. For that purpose Andrews (2015) created a python script, *tpen2tei*, which transforms the raw JSON data of T-PEN into valid TEI XML, and also tokenizes the text directly for use with CollateX⁷. Apart from these issues, the interface is very comfortable to work with, and since version 2.8 it provides new features to enhance the manuscript images, such as controls for brightness and contrast.

6.1.4 Juxta Editions: HumEdit Editor

During the spring of 2014, I was also part of the user group who tested the alpha version of Juxta Editions. In this platform there is some support for TEI XML encoding with the HumEdit Editor, as well as XML export of the transcription file. However, the set of elements available is limited to the TEI Lite version⁸, and other elements have to be included manually. I have found that the amount of space on the screen, reserved to the image of a manuscript and to the transcription, is rather small and less practical than the display of T-PEN, especially on a personal laptop (compare figure 6.3 and figure 6.4). The area need to be scrolled constantly as soon as the transcription is several lines long. However it may be a convenient transcription platform for projects of transcription with the aim of collating witnesses with Juxta, in particular when the elements available in TEI Lite are considered sufficient. Thanks to the collaboration option, it is possible to have multiple persons working

⁷<https://github.com/DHUniWien/tpen2tei> (Accessed August 23, 2017).

⁸<http://www.tei-c.org/Guidelines/Customization/Lite/> (Accessed August 23, 2017).

6.1. Transcription platforms

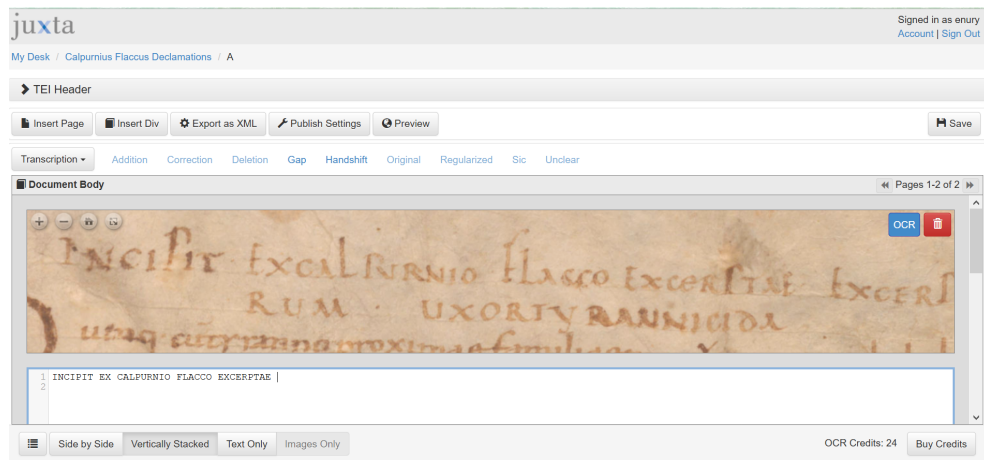


Figure 6.4: Transcription interface HumEdit in Juxta Editions. Retrieved from my personal account (August 23, 2017).

on the same transcription project. The Moravian Church Archive, for instance, uses Juxta Editions as a crowdsourcing transcription platform⁹.

6.1.5 *Transkribus*

Finally, it is worth mentioning the tool *Transkribus*¹⁰, although I have not transcribed within this platform. *Transkribus* is supported by the project *READ*¹¹, and is hosted at the University of Innsbruck in Austria. Although the transcription is not done directly in XML, it can be exported in various formats including TEI (*Transkribus* website). A Handwritten Text Recognition (HTR) engine is also available: after a certain amount of transcription has been done manually, the engine learns from this user-provided data to automatically recognise handwritten text. The HTR engine requires at least a hundred pages of manual transcription (or 20'000 words) in order to be efficient (*Transkribus Wiki*)¹². The tool is quite successful and has performed well at the ICFHR2016 Competition on the Classification of Medieval Handwritings in Latin Script (Kestemont and Stutzmann 2017; Cloppet et al. 2016). The text of Calpurnius Flaccus consists of roughly thirty pages (or 7'000 words) in each manuscript, which does not make it a good candidate for the Handwritten Text Recognition. However, the transcriptions that I have prepared could be reused if needed in order to apply the HTR to other texts present in the

⁹<http://www.moravianchurcharchives.org/online-transcription-project/> (Accessed August 30, 2017).

¹⁰<https://transkribus.eu/Transkribus/> (Accessed August 23, 2017).

¹¹<https://read.transkribus.eu/> (Accessed August 23, 2017).

¹²See https://transkribus.eu/wiki/index.php/Handwritten_Text_Recognition_Workflow (Accessed August 23, 2017).

6.2. Description of the TEI Encoding

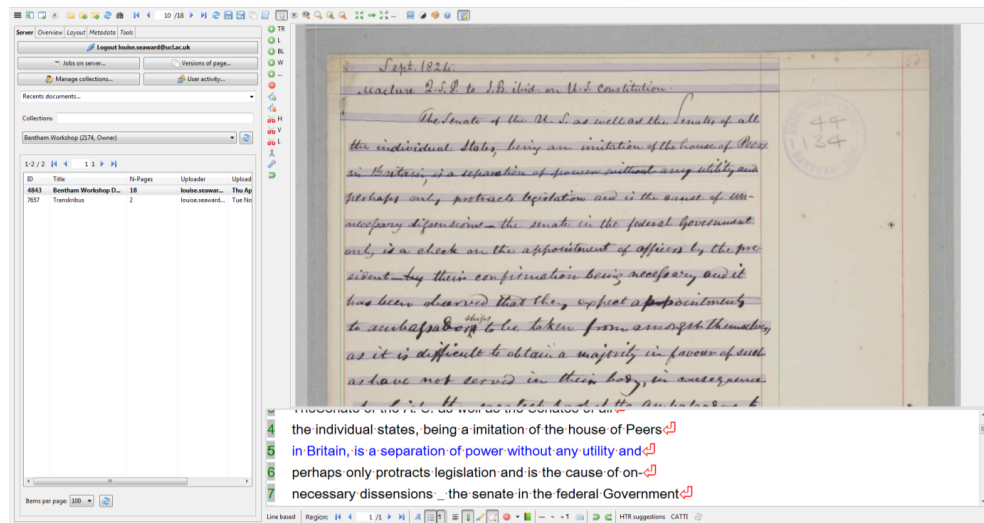


Figure 6.5: Transkribus interface. Retrieved from https://transkribus.eu/wiki/index.php/Main_Page#mediaviewer/File:Transkribus_interface.png (August 23, 2017).

same manuscripts, such as the *Minor Declamations* in manuscripts B and C for which there are hundreds of pages.

In conclusion, the choice of a transcription tool, as for collation tools, would benefit from tool criticism, to help users choose the most adequate tool. In this case I have chosen a tool that allowed for handling complex XML documents, and provided an online repository accessible from different computers. However, other users may have different requirements. The choice of a transcription tool may depend on the purpose of the transcription, the need for text-image linking, or on the data (for instance a very large amount of data would benefit from the Transkribus HTR engine). The list of transcription tools above is only a small sample of the tools available.

6.2 Description of the TEI Encoding

The transcriptions were created in the order in which I received the manuscripts facsimile from the libraries. The first was manuscript A, followed by B and M. I reused the two latter transcriptions and adapted them to match respectively C and N. Finally, Pithoeus' edition was transcribed last and from scratch in a new file. The bulk of transcription was carried out first within the TextGrid XML editor (see p. 182 above) and later on I switched to oXygen for the corrections and for the XSLT transformation into a format suitable for the collation tools (Section 7.1.3).

6.2. Description of the TEI Encoding

According to the levels of transcription listed in Chapter 3, this is a semi-diplomatic — or graphemic — transcription because the manuscript spelling is preserved but not the shape of letters, which belongs to the diplomatic — or graphetic — level (Robinson and Solopova 1993, 22). Some features of a regularised transcription are present as well, such as the regularisation of the letters i/j and u/v or the expansion of abbreviations to a standard form¹³.

I did not try to expand abbreviations to match the scribe's style because identifying the scribe's style correctly can prove delicate (see Section 6.2.3.1 below). I did not keep track of the abbreviation markers either, such as the tildes that indicate a letter m or n. The letter v is consistently used at the beginning of words, while u is used in the middle of words, except for capitalized text of declamation titles where the letter V is always present. Therefore the letters u/v were regularised to the conventional spelling (e.g. *uirum* and *iuuenes*, in manuscript B f. 147r and 148v, were regularised to *virum* and *iuvenes*). The letter j, which appears only as the second letter in a doublet of ij, was regularised to i (e.g. *adulterij* or *iudicij*, in B f. 147v, were regularised to *adulterii* and *iudicii*).

The following sections describe the TEI encoding of Calpurnius Flaccus, which is divided between a <teiHeader> and a <text> section. The description of the text encoding is further divided between the encoding of the structure (such as pages, lines, and words), and the encoding of the textual content, including abbreviations, regularisation of spelling, scribal corrections, and so on. The transcription of Pithoeus text is the only one that has an additional <back> section, which encodes the critical apparatus at the end of Pithoeus' critical edition.

6.2.1 TEI Header

Metadata about each transcription file is encoded in the file Description of TEI Header, such as a title statement and a publication statement: the documents are under the Creative Commons Attribution 4.0 International License (CC BY 4.0), which gives permission to share and adapt the material freely, even for commercial purposes.

The TEI Header also includes a manuscript description following the chapter ten of the TEI Guidelines P5 (TEI Consortium eds. 2017a, §10). Among the items encoded in the Header, some are used during collation: the witness siglum is encoded as the @xml:id attribute of msIdentifier. The handDesc and handNote elements provide

¹³It is also common in diplomatic transcriptions of Middle English to expand abbreviations into a standard form (Robinson and Solopova 1993, 22).

details about the different hands, and the `handNote` is used to create a different witness for each hand. It is also in the Header that special characters are encoded, such as the *e caudata* (e) for instance, as glyphs with two possible mappings, a diplomatic mapping or a more precise unicode mapping (see Section 6.2.3.1 below).

6.2.2 Structure

6.2.2.1 Pages

The start of each page is encoded in a `<pb>` element with `folio` or page number. The `@break` attribute is present when a page break separates a word into two parts. A link to a facsimile page is provided for manuscripts B, C and M, and the edition P1594, for which digital images are available online. Examples:

```
<pb n='244r' /> (N).
```

```
<pb n='252r' break='no' /> (N).
```

```
<pb n='383' facs='http://www.e-rara.ch/gep_g/content/pageview/1098914' />  
(P1594).
```

6.2.2.2 Forme works

The `<fw>` element may contain running headers, or gathering signatures and catchwords at the bottom of the page. These forme works appear mainly in manuscript M (39 occurrences) and in Pithoeus' edition (49 occurrences). Examples:

1. Head

```
<fw type='head' place='top-left'>
```

```
<hi rend='uppercase'>calp <pc>.</pc> flacci</hi></fw>
```

```
(P1594, 384).
```

2. Signature

```
<fw type='sig' place='bot-right'>a.1.</fw> (N, 244r).
```

3. Catchword

```
<fw type='catchword' place='bot-right'>travit.</fw> (N, 251v).
```

6.2.2.3 Declamations

Each declamation is contained in an `<ab>` element and numbered according to the total number of fifty-three declamations. As we have seen, there are two declamations missing from B, M, N, and P1594 (see Section 5.1.2.2). For instance the declamation 32 is combined with declamation 31 because of a lacuna (the end of declamation 31 and the beginning of declamation 32 are missing). I have

6.2. Description of the TEI Encoding

still marked the block of text in an `<ab>` element to indicate that it belongs to declamation 32. The *incipit* and *explicit* are also included in an `<ab>` element, with number 0 and 54 respectively. Examples:

```
<ab n='32' type='decl' />
<ab n='0' type='incipit'>
```

In the witnesses M, N, and P1594, declamations are numbered. This is encoded in a `<num>` tag. The declamation numbers are located in the margins of manuscripts M and N, and as a result they are encoded in a `<note>` of @type 'marginalia'. This avoids the issue of separating a word `<w>` in two with a number `<num>`, which is not allowed in the TEI¹⁴. Example:

```
<w>puni<note type='marginalia'><num>I</num></note>
<lb n='5' break='no' />antur</w> (N, 244r:4-5).
```

6.2.2.4 Lines

Each line starts with an `<lb>` element. The element includes line number, and @break attribute to indicate if a line break separates a word in two. If @break has the value "no", it means that the element `<lb>` is not word-breaking. Example:

```
tyrannici
<lb n='25' break='no' />dium (B, 147r:24-25).
```

6.2.2.5 Words

Each word is encoded in a `<w>` element. This decision was taken at a later stage of transcription: after the `<choice>` element (see below) was added to encode different possible outputs, either for the *e caudata* or for regularisation purpose, a marker was needed to indicate the beginning and end of the word in order to generate both the original and regularised reading. For instance *familię* (in manuscript C, folio 81v, line 22) is encoded as:

```
<w>famili<choice>
  <orig><g ref='#eogon' /></orig>
  <reg>ae</reg>
</choice></w>
```

This encoding allows for generating both the original *familię* and the regularised form *familiae*. For this reason I decided to add `<w>` tags to every word. Examples:

¹⁴See the description of the `<w>` element here: <http://www.tei-c.org/release/doc/tei-p5-doc/en/html/ref-w.html> (Accessed January 20, 2018).

6.2. Description of the TEI Encoding

```
<w>adhuc</w>  
<w>ad<choice><orig>o</orig><reg>u</reg></choice>lescens</w>
```

In other projects, different approaches can be found: the IGNTP guidelines and the Chronicle of Matthew both tag words, but the Textual Communities guidelines do not mention it. The Digital Mishnah project does not tag words either and the Beckett project rather tags segments at the sentence level.

Although the word tagging was necessary, it generated a problem of overlap with other tags, especially in manuscript B, where there are many corrections spanning sometimes letters across more than one word. There are the eight places in B and one in C with an overlap problem:

1. (B) 148v:6-7 *verberantibus* corrected to *verbera cibus*
2. (B) 150r:16 *L anni* to *danni*
3. (B) 156r:20 *et torquere* to *extorquere*
4. (B) 156v:22 *adulteravit* to *adultera vivit*
5. (B) 156v:28 *id idem* to *itidem*
6. (B) 157v:26 *eius* to *me ius*
7. (B) 158v:16 *nesesse* to *nec esse*
8. (B) 159r:11 *postremo* to *post reum*
9. (C) 83v:25 *quid est* to *quidem*

The issue here is that it is difficult to encode words in `<w>` tags to process them for collation, while simultaneously encoding precisely which letters were deleted or added. As a result, the solution was to provide two different segment elements `<seg>`: one segment with the `@type` 'tokenization' to encode the words for the purpose of collation, and one segment with the `@type` 'transcription' to encode a more accurate transcription. Example:

- Tokenization segment

```
<seg type='tokenization'><subst hand='#h2'>  
  <del><w>adulteravit</w></del>
```

```
<add hand='#h2'><w>adultera</w> <w>vivit</w></add>
</subst> </seg> (B, 156v:22).
```

- Transcription segment

```
<seg type='transcription'> adultera<add hand='#h2' place='above'>
vi</add>vit</seg> (B, 156v:22).
```

6.2.3 Content

6.2.3.1 Abbreviations and special characters

Abbreviated letters are supplied in `<ex>` elements. There is no indication of the abbreviation type (by suspension or contraction) or marking. The character entity `&` is used for `&`. Examples:

```
<w>carcere<ex>m</ex></w> (C, 82v:7).
```

```
<w>dem&amp;</w> (C, 86v:7).
```

```
<w><ex>e</ex>n<ex>im</ex></w> (B, 152r:23).
```

There are arguments for encoding more precisely the abbreviation markers so that the reader knows accurately what is present in the manuscript (Driscoll 2009), or for linguistic analysis purposes (Honkapohja 2013). Although the transcription of Calpurnius Flaccus does not record the exact abbreviation markers, it ensures that nothing is silently expanded. Every letter supplied by the transcriber that is part of an expansion but that does not actually appear on the manuscript is marked as such by the `<ex>` element. The practice of silent expansion can be criticised for several reasons. It contributes to ‘linguistic hybridity’, and masks the complex relationship between the grapheme sequence and the abbreviated word they represent (Honkapohja 2013, §5.1). Moreover abbreviations can help date and localise a manuscript (Honkapohja 2013, §1), and could possibly help to establish the text transmission if a variant originated as a copying error from an abbreviation.

Expanding abbreviations is not as straightforward as it seems. According to Driscoll (2009, 19), ‘it is standard practice when expanding abbreviations to do so in keeping with the normal orthographic practice of the scribe in question’. However, it is not always possible to know with certitude what the scribe would have written, since scribes are not always consistent. Consider manuscript B, whose scribe usually wrote *unquam* (148r:24, 152v:20, 153r:1, 155r:3 and 157r:5), *nunquam* (149v:2, 151v:22, 153r:5), etc, but once *nonnumquam* (147v:7). How should the only occurrence of *nāque* (153r:26) be then expanded, *nanque* or *namque*? When a word is always abbreviated, it becomes even more difficult to guess the scribe’s practice. For instance, *quamvis* is always abbreviated in C (82r:14, 85r:16, 89r:3), so that it

6.2. Description of the TEI Encoding

is difficult to know whether the scribe meant *quanvis* or *quamvis*. In addition, it may be difficult to make inferences about the scribe's practice when only a limited portion of text has been transcribed (the text of Calpurnius represents only a small fraction of manuscripts A, B and C, and of Pithoeus edition).

The regularisation of the e caudata is faced with the same kind of choices: for instance *fēmina* in manuscript B (147r:28 and 148r:14-15). Should it be *foemina*, because the e caudata stands for either 'ae' or 'oe'? The scribe of C, in particular, is not consistent in the distinction between 'e' and 'ē': while writing *cepit* for *coepit*, the scribe has also produced *excepit* for *excepit* (86r:28) or *percepit* for *percepit* (84v:6). The word *pēnē* can mean once *poenae* (B 155r:23, C 86v:24) and then *paene* (B 155v:8, C 89v:11) a few lines later.

The expansion to a non-regular form, such as *foemina* or *quanvis*, means that it must be further regularised for collation purposes. Therefore, I have decided to expand abbreviations or e caudata directly to the regularised form. I have declared a `<g>` element in `<charDecl>`, which can be regularised through a `<choice>`:

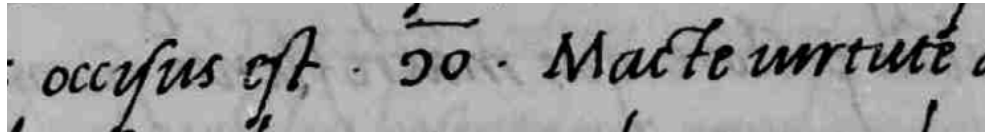
```
natur<choice><orig><g ref="#eogon"/></orig><reg>ae</reg></choice>
```

In a couple of cases, the same e caudata is regularised differently after a correction of a second hand. For instance in manuscript B, the form *quē* (folio 151v:18), seems to indicate that *quae* was corrected to *quem*. The other instance is in manuscript C, the form *al[t]erē* (folio 89v:16) shows that *alterae* was corrected into *alere*. In both cases the scribe changed the meaning of the e caudata without actually modifying the letter. Here is an example of encoding:

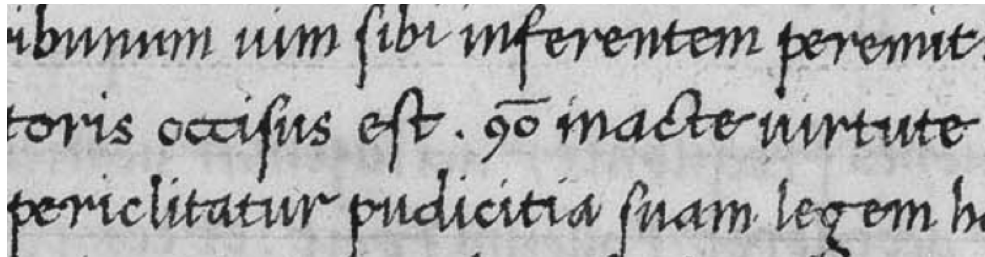
```
<del><w>alter<choice>
  <orig><g ref="#eogon"/></orig><reg>ae</reg>
</choice></w></del>
<add hand="#h2" place="overwrite"><w>aler<choice>
  <orig><g ref="#eogon"/></orig><reg>e</reg>
</choice></w></add>
```

Because of the inconsistencies in the use of the letter e caudata, it may be particularly interesting to keep track of the original form in the collation, especially when different interpretations result in variant readings¹⁵. For instance, Håkanson interprets once *quēri* as *queri* (manuscript C, folio 83v:40) and later he interprets

¹⁵Beneš (2003, 2) notes that scribes were not consistent with regard to the change from the diphthong *ae* to *ē* which 'proved perplexing to contemporary and later scribes, who often hyper-corrected their texts in instances where the diphthong did not appear at all'.



(a) B, folio 148r, line 8.



(b) C, folio 82r, line 18.

Figure 6.6: Unknown abbreviation in manuscripts B and C.

queritur as *quaeritur* (C, folio 87r:13). Therefore it is useful to keep the original form of the word as well as the regularised form for collation purposes.

One abbreviation in manuscripts B and C remains obscure (see figure 6.6). The first abbreviation sign stands for the Latin letters ‘con’ followed by a ‘o’, and the word was interpreted as *contra* by Lehnert (1903). It has been suggested that the scribe has written the abbreviation this way instead as ‘co’ with a tilde on top to avoid the confusion with ‘con’ and to represent graphically the opposition of ‘contra’ by flipping the letter C (Mike Kestemont, private email). This would also avoid a possible confusion with the similar abbreviation CD (*contradicit*), which is present on the same page in both manuscripts.

In the TEI Header, glyphs such as the e caudata, or the reverse C of the abbreviation above are mapped to both a diplomatic transcription and their equivalent character in the Medieval Unicode Font Initiative (MUFI)¹⁶. The abbreviation in manuscript B is mapped to the ‘LATIN ABBREVIATION SIGN CAPITAL CON’ Ɔ (MUFI 2183), and the one in manuscript C is mapped to the ‘LATIN CAPITAL LETTER CON’ (MUFI A76E). However, the character for MUFI A76E is available only in a limited number of fonts¹⁷, which is why the diplomatic transcription of this character is rendered with the character Ɔ in both manuscripts. The other special characters encoded in glyphs are the diples of manuscript MN (see p. 172), and the punctuation marks for punctus interrogativus (Section 6.2.3.7).

¹⁶<http://folk.uib.no/hnooh/mufi/> (Accessed August 17, 2017)

¹⁷See <http://www.fileformat.info/info/unicode/char/a76e/fontsupport.htm> (Accessed November 16, 2015).

Example:

```
<char xml:id='eogon'>
<charName>LATIN SMALL LETTER E WITH OGONEK</charName>
  <desc>The character ě (e caudata) has no standard mapping.
  The ě can be resolved in 'ae' in most cases, but also in 'oe' or 'e'.</desc>
  <mapping type='diplomatic'>ě</mapping>
  <mapping type='MUFI'>0119</mapping>
</char>
```

6.2.3.2 Orthographic Normalisation

One purpose of encoding orthographic normalisations is to be able to separate variants deemed significant from the variants which are considered less significant (see the discussion about normalisation in the Gothenburg model, Section 2.4.4.2). Orthographic variants are often considered less significant because they reflect the practice of the scribe who wrote the manuscript, instead of the authorial practice. However, there are still challenges related to the visualisation of orthographic variants (see Section 8.6.4).

For the transcription of Calpurnius, I have encoded a regularised spelling along the original spelling with a `<choice>` element. When a second hand decided to correct the spelling of a word, I did not regularise it because I wanted to be able to see those corrections as variants in the collation results. Examples:

1. Normalisation

```
<w>ad<choice><orig>o</orig><reg>u</reg></choice>lescens</w>
```

2. Without normalisation

```
<w>d<del>a</del>eniq<ex>ue</ex></w> (N, 252r:15).
```

The issue of normalisation is that it needs to be done after collation, because it may not be a good practice to make a decision without considering the entire manuscript evidence (Robinson 1989a), and it may be difficult to decide on a regularisation and stay consistent throughout the whole transcription process. The Collation Editor for the IGTP project, for instance, includes a regularisation tool to correct the collation results (Houghton and Smith 2016), as well as the CollateX interface in TextGrid (see Section 6.1.1). I have also proceeded similarly, by examining the first collation results that I obtained with CollateX, to find the orthographic differences which needed to be regularised. The problem of this method is that for each

6.2. Description of the TEI Encoding

orthographic difference spotted, I had to go back to the transcription and add the regularised form in every manuscript. This is rather tedious, and it shows how useful it would be to have the transcriptions and collation results linked, so that a correction in the collation would be reflected in the transcription (and vice-versa, a correction in the transcription would be reflected in the collation).

Another aspect of orthographic regularisation is related to inconsistent word division across witnesses. It is a fairly common phenomenon in Latin, since texts used to be written in the *scriptio* (or *scriptura*) *continua* style, without word division. This happens in words with two components, such as *res publica* or *contra dixit*, and it happens especially often in conjunctions or adverbs, such as *postquam*, *etsi*, *antequam*, *priusquam*, *iamdudum*, *iampridem*, *quamdiu*, *vixdum*, and so on. There were so many instances that it became difficult to regularise this phenomenon, since after several iterations of examining the correction results and encoding the regularisations, I would still find examples. In addition, it would generate issues of encoding overlap. However, when studying the collation results, it is easy to ignore whitespace characters, so that these differences in word division do not appear among the significant variants (see Section 8.4.2.3). For these reasons, I have not encoded the regularisation of word division directly into the transcription files.

The usage of letters u/v i/j is fairly consistent throughout a manuscript. V appears in uppercase text in manuscript A, and in later manuscripts in lowercase when at the start of a word. J is used only for the last i of a series (for instance *flij*, or in a number such as *xiii*). In modern editions, the practice seems to vary from one school to the other. The French tradition makes a distinction between u/v and i/j as can be seen in the Gaffiot dictionary, whereas the Oxford Latin Dictionary prints neither v nor j. I have followed the usage of the Teubner collection and regularised u/v to a standard form for a better readability (*vt* is regularised to *ut*, for instance, or *iuuenis* to *iuuenis*). For i/j, I have kept only the letter i and not the longer form j.

6.2.3.3 Scribal Corrections

Corrections are encoded as additions <add>, or deletions . The encoding of these may depend on how the manuscripts are divided into witnesses (see Section 3.3.2). Each layer of text, such as the text of the first hand and the corrections of a second must be clearly identified in order to be collated. However this can be difficult to achieve: for instance, it may be impossible to know with certainty which scribe deleted a word, the first hand or a later hand? For additions as well, scholars may disagree about which hand actually added a word (see for instance the discussion about the corrections in manuscript N, Section 5.1.2.5). I have decided

to avoid creating many artificial witnesses with very similar texts, and so I have not separated the manuscripts into two witnesses for the text of the first hand and the corrections by the first hand itself. I have also decided against separating manuscript N into more than two hands. Instead, I have divided each manuscript in two witnesses, one for the text of the first hand, and the other for the corrections by the first hand or later ones. I have included comments on words to indicate if a correction seemed to have been made by the first hand itself or to indicate a possible third hand in manuscript N.

Deletions There are only three exceptions where it seems clear that the first hand deleted a word or letter because it was a mistake. In manuscript M (4v:19), for instance, the letters '*dene*' are crossed and then followed by the word *dementiae*. This deletion is encoded with an attribute @type 'corrigendum' to denote a mistake, and an attribute @hand 'h1'. The two other cases are located in manuscript B (151v:20 and 154v:14). These mistakes were ignored during the collation. In manuscript N, for only three letters, a correction by the second hand was deleted, possibly by a third hand, or by the second hand itself: *cruciar* was corrected to *cruciararius*, and then back to *cruciar* (250v:15). This deletion has an attribute @hand '#h2'. All the other elements have no attribute to indicate which hand made the deletion. The text which was thus deleted is incorporated into the witnesses for the first hands. Examples:

1. Corrigendum

```
<del hand='#h1' type='corrigendum'><w>dene</w></del>  
(M, 4v:19).
```

2. Correction in N

```
<w>cruciar<del hand='#h2'>  
  <add hand='#h2' place='above'>iar</add>  
  <note>-iar- added by N2 and then deleted by N2 or N3.</note>  
</del>ius</w> (N, 250v:15).
```

3. Normal deletion

```
<del><w>linquetur</w></del> (B, 148v:11).
```

Additions Addition elements <add> contain two attributes: @hand for the hand which made the addition, and @place to indicate where it was written. The possible values of @place are 'margin', 'above', 'inline' or 'overwrite'. The @hand attribute usually indicates the second hand '#h2'. In two cases the addition seemed likely to

6.2. Description of the TEI Encoding

be made by the first hand, and has no @hand attribute (it appears in manuscript B, 154v:14 and manuscript C, 88r:40). Examples:

1. First hand addition

```
<w>p<add place='above'>o</add>stis  
<note>-o- added, likely by C1.</note></w> (C, 82r:40).
```

2. Second hand addition

```
<w>indemnator<add hand='#h2'  
place='above'><ex>um</ex></add></w> (C, 88v:38).
```

6.2.3.4 Unclear words or letter

When in doubt regarding transcription, I followed the text of Håkanson and I encoded the unclear text in <unclear> with an attribute @resp that refers to the edition of Håkanson (in a <bibl> element of the TEI header). I had doubts especially in cases of a correction by a second hand where it is difficult to decide which is the original text and which is the corrected text. This situation happens when the correction overwrites the original text (for instance in manuscript B, f.150r:27, for the readings *decet* and *dicet*). It happens also when the ink from one side of a folio has bled through the paper and may render the text on the other side of the folio difficult to decipher (for example in manuscript M, 18r:24 for the reading *mutuo fratres*). The attribute @reason gives more precision such as 'unknown order of correction'. The content of the attribute @reason will serve the purpose of keeping track of unclear words when examining the collation results (Section 6.2.3.8).

6.2.3.5 Gaps, spaces and supplied text

An illegible passage is noted as <gap/>, if possible with indication of length. Empty lines in B and C are noted as <space/>, as well as spaces left between two words. I have also added a description <desc> for the space in order to provide more details: for instance the empty lines in B and C could have been left blank in order to add a declamation title. In manuscript C, the title of declamation fourteen is 'Calpurnius' followed by seven dots, which might correspond to the author's name 'Flaccus'. Examples:

- Gap

```
<del><gap extent='1' unit='word' /></del> (M, 14v:15).
```

- Space

```
<space extent='7' unit='chars' rend='dots'>
<desc> The seven dots could be for 'flaccus'.</desc></space>
(C, 84r:38).
```

I supplied missing letters which were too close to the spine fold and do not appear in the reproduction image, as well as the initial of the first declamation in manuscript B, that should probably have been added later by another scribe. Example:

```
<w><supplied>q</supplied>uinque</w> (B, 147r:23).
```

6.2.3.6 Highlights

Capitals, small capitals, larger initials as well as colour or italics are encoded in `<hi>` elements, distinguishing between italics, roman, bold, uppercase and red¹⁸. Where different sorts of highlighting were present in one word or sentence, I nested several `<hi>` elements. Example:

```
<hi rend='red'>
  <hi rend='underline'>incipiunt ex calpurnio flacco ...</hi>
... uxor tyra<ex>n</ex>nicida</hi> (C, 81v:21).
```

6.2.3.7 Punctuation

I have encoded the punctuation for the complete text of both Pithoeus and Håkanson. In the case of the manuscripts, I have encoded punctuation only for the first three declamations, as a short sample. Punctuation marks are encoded in `<pc>` elements. While it is rather straightforward to encode punctuation of modern editors, it can be more challenging for manuscripts. The black and white facsimile reproductions make it harder to distinguish between actual punctuation marks and mere specks of ink or dirt, and faded ink is difficult to spot. For an accurate transcription of punctuation it would be best to check the manuscripts themselves. In the manuscripts I have encountered mostly the following marks:

- punctus (. and · in B).
- comma (,).
- colon (:).

¹⁸The colour actually would need to be checked in the manuscripts to be certain, as I have only black and white copies. A highlight in 'red' marks the fact that the shade of ink is obviously different.

- dashes (- BC and = MN) to mark hyphenation at the end of a line.

In addition, the punctus interrogativus is present in CMN, and the colon (;) appears in M only for this short sample. The punctus interrogativus was encoded as a glyph ‘qm’, with a diplomatic mapping to the mark .

6.2.3.8 Editorial Notes

I have added <note> elements to encode a comment that I made during the transcription and which I wanted to be able to see in the collation results. Some examples have already been discussed, for instance in the section about scribal corrections (Section 6.2.3.3). Other notes comment on doubts that I have had during the transcription, which may become relevant during collation if there is a variant reading at this point. Notes also record differences between the 1580 and 1594 editions of Pithoeus. Examples:

- Unknown abbreviation

```
<w>
  <choice>
    <orig><ex><g ref=#condes' />O</ex></orig>
    <reg resp=#Lehnert'>contra</reg>
  </choice>
  <note>Unknown abbreviation.</note>
</w> (C, 82r:18).
```

- Hesitation on the text

```
<w>accedere
  <note>Is there an abbreviation mark? accendere(n)t?</note>
t</w> (C, 82v:19).
```

6.2.4 The Critical Apparatus of Pithoeus

The critical edition of Pithoeus is appended with an apparatus, where he quotes variant readings from the two manuscripts he had at his disposal (see Sections 5.1.2.6 and 8.5). The ancient manuscript (*vetustius exemplar*) identified with manuscript A is referred to as *vet.*, and the other manuscript (*alterum exemplar*), the Italian exemplar, is referred to as *al.* in Pithoeus' apparatus. In addition, Pithoeus proposes a few conjectures, two of which were suggested to him by his nephew Pierre Nevelet [Petrus Neveletus]. The apparatus is added in a section called *variae lec-*

6.2. Description of the TEI Encoding

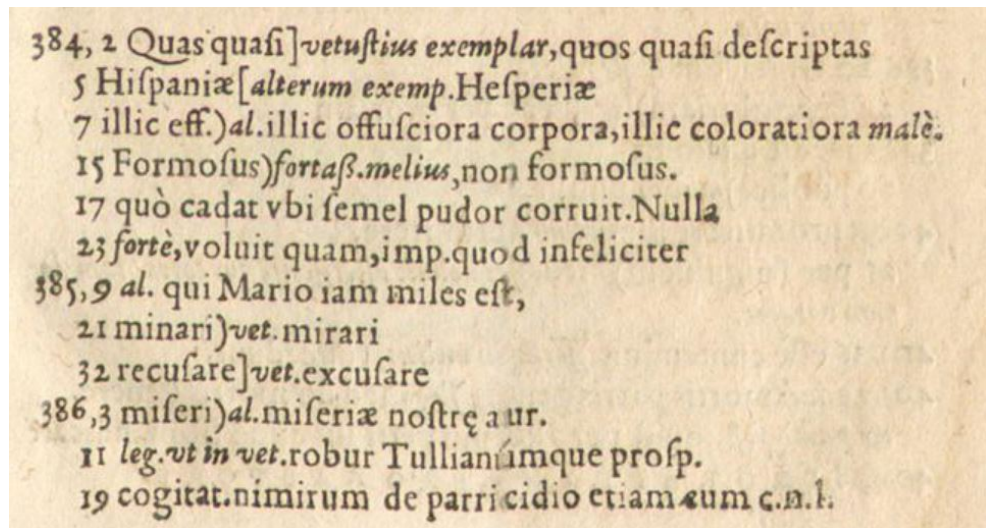


Figure 6.7: The beginning of Pithoeus' apparatus on Calpurnius Flaccus.

tiones (variant readings) at the end of the edition, and has no page numbers¹⁹. It is worth noting that Pithoeus was not an academic, and that publications such as the critical edition of Calpurnius were leisure activities (C. Dionisotti, private email). The critical apparatus was a set of notes, not as elaborate as the main text, for the purpose of presenting the interested public with documents (Dionisotti, same email). This may explain the presence of imprecisions. Figure 6.7 shows an extract of the apparatus.

I will refer to an apparatus entry with page and line number as they appear and the number under which the entry is encoded in the transcription: 384:2(1) is the first apparatus entry of Calpurnius Flaccus. Most entries have a few words of the critical text to help the reader locate the text, followed by a parenthesis, and then by a variant reading. In entry 384:7(3), for instance, *illic eff.* helps the reader to locate the critical text (*illic effusiora corpora, illic collectiora*). This is followed by the reading of the other exemplar *al.* which is *illic offusciora corpora, illic coloratiora*. In some cases Pithoeus does not introduce the apparatus entry with the corresponding critical text, such as for entry 384:17(5). It may also happen that Pithoeus combines two variant readings which are close by in the text under the same apparatus entry with the words & *paulo post* (and a little further), for instance entry 403:29(43).

¹⁹The variant readings of Calpurnius' text start here, and continue on the next two pages: http://www.e-rara.ch/gep_g/content/pageview/1099004 (Accessed August 21, 2017).

6.2. Description of the TEI Encoding

Regarding the imprecisions of the apparatus, it can be noted that line numbers are not always referring to the correct line of the critical text. The text related to apparatus entry 389:31(21) is actually located in lines 28-29 of the text in both the 1580 and 1594 edition. Other examples include entries 390:9(22), 393:18(31) 402:16(42), and so on. In entry 490:13(23), the word *paricida* was misspelled, although it is correctly written as *parricida* in the full critical text. In the apparatus entry 386:3(10), Pithoeus quotes the reading *miseriae nostrę aur.* as the reading from the Italian exemplar. However this reading is not present in any manuscript at this point in the text. Whether this is an important mistake of Pithoeus or an indication about the Italian exemplar is discussed later in the thesis (Section 8.5). According to the apparatus entry 403:29(43), the critical text is *sed mortis patris genere*, however the full text reads *sed de mortis patris genere*.

The apparatus entry 392:29(30) apparently quotes a reading from manuscript A (*vet.*) at a point in the text that Pithoeus likely did not find in A. The text of manuscript A is believed to have contained up to the six or seven first declamations of Calpurnius. It is possible to read the beginning of the sixth declamation in the last folio 116v, followed by ten illegible lines. However, the reading in Pithoeus apparatus is located in declamation 11, which he could not have read unless he had an additional folio at his disposal.

I have encoded the critical apparatus in the <back> element, with the location-referenced method. Each apparatus entry is encoded in an <app> element with a @loc attribute for the location in the text, an @n attribute and an @xml:id. The corresponding words or punctuation marks in the critical text have an attribute @corresp linking to the apparatus entry. Every apparatus entry has a <note> element which encodes the exact text of the entry as printed. The <app> element may also contain a lemma <lem> and the variant readings <rdg>, except when there is more than one lemma discussed in the same apparatus entry. Because of the inconsistencies and errors present in the apparatus, I have not attached a @wit attribute to the <lem> and <rdg> elements.

Example:

1. Text

```
<w corresp='#app8'>minari</w> <pc>.</pc>
```

2. Apparatus

```
<app n='8' loc='385 21' xml:id='app8'>
```

```
<lem>minari</lem>
<rdg>mirari</rdg>
<note>21 minari<hi rend='italic'>vet.</hi>mirari</note>
</app>
```

6.2.5 Modern Editions Encoding

Modern editions have emendations, which are encoded differently from manuscript witnesses, such as additions, deletions, the conjecture of a lacuna, or the marking of a passage as *locus desperatus* with a crux. A *locus desperatus* (that is ‘hopeless passage’) is a passage of text that the editor considers corrupted, but for which they are not able to offer an emendation. In the case of Calpurnius, the only modern edition is the one of Håkanson.

A *locus desperatus* is encoded with the element `<sic>` and an `@ana` attribute which gives the explanation ‘locus desperatus’. The crux symbol † is encoded with the entity `†`. Example:

```
<sic ana='locus desperatus.'><w>&#x2020;mittit</w></sic> (LH, 2:9).
```

Additions and deletions by an editor are represented as a different phenomenon than scribal corrections in manuscripts. In manuscripts, the `<add>` and `` elements encode the action of different hands on the page, the actual addition or deletion of a sequence of words or letters. In an edition, the `<supplied>` element encodes the editorial decision of adding text that is not present in any witnesses. Lacunae are often supplied by editors who suspect that some text is missing. On the other hand, the `<surplus>` elements encode text that is present in all witnesses, but that the editor judged to be superfluous. Examples:

1. Surplus text

```
<w>amavi<surplus>t</surplus></w> (LH, 2:12).
```

2. Supplied text

```
<w><supplied>in</supplied>feliciter</w> (LH, 2:23).
```

3. Supplied lacuna

```
<supplied><space extent='unknown' unit='length' /></supplied>
(LH, 2:8).
```

The text of Håkanson is available online at the Packard Humanities Institute (PHI)

6.2. Description of the TEI Encoding

website for Classical Latin Texts²⁰. I have used this resource to point readings of Håkanson to a digital facsimile. However, the PHI edition does not follow the pages of the original Teubner text. Instead, each declamation is on a different webpage. For this reason I have used the @facs attribute on <ab> elements instead of the <pb> elements.

²⁰<http://latin.packhum.org/author/1100> (Accessed September 1, 2017).

THIS chapter focusses on the practice of automated collation with various tools which were available during this research. The transcription files, described in the previous chapter, were used to collate the *Declamations* of Calpurnius Flaccus with three different tools: CollateX, Juxta and the Classical Text Editor (CTE). This chapter describes the collation procedure to work with each of those tools, and examines both the advantages and issues of each tool. The interface, the various combinations of input and output formats, as well as the possible options for influencing the collation algorithm were reviewed. In the case of CollateX and Juxta especially, there is more than one interface to the collation algorithm, as the tools evolved and new versions were created during the course of this thesis.

The set of collation tools available also evolved during the preparation of this thesis, and resulted in some changes among the tools that were reviewed. At the beginning of this dissertation in 2013, four of the main collation tools were selected to collate the text of Calpurnius: CollateX, Juxta, Schmidt's Compare (formerly nmerge) and TXSTEP. Although Schmidt proposed to add the transcriptions of the *Declamations* to the Ecdosis platform and to compare them with his collation tool, this was not successful due to encoding incompatibilities. As it turned out, the TEI encoding of line-breaks elements `<lb>` at the beginning of each line was causing issues. Since the only straightforward solution was to wrap lines in `<l>` elements, which is reserved for lines of poetry and would be contrary to the Guidelines (TEI Consortium eds. 2017d, §3.12.1), this option was abandoned. I also explored TUSTEP and TXSTEP, but it is incredibly complex to install and to operate¹. TUSTEP requires a long investment in time to peruse the 1400-page long user manual and for a limited benefit, at least with regard to automated collation: a '[c]ollation algorithm

¹TUSTEP 'is highly complex and requires training for use', and requires an advanced level of technical expertise (Huculak and Richardson 2013). See also a discussion of May 2011 on the TEI-L list: 'Learning TUSTEP is not a walk in the park'. <http://tei-l.970651.n3.nabble.com/TEI-gt-print-what-solutions-for-a-neat-printed-version-of-a-critical-edition-td2891563.html> (Accessed September 03, 2017).

is present, although the documentation suggests it is outdated (in comparison to newer languages)' (Huculak and Richardson 2013). On the other hand, a series of new collation tools have become available since 2015, such as iAligner, Traviz, or CTE (see Chapter 2). The proliferation of new tools made TUSTEP a less attractive solution. Of the newly available options, I have chosen to examine CTE because it has a different purpose from other collation tools, and is similar to TUSTEP in some regards: it is a popular editor for traditional printed editions, with a focus on typesetting complex critical editions. In addition, the collation results from CTE may also be used with Stemmaweb in order to automatically create a stemma.

The main focus will be on CollateX, since it was the preferred collation tool for this research, especially for the purpose of visualisation and manipulation of the results. The output from CollateX is particularly easy to process for further manipulation. In Juxta for instance, once the collation is done and results are displayed, there is little left to do apart from visualising and commenting on variants or variant locations. However, scholars may need to filter out collation results in order to find patterns that could indicate a relationship between witnesses, such as agreements in error of a group of witnesses (see Chapter 8).

For each collation tool, this discussion will follow the criteria that were identified previously in Section 2.6 for the assessment of automated collation tools, namely the interface, the data preparation and input format, the process of collation and options available to influence the algorithm, and finally the output obtained from the tool. In addition, I will also consider the perspective of different kind of users: are users scholars working on their own, or in collaboration? How does the users' level of technical knowledge affect their experience with the tool? Who are the intended users: editors aiming at preparing a critical edition, readers of the edition, textual critics, or students? How does the tool integrate the entire collation workflow, from transcription to analysis of results? Has the evolution of these tools in recent years impacted upon any of these issues?

7.1 CollateX

The development of CollateX was centred on the alignment algorithm, and therefore no Graphical User Interface (GUI) was created, except for demo purposes only². There are three interfaces to CollateX, which will be briefly surveyed: the command line tool and the RESTful web service which both run the Java version of CollateX, and the Python version. The interface used in this project is the command line

²The demo version is available here: <https://collatex.net/demo> (Accessed August 23, 2017).

```

usage: collatex [<options>]
      (<json_input> | <witness_1> <witness_2> [[<witness_3>] ...])
-a,--algorithm <arg>          progressive alignment algorithm to
                              use 'dekker' (default), 'medite',
                              'needleman-wunsch'
-cp,--context-path <arg>      URL base/context path of the
                              service, default: '/'
-dot,--dot-path <arg>         path to Graphviz 'dot',
                              auto-detected by default
-f,--format <arg>             result/output format: 'json', 'csv',
                              'dot', 'graphml', 'tei'
-h,--help                     print usage instructions
-ie,--input-encoding <arg>    charset to use for decoding non-XML
                              witnesses; default: UTF-8
-mcs,--max-collation-size <arg> maximum number of characters
                              (counted over all witnesses) to
                              perform collations on, default:
                              unlimited
-mpc,--max-parallel-collations <arg> maximum number of collations to
                              perform in parallel, default: 2
-o,--output <arg>             output file; '-' for standard output
                              (default)
-oe,--output-encoding <arg>   charset to use for encoding the
                              output; default: UTF-8
-p,--port <arg>               HTTP port to bind server to,
                              default: 7369
-s,--script <arg>             ECMA/JavaScript resource with
                              functions to be plugged into the
                              alignment algorithm
-S,--http                     start RESTful HTTP server
-t,--tokenized                 consecutive matches of tokens will
                              *not* be joined to segments
-xml,--xml-mode                witnesses are treated as XML
                              documents
-xp,--xpath <arg>             XPath 1.0 expression evaluating to
                              tokens of XML witnesses; default:
                              '//text()'

```

Figure 7.1: CollateX command line usage (CollateX Documentation).

tool, which is one of the simplest option for scholars working on their own. The RESTful interface web service is most useful for integrating CollateX within a web application. The Python version was not yet fully functional when I tried it (the JSON output that I have been working with was implemented in July 2016).

7.1.1 CollateX Interfaces

7.1.1.1 Command Line Tool

The easiest way to start using CollateX is to download the command line tool. CollateX can be executed from the command line with the keyword ‘collatex’ followed by different options, and by the transcription files as input (see figure 7.1). Between the start of this research and the time of writing, the command line tool has evolved from version 1.4 to version 1.7.1. The collation results which are examined in Chapter 8, and available in Appendix B.3, were obtained with CollateX version 1.7.1.

The first attempts with CollateX 1.4 produced a lot of collation errors when attempting to collate more than a very small number of witnesses: for instance, when I

added Pithoeus' edition to the manuscripts B, M, and N, the readings of Pithoeus were not properly aligned with the readings of the other witnesses. It was suggested that the witnesses should be given to CollateX in order of similarity, from the witnesses which were most alike to the ones which were most different, in order to help the collation algorithm (Gregor Middell, Collation Workshop in Münster 2014). However, the issue was resolved with the next version of CollateX 1.5, which has been available since December 2013.

The procedure for installation and usage has also changed from version 1.4 to version 1.7.1. In any case, the Java Runtime Environment (JRE) is necessary, since this version of CollateX is written in the Java programming language. For version 1.4 and 1.5, CollateX came as a zip file which could be saved anywhere on the computer and its content extracted. Since I was working on a Mac OS 10.6, I had to change the PATH variable in order to access CollateX. The PATH variable is stored in a special hidden file called 'bash_profile' which stores instructions for the command line tool, such as a shortcut to CollateX so that the program can be accessed from anywhere on the computer. See also Raabe (2014) for a blog post about installing and running version 1.5 of CollateX. As Raabe's experience shows, the installation of CollateX can quickly become complex and could be offputting for someone with a limited knowledge of computing.

The next versions of CollateX come as a Java archive (JAR), a file format which stores many Java files and 'allows to deploy an entire application [...] in one single request' (Wikipedia)³. Unlike the previous zip file, the Java archive content is not extracted. The JAR file is called via the command line as specified in CollateX download page: `java -jar collatex-tools-1.7.1.jar -h` This command is the 'help' command that will show how to use CollateX (see figure 7.1 above). Here is an example of a command that was used to collate the witnesses of Calpurnius Flaccus:

```
java -jar -Xmx1g collatex-tools-1.7.1.jar -f json
-o calpurnius-nov2016-collated-norm-joint-BCMNP.H.json
../json-tokenisations/nov2016-calpurnius-json-tokenized-BCMNP.H.json
```

There are several elements to note in this command, namely:

1. `-Xmx1g`: it was necessary to add this parameter to the command, due to a Java error ('out of memory' error). This parameter determines the maximum memory that Java is allowed to use and the minimum value is usually 256MB.

³[https://en.wikipedia.org/wiki/JAR_\(file_format\)](https://en.wikipedia.org/wiki/JAR_(file_format)) (Accessed August 27, 2017).

Setting the parameter to 1G allows CollateX to use enough memory in order to finish the collation.

2. `-f json`: this is the desired output format for the collation results.
3. `-o calpurnius-nov2016-collated-norm-joint-BCMNPB.json`.
This is the name of the output file which will be created by CollateX, and which will contain the collation results.
4. `../json-tokenisations/nov2016-calpurnius-json-tokenized-BCMNPB.json`.
This is the path to the input file containing the transcriptions of witnesses.

The advantages of using the Java version of CollateX in the command line environment were first that it was rather easy to install and run. Second, at the time of collation, the Java version was the most advanced version while the Python version was still under development and had a few shortcomings. The most important of those shortcomings was the absence of the JSON output which I was working with in order to analyse the collation results. On the other hand, there is also some inconvenience involved in collating with the command line: it means that the workflow of collation is divided into separate steps which are not linked together. First the transcription files are processed in oXygen with XSLT to obtain the input for CollateX (see Section 7.1.3 below), then the collation is performed within the command line, and finally the collation results are analysed with a Python script (PyCoviz, a Jupyter notebook described in Chapter 8). This split workflow makes it more difficult to achieve reproducible results, and to keep track of which transcription files were collated with which version of CollateX, as well as which CollateX results were processed with which version of the PyCoviz notebook.

7.1.1.2 RESTful Web Service

'**Representational state transfer (REST)** or **RESTful** web services is a way of providing interoperability between computer systems on the Internet' (Wikipedia)⁴. The RESTful web service lets online applications or websites access the Java version of CollateX through the HTTP-based Javascript API. For instance, projects such as the Workspace for Collaborative Editing (Houghton, Sievers, and Smith 2014), the Digital Mishah, or the Beckett Digital Manuscript Project (BDMP), make use of the RESTful service to provide on-the-fly collation in a web application: users can select a passage of text and a number of witnesses, and receive the collation results almost instantly. For instance in the BDMP, users send requests which are

⁴https://en.wikipedia.org/wiki/Representational_state_transfer (Accessed August 24, 2017).

transmitted to the Collatex server. CollateX performs the collation and sends back an output which is transformed into HTML for users to visualise online in the browser (Dekker et al. 2015).

The advantages of such an on-the-fly collation is that it prevents the server from storing redundant data on the project's server, since the collation results are not stored but generated each time a users sends a collation request. The system also gives users a relative freedom: the Digital Mishnah, for instance, was conceived as a tool rather than a traditional critical edition, in the sense that the edition does not commit to one theory of text presentation, but gives several options to the user (Lapin 2013, 448). Users of the Digital Mishnah have to select the passage of text they want to work with, then select the witnesses and the order in which they are collated, and finally choose a visualisation option (collation table, text and apparatus, or parallel texts in column). With collation via the RESTful web service, it is also possible to have slightly more control over the collation process than from the command line interface, since there is limited support to customise the token equivalence function (see Section 2.4.4.1). The major inconvenience of this method is that it is a very complex system to set up, and may require the help of a developer. It is a good solution for a large project with appropriate support, but not the best method for a single scholar who does not need to provide on-the-fly collation to users over the web.

7.1.1.3 Python Version

The last way to access CollateX is via the Python version. It is a port from the Java version to Python: the program is not only adapted to a new programming language, but it also implements a new collation algorithm. Between its first release in June 2014 and the latest update in January 2017, the Python version was under active development, and a number of bug fixes and new features were added over twenty-six updates⁵. The Python version requires a small amount of coding knowledge, which may be problematic for Humanities scholars who have not received the relevant training (see Raabe 2014, 2015).

The main issue with the Python version, for this project, was that the JSON output did not include full token representation, which was needed for visualisation purposes. This output was added in mid 2016 (in the 2.0.0orc20 release). For this reason I did not collate with the Python version. However, as the Python version of CollateX improves, it will become more attractive than the command line. In

⁵<https://pypi.python.org/pypi/collatex/2.1.2> (Accessed August 28, 2017).

particular, it makes it possible to combine the entire collation workflow directly in Python. In addition, the Python version, as well as the RESTful web service, offers more control over the collation process than the command line with the possibility to change a wider range of parameters. As a matter of fact, the Python version of CollateX is the one that is now taught to interested scholars during workshops, such as the Code & Collation workshop (Bleeker and Spadini 2016), or the Make Your Edition summer school in 2017⁶.

7.1.2 *CollateX Input Formats*

There are three input formats accepted by CollateX: plain text, XML, and JSON. Plain text and XML are in fact very similar, since plain text tokens are extracted from a selection of XML elements. This input format implies a loss of context: there is no connection to the XML markup context, or to the witnesses' facsimile (with information such as page number). In this case, the correction and analysis of the collation output can be limited by this loss of context (see also Section 2.4.4.1 about Tokenization and its issues). On the other hand the JSON input is more flexible.

JavaScript Object Notation (JSON) is a syntax for storing and exchanging data. The format, comprising only the two data structures of objects and arrays, is easy to understand (see p. 212 below). Because data is stored in a textual format, it is language independent: it can be used by any programming language⁷. There are two JSON input formats which are accepted by CollateX: the first one is a plain text input, and the second one is a pretokenised input (see figure 7.2). The difference between the two inputs is that CollateX will do the tokenization stage of the Gothenburg Model with its own tokenization function on the plain text input, while the other input has already been divided into tokens according to the user's need. In addition, the pretokenised JSON input includes additional properties added by users which will be ignored during collation, but will still be available in the results in order to improve the visualisation (see also Chapter 4, and Chapter 8 for examples of how these properties can be used).

In conclusion, the pretokenised JSON input appeared to be the best format for collation, because it could retain information about the tokens, such as the position in the text that would make it easier to check the results against the manuscripts, or such as editorial comments made during the transcription that would be helpful when analysing the collation results (see Section 8.1.1). Therefore the TEI transcrip-

⁶<https://pittsburgh-neh-institute.github.io/Institute-Materials-2017/> (Accessed August 28, 2017).

⁷See <http://www.json.org/> and https://www.w3schools.com/js/js_json_intro.asp (Accessed August 28, 2017).

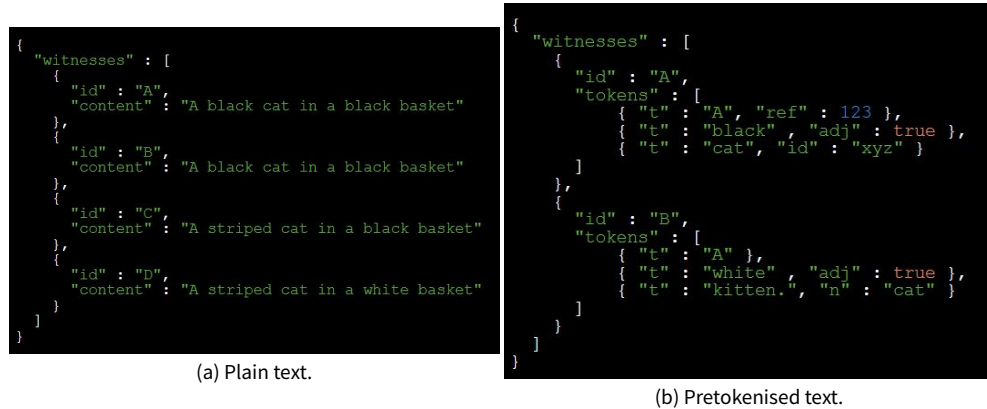


Figure 7.2: CollateX JSON input format — plain text vs. pretokenised text.

tions needed to be transformed into CollateX JSON input format, with the help of eXtensible Stylesheet Language Transformations (XSLT). This transformation process is described in the next section.

7.1.3 From XML to JSON

In this section I will first describe the desired JSON input for CollateX, and then the XSLT file which takes the transcriptions and transforms them into JSON for collation. I have chosen several token properties to include in the JSON input format. Some of these properties are present for every token, namely:

- Original token (t): this is the actual token that will be used by CollateX to align the witnesses, unless a normalised form is also supplied. It is a mandatory property of CollateX JSON tokens.
- Declamation number (decl): I have included the declamation number as an identification property, in order to be able to study the collation results of one declamation at a time. However, I have not used this property in the final visualisation (see Chapter 8).
- Location in manuscript (locus): I have included the location of each token in the form of folio or page number followed by the line number, in order to easily find a token back in the transcription file or in the manuscript facsimile. This property is particularly useful to correct errors that appear in the collation or to check uncertain passages directly in the facsimiles.

Other token properties are present only in a limited number of tokens and are optional, namely:

- Normalised form (n): a normalised form of a token. When present, it is used by CollateX algorithm instead of the original form (t) to align the witnesses.
- Editorial Note (note): these are comments that appear in the transcriptions and which I expected to need while analysing the collation results.
- Link to a digital facsimile (link): when possible, I have included a link to a digital facsimile available online.

In summary, the token objects contain all the information from the transcription that I would like to see in the collation results. The tokens and their properties are then translated into JSON objects.

7.1.3.1 Tokens - JSON objects

An object is composed of pairs of name and value, separated by a comma and enclosed in curly brackets. I will also refer to these pairs as 'properties'. Tokens are represented as JSON objects, and the pairs of name and values are the properties of tokens described above. Here is an example of a token in JSON format:

```
{'t' : 'uxor', 'decl' : '1', 'locus' : '244r:3'}
```

This is a token in the form of a JSON object that contains three pairs of name and value. First, the name 't' has the value of 'uxor', the word as it appears originally in the manuscript. The name 'decl' has a value of '1', which means that this word is part of the first declamation of Calpurnius. Finally, the name 'locus' has a value of '244r:3', which describes the location of the token in the manuscript N: the token is located on the third line of folio 244r. These three items, original word (t), declamation number (decl) and location of the token (locus) are always present in a token object. Their order within the curly braces is not important. The other optional properties can also be present in a token as pairs of name and values, such as a normalised form (n), an editorial note (note) and a link to a digital facsimile (link). Gaps and lacunae are rendered as special tokens with an original form of '...' and a normalised form 'lacuna'. When possible, a note mentions the extent of the missing text as well as the reason why the text is missing.

Here are a few examples of tokens to illustrate how their properties are encoded in the JSON notation:

1. Normalised form (n)

{'t' : 'foemina', 'n' : 'femina', 'decl' : '1', 'locus' : '244r:11'} (Manuscript N).

2. Editorial note (note)

{'t' : 'OO', 'n' : 'contra', 'note' : 'Unknown abbreviation. Normalised form supplied by Lehnert.', 'decl' : '3', 'locus' : '82r:18'} (Manuscript C).

3. Link to facsimile (link)

{'t' : 'tyranno', 'link' : 'http://daten.digital-e-sammlungen.de/bsb00090859/image_294', 'decl' : '1', 'locus' : '147r:23'} (Manuscript B).

4. Lacunae and gaps

{'t' : '...', 'n' : 'lacuna', 'note' : 'lacuna (gap) of 1 word. Reason: illegible.', 'decl' : '30', 'locus' : '14v:24'} (Manuscript M).

{'t' : '...', 'n' : 'lacuna', 'note' : 'lacuna (space) of 1 lines.', 'decl' : '4', 'locus' : '82r:29'} (Manuscript C).

7.1.3.2 Witnesses - JSON arrays

Besides objects, the second data structure in JSON is an array: an ordered list of objects enclosed in square brackets. In the case of the collation input, the text of a witness is an array containing an ordered list of tokens:

```
[{token 1}, {token 2}, ... {token n}]
```

Contrary to the order of properties within the token object, the order of tokens within the array is important, because it follows the word order of the text. If the word order is lost, it is not possible to collate anymore. Objects and arrays can be nested to form complex structures, such as a witness. A witness is an object with two pairs of name/value, a siglum (id) and an array of tokens (tokens):

```
{
'id' : 'B1',
'tokens' : [token 1, token 2, ... token n]
}
```

As we have seen in figure 7.2 above, the CollateX JSON input requires an object with the name 'witnesses' and a value which is an array that contains a list of witness objects:

```
{'witnesses' : [wit-1, wit-2, wit-n]}
```

In summary, the JSON input for CollateX is a complex data structure of nested

objects and arrays: the top object in the structure has a name ‘witnesses’ and a value of an array (list) of witness objects. A witness object has two properties: an ‘id’ for its siglum, and a ‘token’ property which is an array of tokens. And finally, a token object has several properties such as ‘t’, ‘n’, and so on.



I will now describe the transformation of the XML transcriptions into JSON, with the file `witnesses-to-json.xsl` (see Appendix B.2). The transcription XML format (see the previous Chapter 6) and CollateX JSON input format described above are quite different, which means that the transformation from the former into the latter is a complex task. In order to simplify the process, the XSLT transformation is done in two phases. During the first phase, the XML transcription is converted to another XML, closer to the JSON format needed as a collation input. In the second phase, this XML is transformed into the JSON format of witnesses and tokens. Here is an example of the new XML format for one witness:

```
<witness siglum='B1'>
  <token>...</token>
  <token>...</token>
  ...
</witness>
```

Once the witnesses are transformed into this new XML format, it becomes easier to transform this XML into the proper JSON input format for CollateX.

7.1.3.3 Witnesses

To obtain this new XML format, the XSLT transformation first checks how many witnesses must be created from one transcription file. This information is located in the `handDesc` of the TEI Header: for each `handNote` element, a new witness is created. The witness siglum, which is the `siglum @xml:id` attribute of `msIdentifier`, will serve as the ‘id’ property of the JSON witness object. The siglum is combined with the `handNote` number if there is more than one hand. For instance there are two `handNote` elements in manuscript B, therefore there will be two witnesses created (B1 for the first `handNote` and B2 for the second `handNote`). If there is only one hand, as for the editors Håkanson and Pithoeus, the value of the siglum is the `msIdentifier` only, respectively LH and P1594.

For each witness created, the XSLT transformation goes through the complete transcription file and select the word elements `<w>` which are written by the

```

<xsl:template match="tei:del" mode="wit attested normal">
  <xsl:param name="phand" tunnel="yes"/>
  <xsl:variable name="mss-hand">
    <xsl:choose>
      <!-- when there is no @hand attribute,
           the deleted text is written by h1 -->
      <xsl:when test="not(@hand)">h1</xsl:when>
      <!-- otherwise by h2 -->
      <xsl:otherwise>h2</xsl:otherwise>
    </xsl:choose>
  </xsl:variable>
  <!-- test if the hand who wrote the deleted text (mss-hand)
       is the hand that is tokenised (phand) -->
  <xsl:if test="$phand = $mss-hand">
    <xsl:apply-templates mode="#current"/>
  </xsl:if>
</xsl:template>

```

Figure 7.3: Sample 1 from the XSLT transformation: dealing with `` elements (Appendix B.2).

hand that is currently transformed. If the first hand of a manuscript is currently tokenised, for instance, the words added by a second hand will be ignored, but the words deleted by the second hand are included. Based on the encoding of scribal corrections described in Section 6.2.3.3, it means in practice that the `` elements with no attribute are included in the text of the first hand, but ignored for the text of the second hand. On the other hand, the `<add>` elements with a `hand` attribute of `'#h2'` are ignored for the text of the first hand, but included in the text of the second hand. The few mistakes of a first hand, encoded with `` elements of `type='corrigendum'`, are ignored, while the one `` element in manuscript N with a `hand` attribute `h2` is included in the text of the second hand (see Section 6.2.3.3 for more detail about this particular deletion).

Figure 7.3 shows an example of a template from the XSLT transformation file for dealing with scribal deletions. When witnesses are tokenised, a variable that holds the hand number (`h1` or `h2`) is created and passed as a parameter `'phand'` when the templates are applied. When the transformation encounters a `` element in the transcription, a variable `'mss-hand'` is created: it will have a value of `'h1'` if there is no attribute to the `` element, and otherwise it will have a value of `'h2'`. The value corresponds to the hand which wrote the text in the `` element. If the value of the `'mss-hand'` variable corresponds to the hand that is tokenised (the `'phand'` parameter), then the text inside that `` element is processed to be included in the tokens. Otherwise, the text inside the `` element is ignored.

7.1.3.4 Tokens

Tokens are created as the word elements `<w>` are processed. Here is the structure of the most complete `<token>` elements in the new XML format that is created,

when every property is included:

```
<token folio='xxx' line='yyy' decl='zzz'>
  <t>original word</t>
  <n>regularised word</n>
  <note>information from the transcription</note>
  <link>link to facsimile</link>
</token>
```

For each `<w>` element, a `<token>` is created with three attributes: `folio`, `line`, and `decl`. These attributes correspond to the two token properties 'locus' and 'decl' described above. If the `<w>` element in the transcription contains a `<choice>` element, the word is first transformed with the content of `<orig>`, in order to obtain the 't' property of a token, and then it is transformed again with `<reg>` to obtain the normalised form 'n'. If the preceding `<pb>` element has a `fac` attribute with a link to a digital facsimile, this is encoded in `<link>`. The note element in the new XML is the most complex to create, because the content of a note may be encoded in various places of the transcription. An editorial note maybe encoded in a `<note>` element in a word `<w>`. However, the content of a note may also be found in other places, such as the `reason` attribute of an `<unclear>` element, or the `resp` attribute of `<reg>` or `<supplied>` elements.

Figure 7.4 shows the four templates that help to deal with original and regularised forms of a token, when the `<choice>` element is present in a word `<w>`. There are two modes, which makes it possible to choose the right template to apply. In order to create the `<t>` property of a `<token>`, only the templates with the mode 'attested' are applied (for the form of the word which is attested in the manuscript). In order to create the `<n>` property, only the templates with the mode 'normal' are applied. In practice, it means that the `<orig>` element is ignored in 'normal' mode, but processed for the 'attested' mode. On the other hand, the `<reg>` element is ignored in 'attested' mode, but processed in 'normal' mode.

Since there are differences in the encoding of manuscripts and modern editions (see Section 6.2.5), the XSLT transformation will also be slightly different. This is done with modes again: the templates with 'mss' mode are applied to manuscripts and to the old edition of Pithoeus, whereas the templates with 'edd' mode are applied to modern editors (here it applies only to Håkanson, but it would also apply for other editors such as Lehnert or Sussmann if their texts were to be added).

There are TEI elements which are absent from the modern editions, such as ``

```

<xsl:template match="tei:choice/tei:orig" mode="wit normal"/>

<xsl:template match="tei:choice/tei:orig" mode="wit attested">
  <xsl:param name="phand" tunnel="yes"/>
  <xsl:apply-templates mode="#current">
    <xsl:with-param name="phand"/>
  </xsl:apply-templates>
</xsl:template>

<xsl:template match="tei:choice/tei:reg" mode="wit attested"/>

<xsl:template match="tei:choice/tei:reg" mode="wit normal">
  <xsl:param name="phand" tunnel="yes"/>
  <xsl:apply-templates mode="#current">
    <xsl:with-param name="phand"/>
  </xsl:apply-templates>
</xsl:template>

```

Figure 7.4: Sample 2 from the XSLT transformation: dealing with original `<orig>` and normalised `<reg>` form of a token (Appendix B.2).

```

<xsl:template match="witness">
  <!-- Open JSON object for witness -->
  <xsl:text/>{&#xA;"id" : "<xsl:value-of select="@id"/>",&#xA;"tokens" : [&#xA;<xsl:text/><!-- transform tokens into JSON -->
  <xsl:apply-templates/>
  <!-- close JSON witness -->
  <xsl:text>]&#xA;}</xsl:text>
  <!-- if it is not the last witness, add comma (following-sibling::witness)-->
  <xsl:if test="position() != last()"><xsl:text>,</xsl:text></xsl:if>
  <xsl:text>&#xA;</xsl:text>
</xsl:template>

```

Figure 7.5: Sample 3 from the XSLT transformation: witness template (Appendix B.2).

and `<add>`, but their equivalent `<surplus>` and `<supplied>` must be processed. The `<w>` elements for tokens are also treated differently within modern editions because some properties are obtained from different elements. For instance, the link to the digital facsimile of Håkanson's edition points to the Packard Humanities text, where the edition presents one declamation on each webpage. Therefore, the `<link>` property is the `facts` attribute of `<ab>` and not of `<pb>`. The content of notes is found in the `ana` attribute of `<sic>`, to indicate words that were considered part of a *locus desperatus* by the editor and marked with a *crux*.

7.1.3.5 Final steps: JSON format

When the witnesses have been tokenised in the intermediary XML format, they can be transformed into JSON with another set of two templates: one template transforms a witness into JSON, and the second template transforms tokens into JSON objects. Figure 7.5 shows the template that transform a witness into a JSON object.

The template dealing with tokens is more complex because it contains another series of normalisations. The orthographical normalisations were directly encoded in the transcription files with the help of a choice between `<orig>` and `<reg>` elements, however the transformation into JSON adds more normalisation, namely:

- Replacing the symbol & with ‘et’;
- Replacing accented letters with their unaccented counterparts: for instance, é, è, ê, and ë are all replaced with the letter ‘e’⁸;
- Removing non-literal symbols such as punctuation, whitespaces, square and angle brackets, cruces symbols (†);
- Transforming all uppercase letters into lowercase.

All witnesses are then gathered in one single document with XInclude⁹, a mechanism which merges XML documents automatically, and a transformation scenario is applied to the master file so that each transcription file is processed by the XSLT described above (see Appendix B.2).

7.1.4 CollateX Output Formats

CollateX’s command line offers various output formats: there is a JSON output, two XML outputs, and two outputs in a graph format (dot and graphml), all of which can be found in CollateX Documentation. The two XML outputs are either a TEI XML output with `<app>` elements to encode variants, and an XML format specific to CollateX that records the collation results as a collation table with rows and cells (see Section 8.2). The CollateX XML format, although shown in CollateX Documentation, could not be obtained from the command line interface. A CSV output format is also proposed as an option in the command line. Similar output options are available from the other interfaces: in addition to JSON and XML, one more graph format SVG is available from the RESTful web service, and an HTML table output is available from the Python version. The main issue related to CollateX output is the fact that it cannot be reused as an input again, for instance to add a new witness to an already collated set of witnesses.

The choice of an output format depends mainly on how the collation results will be further processed, but also on the input format. For instance, not all outputs will

⁸Accented letters appear especially in Pithoeus’ edition, and in manuscripts M and N as well.

⁹<https://www.w3.org/TR/xinclude/> (Accessed August 28, 2017).

return results which keep the additional token properties that are included in the JSON input: only the JSON output will return results where each token still has its properties attached to it (normalised form, note, link, location and declamation number).

The CSV, TEI, and dot formats provide results which only include the original form (t) of a token, and the other properties are then lost. It should also be noted that the pretokenised JSON input cannot be combined with output formats other than JSON in some circumstances. For instance, the combination of a JSON input and a TEI output can have surprising consequences: it shows the text of the tokens all combined together with no whitespace separation (see figure 7.6(a)). The reason for this issue comes from a conjunction of two aspects: the first is that the pretokenised JSON input encodes only the letters which make up the words of the text, and not the white space separations. The second aspect is related to CollateX's output. The output can either join together matching tokens into segments, or keep tokens separated. In the example of figure 7.6(a), the text '*Incipiunt ex Calpurnio Flacco excerptae*' is a chunk of text which is matching together in the manuscripts and Pithoeus, but is absent from Håkanson's edition. Therefore this chunk of text is combined into one reading element <rdg>. However, by requesting that CollateX keep each token separated (with the parameter `-tokenised` or `-t`), the results will not combine the tokens together and the output will be more accurate (see figure 7.6(b)).

To examine the collation results, I have chosen the JSON output of CollateX, because it is the most complete output format that contains all token properties, and because it is a format that is easily converted into data usable with other programming languages (as opposed for instance to the XML output which would require another XSLT transformation) (see Section 8.4.1). The analysis of CollateX's results will be discussed more in detail in Section 8.5.

7.1.5 Discussion

CollateX interfaces require more technical knowledge than Juxta or the CTE: at least to be comfortable using the command line, or more advanced programming capacities in Java or Python. There are multiple input and output formats, which are not always well combined. The pretokenised JSON input, for instance, should be combined with a JSON output, or a separated tokens output. The pretokenised JSON input necessitates more preparation than the others, but it provides advantages later in the visualisation of the results (Chapter 8). Although the collation algorithm is the same across the various interfaces, there are minor differences in

```

<app>
  <rdg wit="B1 C1 B2 C2">IncipiuntexCalpurnioFlaccoexcerptę</rdg>
  <rdg wit="LH"/>
  <rdg wit="M2 N1 M1 N2 P1594">IncipiuntexCalpurnioFlaccoexcerptę</rdg>
</app>

```

(a) Joint tokens.

```

<app>
  <rdg wit="M2 N1 B1 M1 C1 N2 B2 C2 P1594">Incipiunt</rdg>
  <rdg wit="LH"/>
</app>
<app>
  <rdg wit="M2 N1 P1594 B1 M1 C1 N2 B2 C2">ex</rdg>
  <rdg wit="LH"/>
</app>
<app>
  <rdg wit="M2 N1 B1 M1 C1 N2 B2 C2 P1594">Calpurnio</rdg>
  <rdg wit="LH"/>
</app>
<app>
  <rdg wit="M2 N1 B1 M1 C1 N2 B2 C2 P1594">Flacco</rdg>
  <rdg wit="LH"/>
</app>
<app>
  <rdg wit="B1 C1 B2 C2">excerptę</rdg>
  <rdg wit="LH"/>
  <rdg wit="M2 N1 P1594 M1 N2">excerptę</rdg>
</app>

```

(b) Separated tokens.

Figure 7.6: CollateX JSON input and XML TEI output (Command Line interface).

the level of control users have over the collation process. In the Python version, there are more parameters available, and in the RESTful web service there is the possibility to customise the token equivalence function. In conclusion, the choice between the three interfaces depends on factors such as the technical knowledge of CollateX's users, the editing workflow or the purpose of the collation.

7.2 Juxta

Collating with Juxta is a very different experience from using CollateX. Whereas CollateX requires users to have at least a small amount of prior computing knowledge, at least being comfortable with the command line, Juxta offers an intuitive Graphical User Interface (GUI). In fact, there are three different user interfaces: a desktop application, and two web interfaces which are powered by the Juxta Web Service API. Juxta proceeds similarly in each interface. The user uploads documents to Juxta, which are transformed into witnesses, i.e., the documents are tokenised. Then the user can form a comparison set by selecting witnesses to be compared. Once the witnesses in the comparison set are collated, the user can visualise the results in different ways. The following section examines the three interfaces and their differences.

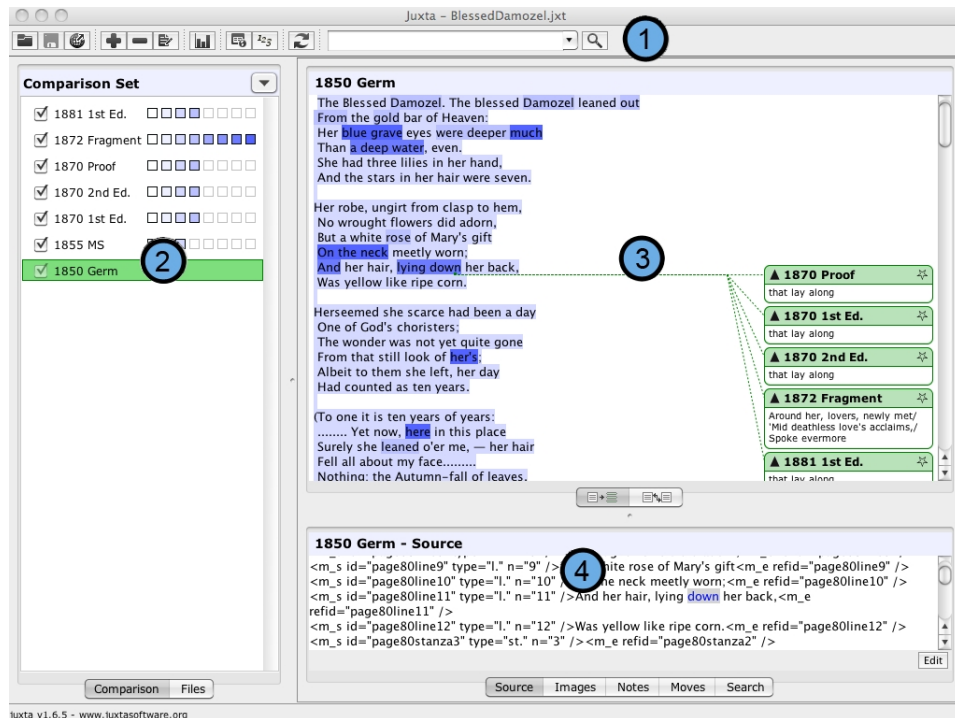


Figure 7.7: Juxta Desktop interface. Retrieved from <https://github.com/performant-software/juxta-desktop/wiki/UserInterface> (August 29, 2017).

7.2.1 Juxta Interfaces

7.2.1.1 Juxta Desktop Application

Juxta Desktop is a legacy version of the software, which can be installed on any operating system (Windows, Macintosh and Linux)¹⁰. Although the web interfaces are more recent, the desktop application may still be a good solution for textual collation. A detailed user manual is available on Github¹¹. The version that was used during this research is version 1.7.0. Figure 7.7 shows what the interface looks like: there are four panels that contain (1) a toolbar, (2) the list of witnesses in a comparison set, (3) the collation visualisation and (4) a secondary panel in the bottom half which can display the source documents. A comparison set contains the witness transcriptions which are all collated by default, unless otherwise specified.

Juxta Desktop accepts input in plain text or in XML. There are prepared templates

¹⁰Macintosh users with OS 10.7 or higher may encounter an installation issue. See the download page here: <http://www.juxtasoftware.org/download/> (Accessed August 29, 2017).

¹¹<https://github.com/performant-software/juxta-desktop/wiki/OnlineDocumentation> (Accessed August 29, 2017).

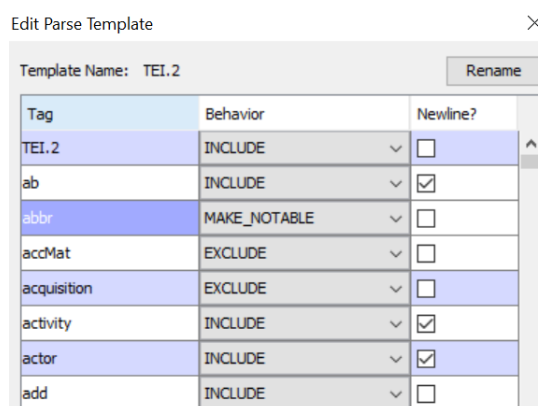


Figure 7.8: Juxta XML parsing template (Juxta Desktop).

to parse XML from TEI documents or Juxta documents (with the extension .jxt), but users may also create their own templates in order to parse texts encoded in other XML formats. It is also possible to influence how Juxta transforms a document into a witness by editing the parsing templates (see figure 7.8). The dialogue box to edit the XML parsing templates offers several options to change Juxta's behaviour. Juxta can either INCLUDE, EXCLUDE or MAKE_NOTABLE each XML tag. Including a tag means that its content will be collated, provided that all other tags in the XML hierarchy are also included. For instance, to collate the text of the `<title>` in the `<titleStmnt>` of the `<fileDesc>` in the `<teiHeader>`, all of these elements must be marked as included. The `<note>` elements, if included, are not collated but appear as comments in the right margin. On the other hand, excluding a tag means that its content and the content of its children elements will be ignored for collation purposes, and will not be displayed in the collation visualisations. Finally, MAKE_NOTABLE makes it possible to see changes in markup encoding as variants. Assuming that the tag `` indicates bold text and `<i>` italic text, if both tags are made notable, then the following words will appear as variants in the collation results: `word` and `<i>word</i>`. However, this feature is not extended to attributes, so that it is not possible to see the following as variants: `<hi rend='bold'>word</hi>` and `<hi rend='italics'>word</hi>` (see also Section 3.3.3). The last column 'Newline?' adds a line break after the content of checked tags.

A dialog lets users decide if they want to change some settings in the collation algorithm in order to filter out whitespace, punctuation or case from the collation results. After the collation has been performed, users can analyse and comment on the results. Individual variants can be annotated, and moved text (transpositions) can be indicated. In the bottom half of the screen, the transcription of the source

document is displayed: it can be corrected by users directly in Juxta, and then saved as a new document. The corrections will then be automatically incorporated in the collation. The corresponding text in the source document is very easily accessible by clicking on the text of the collation results, which is an advantage of Juxta Desktop. In addition, there is also a search text option, the possibility to change the font, and to display location marks such as line breaks and page breaks. The option to ‘toggle the revision view’ allows for dealing with `<add>` and `` elements in the markup. By default, Juxta ignores additions and deletions marked with those tags (Juxta Manual), but they can be incorporated to the collation either in a case-by-case basis, or in bulk. Finally, it is possible to associate digital images of the source to a transcription.

The visualisations available in Juxta Desktop include the heat map, two texts side-by-side, and the histogram described in Section 2.6.4. It is also possible to generate a critical apparatus in the form of an HTML file. The critical apparatus does not include a base text alongside, which makes it difficult to work with outside of Juxta. The collation results can be saved in the Juxta file format, or exported as a comparison set to the web interface of Juxta Commons, where it can then be made publicly visible and be shared with others.

7.2.1.2 Juxta Commons

Juxta Commons takes the functionalities of the desktop application, and makes them available through a web interface that is accessed from a browser¹². A short user guide is also available for Juxta Commons¹³. Although the two interfaces are very similar, there are some differences between Juxta Desktop and Juxta Commons, which will be noted within the description of Juxta Commons. The interface contains an upper panel with three sections: sources, witnesses, and comparison sets (see figure 7.9). There is no way to organise sources or witnesses in groups of texts for the different comparison sets, so that it may become confusing for users with many sources and witnesses. The best solution is to have a consistent name scheme, in order to sort the documents in a convenient way.

The bottom panel changes according to the document selected. For sources, it is possible to edit and save the content of the document. For witnesses, it is possible to either view the plain text content, with the option to visualise page and line breaks, or to view the XML content in the case of XML sources. In this XML view, users can select tags to include or exclude from the collation. However, the `MAKE_NOTABLE`

¹²<http://juxtacommons.org> (Accessed August 29, 2017).

¹³<http://juxtacommons.org/guide> (Accessed August 29, 2017).



Figure 7.9: Juxta Commons interface. Retrieved from my personal account (August 29, 2017).

option is not available in Juxta Commons. For comparison sets, the bottom panel displays the collation results and offers various visualisations.

While Juxta Desktop accepts only plain text and XML as input, Juxta Commons can also take HTML files, Microsoft Word DOCX and Open Office documents, as well as EPUB and PDF files which contain text (and not only images). Variant annotations were only local in Juxta Desktop, i.e., attached to a single variant reading. Juxta Commons allows for ‘regional’ annotations as well, that is annotations which apply to a whole variant location. Again, it is possible to ignore punctuation and/or capitalisation, and set the desired hyphenation sensitivity. Kingsley (2014) remarked that these settings still give room for imprecisions, for instance if the user wants to view punctuation but is not interested in differences of spacing around punctuation marks.

Visualisations in Juxta Commons include three more options, in addition to the heat map, the side-by-side view of two witnesses and the histogram: the edition starter kit, the Versioning Machine, and an XML TEI output. The edition starter kit creates a basis for a critical edition, with both a text and a critical apparatus. The apparatus of the edition starter is very much identical to the Classical Text Editor (see Section 7.3.3 below). The CTE, however, has much more options for typesetting the edition; consequently, if a traditional edition with a base text with and a critical apparatus is the final goal, it is probably worth using the CTE instead of Juxta.

Thanks to the integration of the Versioning Machine visualisation, it is also possible to view multiple witnesses side-by-side instead of only two. When clicking on a

Witness 1: wit-26307	Witness 5: wit-26303	Witness 9: wit-26299
<p>incipiunt ex calpurnio flacco excerpte excerpta decem rhetorum minorum</p> <p>uxor tyrannida quinque cum tyranno proxime familie puniantur quedam que habebat duos filios et tyrannum virum tyrannidum fecit praemio impunitatem liberis postulavit meruit ex his alter occupavit arcem at cum mater occidit petit alteri impunitatem contradicitur occidam inquit quanta</p>	<p>incipiunt ex calpurnio flacco excerptae</p> <p>uxor tyrannida quinque cum tyranno proximae familiae puniantur quaedam quae habebat duos filios et tyrannum virum tyrannidum fecit praemio impunitatem liberis postulavit meruit ex his alter occupavit arcem at eum mater occidit petit alteri impunitatem contradicitur contra matrem occidam inquit quanta nobis patiendi sunt dum occiderit</p>	<p>incipiunt ex calpurnio flacco excerptae x rhetorum minorum</p> <p>uxor tyrannida quinque cum tyranno proximae familiae puniantur Quae habebat duos filios & tyrannum virum tyrannidum fecit praemio impunitatem liberis postulavit meruit ex his alter occupavit arcem eum mater occidit petit alteri impunitatem contradicitur contra matrem occidam</p>

Figure 7.10: Juxta Commons, Versioning Machine visualisation.

variant in a witness, it will highlight the parallel variant readings in the text of every witnesses.

Finally, it is possible to export the collation results in the TEI XML format with variant readings encoded according to the parallel-segmentation method. This output works best for plain text or basic XML encoding (User Guide). Original markup of the source's transcription may not be kept in the TEI XML output. In fact, most of my original markup disappeared with the exception of linebreaks: for instance, elements such as `<hi>` for highlighted text or `<ex>` for abbreviations were not included in the TEI output.

A review of the TEI output from Juxta Commons shows that there are a few errors in the alignment, which are not obvious directly from the other visualisations such as the Heat Map. While variant locations are correctly highlighted in blue, the alignment of readings within a variant location is sometimes incorrect, or at any rate not satisfying for the editor. There are mistakes of alignment such as one reading *excerpta* from the incipit aligned with the reading *X*, although *excerpta* should be aligned with *excerpta* in other witnesses and *X* should be aligned with *decem*. Differences in word division may become an issue, especially if there is another variant nearby (see also Section 7.3.3 below for word division issues with CTE). For instance, the readings *verbera cibus* and *verberantibus* are correctly aligned; but the variant *nescioquid/nescio quid* immediately followed by the variant *diu/adhuc* gives a mistake: one reading *diu* is not correctly aligned (see figure 7.11). These are issues of alignment for the regularised texts. If the original tokens are collated, the chance of having several variants in a row increases, and the mistakes or imprecisions increase in the collation results.

These errors in the alignment can illustrate how the order in which witnesses are collated can influence the results, in combination with the heuristic method. For instance in figure 7.11, the witnesses are collated against LH (wit-28535) which reads *nescio quid adhuc*. The next witness P1594 (wit-28536) reads *nescioquid diu*, which does not match exactly the text of LH, and therefore the heuristic method

```

<app>
  <rdg wit="#wit-28535 #wit-28532 #wit-28534 #wit-28533 #wit-28530 #wit-28531 #wit-28528 #wit-28529 #wit-28527">nescio
  quid</rdg>
  <rdg wit="#wit-28536">nescioquid diu</rdg>
</app>
<app>
  <rdg wit="#wit-28535">adhuc</rdg>
  <rdg wit="#wit-28527">diu</rdg>
  <rdg wit="#wit-28536"></rdg>
  <rdg wit="#wit-28534">diu</rdg>
  <rdg wit="#wit-28533">diu</rdg>
  <rdg wit="#wit-28533">diu</rdg>
  <rdg wit="#wit-28531">diu</rdg>
  <rdg wit="#wit-28530">diu</rdg>
  <rdg wit="#wit-28529">diu</rdg>

```

Figure 7.11: Example of erroneous alignment in Juxta Commons.

produces the following alignment:

LH	nescio	quid	adhuc
P1594	nescioquid	diu	

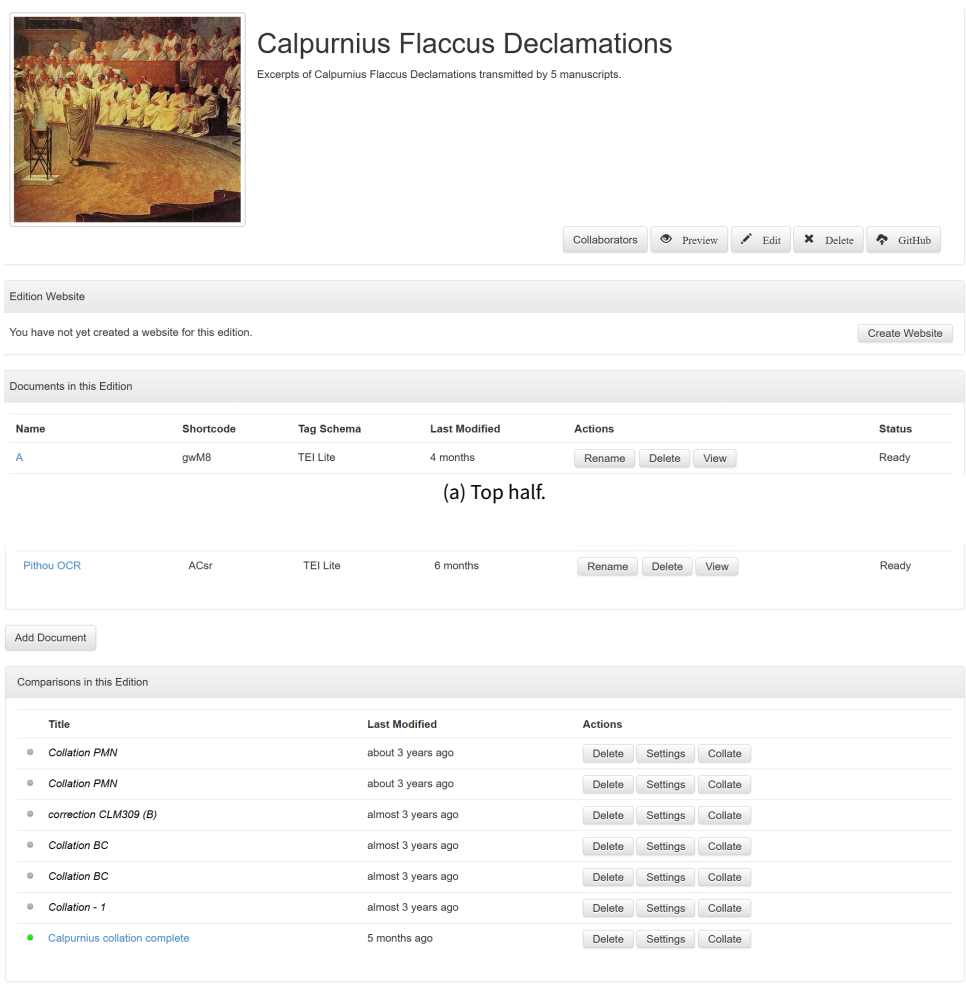
The witnesses subsequently collated read *nescio quid diu*, and therefore the readings *diu* are properly aligned with *adhuc*, and P1594 is the only witness for which *diu* is not correctly placed in the results.

Juxta Commons is advantageous for sharing the collation results on the web, or having more options of input formats and visualisations. However, Juxta Desktop may be better with regard to other aspects. For instance, Juxta Commons often takes a long time to collate, while collation happens instantly in Juxta Desktop. The desktop application also makes it easier to correct transcription errors spotted in the collation results because the transcription appears below the collation and is linked to it. In Juxta Commons, it is necessary to open the source document in a new browser window (Kingsley 2014). Finally, the option to include source images in Juxta Desktop is not available in Juxta Commons.

7.2.1.3 Juxta Editions

The latest interface for Juxta was released in 2015. It aims at providing a professional suite of tools for born digital critical editions, covering the entire editing workflow from transcription to publication. As a result, this new interface brings a bigger change than the transition between Juxta Desktop and Commons. The new transcription interface HumEdit is introduced, and free or paid subscription plans offer different ranges of options. The documentation, however, is limited to the ‘features’ page and a series of eight short screencast videos which become available once logged into a user account¹⁴. Some information is missing, such as the number of witnesses which can be collated.

¹⁴<http://www.juxtaeditions.com/features> (Accessed August 30, 2017).



Calpurnius Flaccus Declamations
Excerpts of Calpurnius Flaccus Declamations transmitted by 5 manuscripts.

Collaborators Preview Edit Delete GitHub

Edition Website
You have not yet created a website for this edition. [Create Website](#)

Documents in this Edition

Name	Shortcode	Tag Schema	Last Modified	Actions	Status
A	gwM8	TEI Lite	4 months	Rename Delete View	Ready

(a) Top half.

[Pithou OCR](#) ACsr TEI Lite 6 months [Rename](#) [Delete](#) [View](#) Ready

[Add Document](#)

Comparisons in this Edition

Title	Last Modified	Actions
• <i>Collation PMN</i>	about 3 years ago	Delete Settings Collate
• <i>Collation PMN</i>	about 3 years ago	Delete Settings Collate
• <i>correction CLM309 (B)</i>	almost 3 years ago	Delete Settings Collate
• <i>Collation BC</i>	almost 3 years ago	Delete Settings Collate
• <i>Collation BC</i>	almost 3 years ago	Delete Settings Collate
• <i>Collation - 1</i>	almost 3 years ago	Delete Settings Collate
• <i>Calpurnius collation complete</i>	5 months ago	Delete Settings Collate

(b) Bottom half.

Figure 7.12: The interface of Juxta Editions, edition page — top half and bottom half. Retrieved from my personal account (August 30, 2017).

The interface is considerably different from the previous versions of Juxta. The purpose of Juxta Editions is explicitly to create scholarly editions, and this is reflected in the presentation of the interface: the homepage is a ‘desk’ where users can access ‘editions’, which are limited in number from one (free account) to ten (premium account). The edition contains a list of documents that are the witnesses, and a list of comparisons corresponding to the Comparison sets.

In Juxta Editions, it is possible to transcribe documents directly in the HumEdit transcription interface (see also Section 6.1.4). The transcriptions are done either in plain text, or with XML TEI Lite tagging. It is also possible to upload XML files,

however there is no indication in the documentation regarding the accepted XML format. It is likely that only TEI Lite documents can be uploaded, because after uploading one of Calpurnius transcription files, the result was rather deceiving: the content of the word elements `<w>` were not separated by white spaces for instance, even if the white spaces are actually present in the transcription file in between `<w>` elements. On the other hand, `<add>`, `` and `<unclear>` elements were properly recognised and colour coded in blue, red and grey respectively. Juxta Editions also provides an integrated Optical Character Recognition (OCR) solution for printed pages with ABBYY FineReader, a commercial OCR software (see Section 2.6.2 for an example of Juxta Editions OCR with Pithoeus' edition). Images of the source can also be uploaded, and displayed side-by-side with the transcription.

Visualisations in Juxta Editions are limited to the heat map and the side-by-side view (at least in the free account). There is no support to choose which XML tags should be included or excluded from collation, contrary to Juxta Desktop and Commons. Among other features specific to Juxta Editions are collaboration and web publishing, both requiring a subscription. Although collaboration is not free, one single paying account can invite any user to collaborate on a project. *Blessed Damozel* is an example of an edition prepared with Juxta Edition¹⁵. However, this website does not show any collation results, so that it is difficult to evaluate the editions created with this technology. There are few other examples of editions online: The Melville Electronic Library has published two editions of Melville that integrate with Juxta Editions, “Versions of Moby Dick: A Fluid-Text Edition” (2018), and “Versions of Billy Budd: A Fluid-Text Edition” (2018). In 2017, both editions were showing the side-by-side view of two witnesses, which is also available in the other versions of Juxta. However this will probably evolve in the future, as the full integration with Juxta Editions should be available in the Summer of 2018¹⁶.

According to the video number 8, available to logged users, new features will be added to Juxta Editions in the future, such as extended support for transcribing correspondence, poetry and drama, customised collation, generating RDF metadata, and institutional subscriptions.

7.2.2 Discussion

The comparison of the three different Juxta interfaces helps to understand the benefits or drawbacks of each interface. Although the tools share a name, and

¹⁵<http://sites.juxtaeditions.com/BlessedDamozel/> (Accessed August 30, 2017).

¹⁶<https://mel.hofstra.edu/versions-of-moby-dick.html> (Accessed February 20, 2018).

apply the same collation algorithm, the differences between the three may be important enough as to influence the choice of a scholar who wishes to use Juxta for a collation project. Juxta Desktop is much quicker to collate, and makes it easier to switch between collation and transcription, to correct mistakes for instance. Juxta Commons offers more choices of input and output formats, especially a TEI XML compliant output which can be reused outside Juxta. Juxta Editions, on the other hand, is more restrictive in terms of input formats and visualisations available. Juxta Editions has no particular advantage with regard to collation itself, it has no option yet for influencing the results by including or excluding XML elements. The advantages of Juxta Editions lie rather with the collaboration option and the transcription platform with source images, which allows together for crowdsourcing transcriptions. Juxta Editions also provides an environment for the entire workflow of editing, from transcription to online publication.

In the context of scholarly editing, the intuitive user interface makes Juxta an accessible tool to Humanities scholars and students, even with little or no computing knowledge. Juxta is useful in particular for transcription correction (Kingsley 2014). The blue highlighting of the heat map visualisation is helpful to locate ‘false positive’ variants, that is text falsely marked as variant because of a transcription error. The opposite ‘false negative’ variants, real variants which are not highlighted in blue, are more difficult to spot. Juxta offers also interesting collation visualisations, to display the final results of a critical edition with multiple witnesses. Otherwise, for the steps between transcription and visualisation of the final results, Juxta does not provide an environment for correcting the collation results, or choosing a lemma among the variant readings to create a critical text (Kingsley 2014). The only solution at the moment is to duplicate one of the existing witnesses, and then modify the source text of this new witness with readings selected by the editor. Another desirable feature suggested by Kingsley (2014) would be a way to categorise variants according to a typology (for instance between accidentals and substantives). The issue with such a feature would be that there are many ways to categorise variant readings, and it may be difficult to satisfy a majority of scholars (see Chapter 4). I would also argue that editors need not only to categorise readings, but more importantly to analyse the readings shared by group of witnesses, in order to study the witnesses’ relationships (see Section 8.1.2). While Juxta is an excellent tool for collation visualisation, it may not be a perfect philological tool for scholarly editors.

7.3 Classical Text Editor

In contrast to Juxta, the Classical Text Editor (CTE) was designed first as a philological tool, for the purpose of helping editors create mainly print critical editions and to a lesser extent also digital editions (Hagel 2007). The collation algorithm is a recent addition from 2015 to a project that started twenty years ago in 1997. CTE has been successfully used to publish over 170 editions, between 1999 and 2017¹⁷. It is worth examining how automated collation is integrated into a software that is already widely used for preparing critical editions of Classical texts. We will see how CTE performs automated collation, and how well it deals with orthographic differences and word division, since this is supposed to be a major advantage of the collation algorithm in CTE. Since the software is under commercial license, I have used the free thirty-day trial version.

7.3.1 Interface

The CTE is a graphical word processor explicitly created for a manual workflow of editing, and therefore it is centred on one document that contains the base text. A series of documents are associated to the base text, for various notes and references such as a critical apparatus, the list of manuscripts, an apparatus of sources (*apparatus fontium*), or editorial notes (see figure 7.13). There are many options and settings available, especially focused on setting references or cross-references between those various documents, and formatting the layout of the final output of the critical edition in PDF or in TEI XML format.

The Microsoft Word-like interface, familiar to Windows users, is meant to improve the efficiency of editors by reducing the amount of time spent in secondary tasks. Any sort of code or tagging is hidden from users to prevent distractions or confusions. Therefore the CTE requires minimal technical expertise. In addition, it provides a framework to the editorial workflow, so that a change at any step is reflected in the final output without effort.

7.3.2 Input

The CTE lets users import plain text documents from Microsoft Word or in RTF format, as well as XML documents. There are no guidelines however regarding the XML format that documents should follow in order to be imported. The transcrip-

¹⁷<http://cte.oeaw.ac.at/?id0=pub> (Accessed August 31, 2017).

7.3. Classical Text Editor

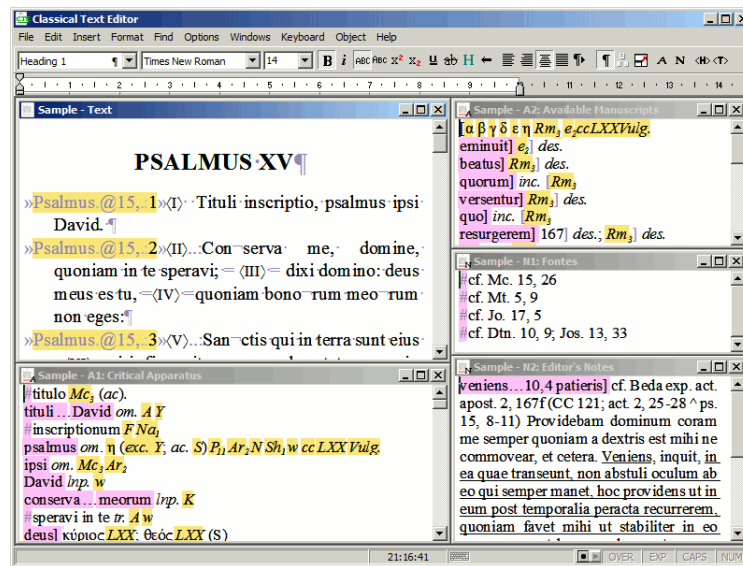


Figure 7.13: Classical Text Editor interface. Retrieved from http://cte.oeaw.ac.at/?id0%3Dsnapshots%26id1%3D0%26m0%3Dc_3 (August 31, 2017).

tion files of Calpurnius Flaccus were imported, however with only relative success¹⁸. It seems that the text of each XML element was extracted except <note> elements. Both additions and omissions alike were included, separated by semicolons, as well as original and regularised spellings without distinctions. For instance, the original word *foeliciter*, encoded with a regularised spelling *feliciter*, was imported as *foeliciter*. More problematic was the absence of whitespace separation between words. The fact that everything code-related is hidden makes it difficult to understand how the XML is processed by the CTE, and how to change the process for better results.

To test the Classical Text Editor, I have therefore prepared plain text versions of the transcription files by updating the XSLT transformation that was used to produce CollateX JSON input (see above). The witnesses thus created are in a very basic plain text format, without punctuation, line breaks or notes, for a first collation attempt. It is also possible to transcribe directly in plain text format within a CTE document, and to open images of manuscripts in a Graphic Viewer window in order to create links between text and image. However, PDF images must be transformed into an image format such as JPG to be opened in the Graphic Viewer.

¹⁸The first time the files were imported, the program ran into an error, and the developer of CTE needed to examine the files in order to understand the problem. In a second attempt (without any change to the transcriptions), the documents were imported into plain text.

7.3.3 Collation

The collation process works with a base text and a pairwise algorithm (see Chapter 2). Each witness is compared to the base text, and the results of the successive comparisons are merged into an apparatus. As we have seen in Section 2.2, the order in which the witnesses are collated can influence the results. The choice of the base text also changes the collation results: in the case of Calpurnius Flaccus, choosing Håkanson's edition as a base text resulted in the fact that the *incipit* was not collated because Håkanson did not print it in his critical text. This is a known issue of a base text comparison process, and so it must be taken into account when collating with the CTE (see Section 2.4.2).

Several steps must be followed in order to collate in CTE: the first step is to select one witness as a base text, then give a unique siglum and unique number to each witness, and finally collate the witnesses. Each of these steps is composed of a precise series of actions. Since it is not a perfectly intuitive procedure, I will list here the steps that I have been through to collate with the CTE. First, an apparatus document must be created for the witness which was chosen as a base text:

1. Open the base text witness in CTE.
2. Open the menu `Format > Document > Templates`: the template 'Apparatus 1' must be selected instead of the default value 'Text'.
3. In the same dialog box where the 'Apparatus 1' template was selected, click on `Settings`, and in the `Style` tab, check the box 'is a critical apparatus'.
4. Finally, open the menu `Windows > Apparatus`: this opens a new apparatus document.

The second step is to attribute a unique siglum and a unique manuscript number to each witness. This should be done directly from within the base text, and not by opening successively each witness:

1. Open the base text document.
2. Go to the menu `Format > Sigla` to open the dialog box where the sigla can be created.

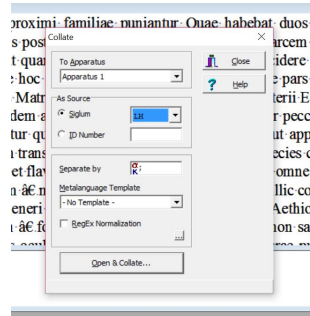


Figure 7.14: The Collation dialog box (CTE).

3. Create sigla for all the witnesses. To add a new siglum, two fields must be filled. The field 'Text' is the siglum of a witness ('LH' or 'B1' for instance) and the field 'Manuscripts' is a unique number for this witness.

After the base text was selected, and the apparatus file and the witness sigla were created, it is possible to collate. Again, this must be done from within the base text, since the program is centred on the base text document. The option to collate is in the menu File > Import > Collate. A dialog box opens and offers users to select in which apparatus the collation should be saved (see figure 7.14). Users have to select the siglum or number of the witness they wish to collate, and can then click 'open and collate' to open the CTE file corresponding to the witness that they want to collate.

It is possible to specify a file with normalisations by checking the box 'Regex Normalisation' and uploading a file. This file should contain on each line a pair of regular expressions separated by a tab. The first expression is the string of characters that needs to be replaced, and the second expression is the string of characters that should replace the first. The CTE help manual gives the examples of normalisations for the German groups of letters 'sz' and 'ß' to 'ss'. While it was indeed possible to use similar expressions to ignore orthographic variants such as *familię/familiaē*, it should be noted that this solution is not very satisfying on a large scale. In order to have an efficient normalisation, we have seen that it is not possible to simply normalise every 'ę' to 'ae', but it should be normalised case by case: *familię* is normalised to *familiaē*, but *cepit* to *coepit*, and so on. It becomes more tricky for *penę* which is used in manuscripts B and C to stand for *poenae* once, and later *paene*. For an efficient normalisation file, it means that the witnesses need to be collated first in order to spot orthographic differences, then the normalisation file must be created manually, and finally the witnesses are collated again. There is no

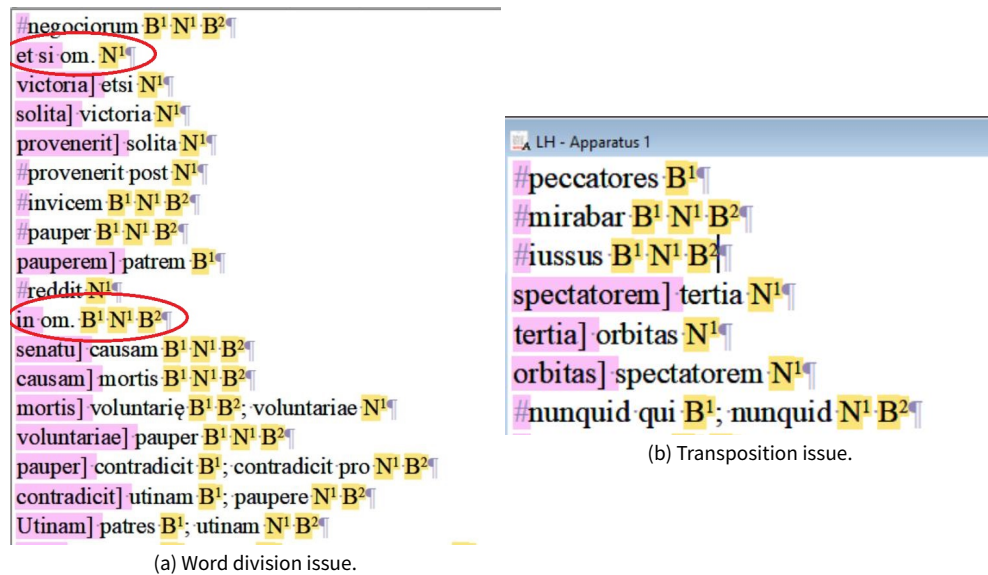


Figure 7.15: Issues of alignment with CTE algorithm.

option to ignore automatically other accidentals during collation, such as case or punctuation. In fact, it may be more practical for users to collate without normalisation, and then to hide or delete the apparatus entries which contain insignificant variants.

The collation algorithm of CTE was presented as very efficient to recognise orthographic variation and even word division difference, thanks to an implementation of the Levenshtein distance (see Section 2.2). Upon examination of the collation results, it seems that spelling differences did not cause any mistake. However, the differences in word division clearly had an important impact on the collation results. The collation extract in figure 7.15(a) shows how the variant *etsi* in witness N1 was not recognised as a variant for *et si* in Håkanson's text. As a result, the next four readings were also misaligned. Similarly a few entries below, the omission of the words *in senatu* in B2 and N1 resulted in a wrong collation (see figure 7.15(a)).

The algorithm recognised that the two-word transposition *proximi familiae* was a variant of *familiae proximae*. For more complex transpositions, the CTE manual warns that they will be treated as pairs of insertions and omissions. This was the case for the three-word transposition of *tertia orbitas spectatorem* for *spectatorem tertia orbitas*, which resulted in a meaningless apparatus entry (see figure 7.15(b)).

Hagel (2007, 78) has noted that a lack of feedback from users has limited the devel-

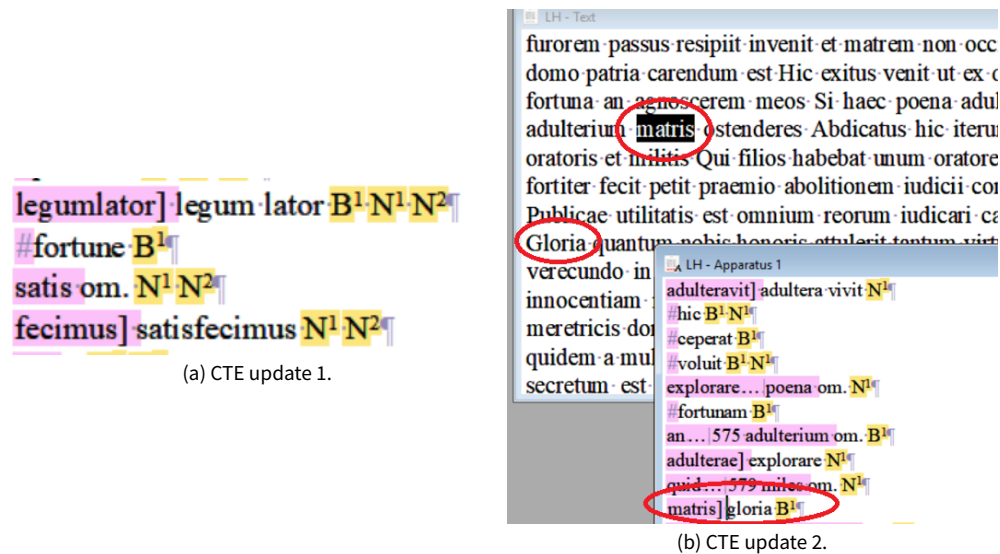


Figure 7.16: Updates to the algorithm of CTE provided by Hagel.

opment of the digital aspects of the CTE. As I shared the results of the collation with Hagel, he took notice of the issues and provided several updates to the algorithm, so as to improve the collation results. After a first update (October 4, 2017), the word division issue was partially solved, in cases when the base text had a single word aligned with two in other witnesses. For instance *legumlator* is properly aligned with *legum lator*, but *satis fecimus* was not aligned with *satisfecimus* (figure 7.16(a)).

A second update (October 5, 2017) fixed such errors of word division, but introduced new issues in the alignment due to transpositions and lacunae. For instance in figure 7.16(b), the algorithm is matching *matris* in the base text of LH with *gloria* in B1, when actually *matris* is missing from B1 due to a lacuna, and *gloria* should match the other *Gloria* four lines below in the base text.

A last update (October 6, 2017) seems to have eliminated the issues, although a few imprecisions of alignment can still be found. The algorithm, therefore, needs a lot of feedback to be fine tuned, and it may be difficult to reach a solution that will be effective in all situations. As we can see from the examples above, fixing the issue of word division resulted in errors in other parts of the text. The algorithm was made less strict about the distance between two readings, so that it would also match words separated by whitespace. This in turn led the algorithm to misalign words such as *gloria* and *matris*.

7.3.4 Output

The apparatus data can be exported for cladistic analysis with software such as PHYLIP¹⁹ or CIPRES²⁰, if the apparatus follows certain rules (CTE help manual). With some reservations, the CTE output may also be imported in Stemmaweb. It is very difficult to reconstruct the complete text of each witness from data prepared for the purpose of a print critical apparatus (Hagel 2007, 83), which is a requirement for Stemmaweb for instance. This demonstrates the potential significance of the difference between a print apparatus and data prepared for stemmatic analysis. It is an issue to prepare an apparatus for both printed and electronic editions. Ultimately, the final output of the Classical Text Editor is a print critical edition in PDF format, or a digital edition in TEI XML. Although an HTML export is available, it is now outdated (Hagel 2007, 80). There are templates provided with CSS stylesheet to visualise digital editions in a browser: for instance as synchronised texts in different frames²¹, as a text and apparatus²², or as transcription with words linked to a facsimile²³. Hagel (2007, 79) argued that ‘every scholar able to make a print edition with the CTE is at the same time able to make a digital edition’, but recognised that ‘if this editor does not set out for digital publication from the start, this affects of course the preparation of the data. Above all, a machine-readable critical apparatus will almost certainly not be maintained’ (Hagel 2007, 82).

Although CTE is dedicated to print editions, CTE’s TEI output can also provide an option for publishing printed versions of digital scholarly editions. Schulz (2017) for instance regrets that there are still limits to producing a decent printed output from an XML encoded edition, and describes the editing workflow of the *Capitularia* project where medieval capitularies are collated with CollateX and then prepared for a printed output with CTE. Schulz highlights the issues of this workflow, as well as the need for a better integration between print and digital for the future (Schulz 2017, 344).

7.3.5 Discussion

The primary purpose of Classical Text Editor is to ‘do the automatable work which consumes so much time and energy, and let the scholar concentrate on scientific issues’ (CTE website)²⁴. The program is designed to assist scholars doing tradi-

¹⁹<http://evolution.genetics.washington.edu/phylip.html> (Accessed August 31, 2017).

²⁰<http://www.phylo.org/> (Accessed August 31, 2017).

²¹<http://cte.oeaw.ac.at/tei/CBCat/> (Accessed September 1, 2017).

²²<http://cte.oeaw.ac.at/tei/en15/> (Accessed September 1, 2017).

²³http://cte.oeaw.ac.at/tei/il9/il_9.html (Accessed September 1, 2017).

²⁴<http://cte.oeaw.ac.at/?id0%3Dmain> (Accessed September 1, 2017).

tional print editions, and the addition of a collation feature is one aspect of this automatisisation (see Section 2.5 regarding the debate between using the computer as an assistant or as a research tool). As collation tool, the main interest of the CTE depends on the collation's purpose. If the purpose is to typeset and layout a complex critical edition (with several apparatuses, cross-references, indexes) especially a printed edition, then it may be worth investing the time to learn how to work efficiently with the CTE interface and the collation output. In other regards, CTE may not be the best collation tool. If visualising or printing a critical apparatus is not the primary purpose of collation, such as in the Beckett Archive project or the Digital Mishnah, then the CTE is not the best solution. In addition, the apparatus visualisation is not necessarily the best one to analyse the witnesses' relationships (Robinson 1989b): CTE does not, for instance, make it possible to visualise readings shared by a particular group of witnesses.

The fact that CTE is under an expensive commercial license can be an issue. Although Stefan Hagel has kindly offered his help to solve many problems, including updating the collation algorithm, this is unlikely to continue in the long term. After the license expires, it will be difficult to read the CTE files. Because of the license, the collation algorithm is also hidden, and users cannot tweak it for their needs. There is less control over the collation process, with only a normalisation feature that may require a lot of manual input. Users have no access to the underlying data structures, but only to plain text files. Regarding the input data, the import of XML documents is far from perfect and would benefit at least from documentation about how it is processed. While the collation results are easily modified by hand, it can break the original data structure and prevent from exporting the apparatus as genealogical data for cladistic analysis, or to a TEI digital edition²⁵.

7.4 Conclusion

In this chapter we have tested three collation tools with very different purposes. The first one, CollateX, is focused on optimising the alignment algorithm. It has flexible input and output formats, but it is the user's responsibility to find or create a visualisation interface. CollateX also requires users to have some coding capacities, at least to be comfortable with the command line tool. The second tool, Juxta, is more oriented towards visualisation and provides user-friendly intuitive interfaces. The possibility of data reuse is more limited in Juxta than in CollateX. Juxta still lets users have some control over the collation process by ignoring accidentals (case,

²⁵For that reason, the CTE manual recommends to copy and paste the collation output to a new document, before adding manual corrections.

whitespace, punctuation) or by choosing the XML elements to include or exclude from collation. Finally, the Classical Text Editor is a tool designed for print critical editions prepared from a base text, which means that the CTE may not be best suited to the automated collation workflow (see Section 2.4).

The least precise collation results for Calpurnius Flaccus were the ones obtained with the Classical Text Editor, although it must be noted that I collated only non-regularised tokens, and the updates provided by Stefan Hagel did improve the results considerably (see Section 7.3.3 above for the issues of word division and transpositions). The collation results of Juxta were satisfying for the purpose of visualisation in Heat Map format, and the collation from regularised texts was much better than from original texts when looking at the XML output. However I have found that CollateX results were ultimately more interesting, both for correcting the results and for creating new visualisations, because of the JSON input and output. In particular, the combination of a pretokenised JSON input and a JSON output makes it possible to include useful information associated to a token, such as normalised forms, or any other information that is of interest, such as the precise location of a particular reading in a manuscript, links to digital images of the manuscript, editorial notes or information about the markup context. There is no limitation: any information that a scholar wishes to see in the collation result can be passed as part of JSON. This information is ignored during the collation process but will still be available for visualisation. In the next chapter, we will see how to make use of CollateX JSON output for a visualisation with Python.

8.1 Assessing Scholarly Needs

THIS chapter focuses in more depth on collation visualisation. In Section 1.3, the purpose of collation was discussed: whether the purpose of collation is to create a critical edition, to check another scholar's work or to assess a manuscript tradition, collation is not the final goal. On the contrary, collation provides data that will be used for further research, such as establishing a *stemma codicum* for instance. Therefore, collation results from automated collation tools need to be analysed and visualised in some way. There are different options when it comes to analysing collation results: either an automated analysis with phylogenetic tools, or a more traditional analysis such as the application of Lachmann's method. In both cases, a good visualisation of collation data is required, if only to check and correct the results before submitting them to another tool for phylogenetic analysis.

What makes a good collation visualisation? How can it help the editor to assess the witnesses, their relationship, and to prepare a critical text? What information should be included in the visualisation? How should it be displayed? To answer these questions, the first step is to assess the needs of editors and readers. It is particularly important to take the perspective of literary scholars into account as it may differ significantly from the perspective of the computer scientists who create collation programs. For instance, the scientists may tend to think of variants as either additions, omissions or substitutions, whereas literary scholars may need to categorise variants as substantial or accidental, or distinguish between errors and 'true' readings (Hilton 1992, 142).

The rest of the chapter will be divided into two sections that describe two aspects of collation visualisation: the first is a fixed visualisation in the form of a collation table; the second is an interface that offers users the possibility of interacting with the collation results. The purpose of these visualisations is to propose a solution that takes into account the editorial needs which may not be covered in existing

visualisations, such as viewing paratextual elements, or helping scholars to find variant readings relevant to the application of Lachmann's method.

8.1.1 *Visualising Paratextual Elements*

Collation is more than just a record of variant readings: it should incorporate as well a number of 'paratextual' elements (Macé et al. 2015, 331). These elements include, amongst others, the structure of the manuscript such as the start of folia or pages, eventually the start of a new column, even the start of each line if one needs that level of precision. The reason for recording at least the beginning folia and pages is twofold (West 1973, 67). First, it may help during the analysis of the relationship between the witnesses: if an omission in one manuscript corresponds exactly to an opening of another manuscript, it may be a good indication that the first manuscript is derived from the second (for instance if the scribe turned two pages together and missed an opening while copying the exemplar). The second reason is that such indication is also useful for checking purposes: if there is any doubt that a mistake may have occurred during the collation, it will then be easier to go back to the right place in the manuscript and check that the reading recorded in the collation corresponds to the reading in the manuscript.

A change of hand or ink in the manuscript, if noticed, should be recorded as well: it could be an indication that a different scribe has taken up the work of copying the manuscript, and the new scribe may use a different exemplar than the first scribe. The use of two different exemplars results in contamination, which is important information that will help to understand the manuscript tradition and to create a stemma. Any damage that impairs the editor's capacity to read the text from the manuscript should also be noted, such as holes, gaps or lacunae, missing folia, illegible or difficult readings, or unclear abbreviations. Corrections, either by the copyist or by another hand, are noted. Marginal notes may be recorded as well. All these paratextual elements may require extensive comments and careful description from the collator (Whittaker 1991, 125).

Visualising paratextual elements is therefore one important aspect of what a scholar may need from a collation visualisation. Another aspect pertains to the analysis of the collation results in order to detect the relationships between witnesses. Here we will consider how collation results can support this analysis through the application of Lachmann's method.

8.1.2 Applying Lachmann's Method

To detect relationships between witnesses, many scholars follow the (Neo-) Lachmannian method of text editing (Trovato 2014). In this discussion, 'Neo-Lachmannism' refers to the improvements to Lachmann's method brought by Pasquali and other Italian scholars, who took Bédier's criticism into account and incorporated the study of the textual tradition and material documents (the manuscripts themselves) to the creation of stemmata¹.

Lachmann's method focuses on identifying common errors shared by a group of witnesses in order to postulate relationships between those witnesses: a group of witnesses are likely to be related if they (1) agree together on readings that (2) they do not share with the other witnesses, and especially (3) they agree in errors, i.e., they share readings that have no manuscript authority. A reading with manuscript authority is 'a reading that may have reached us through a continuous sequence of accurate copies of what the author wrote back in antiquity and may therefore be authentic and (by definition) right' (Damon 2016, 202-203). The opposite of an error, i.e. a reading with manuscript authority, is called here a 'true' reading². Errors and true readings are not definitive, but rather reflect the views of an editor on the text.

In sum, witnesses are more likely to be related when they share readings that do not represent the original text, and that are absent from other witnesses. Such shared errors are conjunctive, that is errors which help to group related witnesses together. On the other hand, separative errors are readings indicating that two witnesses are not directly related and should appear on different branches of the stemma:

We can prove that a witness (B) is independent of another witness

¹Bédier (1864-1938), a French Romance philologist, rejected the common-error method of reconstructing a stemma, and decided it was best to edit one manuscript instead of publishing a text that incorporates readings from various origins and therefore never actually existed (Bédier 1928). Pasquali (1885-1952) showed that the so-called 'Lachmann' method was in fact the result of contributions from many scholars (Pasquali 1952).

²It seems that 'true reading' is often used technically as the opposite of 'error'. The expression 'true reading' appears in Maas (1958, §38) (from the German *das Wahre*) and in articles by editors in Classics such as Reeve (2000) and Courtney (1967). D'Avray (2012, 70) states that 'the whole purpose of a stemma is to distinguish between true readings and errors'. West (1973, 31) explains that the evaluation of variants 'involves deciding not only which variants are true and which false, but also which are scribes' emendations'. West refers again to the expression 'true readings' (pages 32, 39, 41, 43 etc.), but also to 'good', 'correct' or 'right' readings (for instance West 1973, 33 and 45). The alternative 'correct reading' can be found as well in Trovato (2014, 252) and 'genuine reading' in Macé et al. (2015, 416).

(A) by finding in A as against B an error so constituted that our knowledge of the state of conjectural criticism in the period between A and B enables us to feel confident that it cannot have been removed by conjecture during that period. Errors of this kind may be called ‘separative errors’ (*errores separativi*) (Maas 1958, 42).

Finally, other readings of interest in Lachmann’s method are ‘unique errors’, which are errors found only in one witness. Those unique errors may help determine the relationship of two related manuscripts, and decide which one is the descendant of the other. Between two related witnesses A and B that share common errors, if B has in addition unique errors that could not have been easily corrected, it can be concluded that the witness B is a direct descendant of A (West 1973, 33). It is important that those errors could not have been easily corrected, otherwise, A could also be the direct descendant of B, since the scribe of A would have easily corrected the mistakes in B.

In order to be significant, the errors considered in Lachmann’s method must also be monogenetic, i.e., errors which could not have been produced independently in two unrelated witnesses (see Trovato 2014, 55-56). Omissions are often useful monogenetic errors for reconstructing the manuscript tradition, because it is difficult for a scribe to correct an omission and recover the missing text by conjecture only. On the other hand, the omission occurring because of a *saut du même au même* is not a monogenetic but a polygenetic error, which can happen independently in several unrelated witnesses. In a *saut*, the copyist omits a section of text between two instances of the same word which appear in close proximity on the manuscript’s page. The *saut du même au même* is a common error among scribes (Reynolds and Wilson 1991, 204), and therefore it is not an error that can be used to infer relationships between witnesses.

In summary, the application of Lachmann’s method requires scholars to compare witnesses in several ways: for instance comparing a group of witnesses to find shared conjunctive errors, or comparing one witness against another to find separative errors, or finding readings unique to a single witness. In addition, scholars could also benefit from help in distinguishing between actual errors and authentic readings. Applying Lachmann’s method, the analysis of shared errors and unique errors will result in conclusions regarding the textual tradition and in some cases in a stemma. Those conclusions will in turn influence the selection of variants to include in a critical text, and are important in the process of editing. Therefore, scholars need to share their results with other scholars who should be able to

confirm the conclusions by reproducing the steps of reasoning that led to them.

8.1.3 *Sharing and Reproducing Research*

The traditional way for scholars to share the conclusions of their textual editing is in a critical edition, with introductory notes and apparatus; collations themselves are not always published. However, there are shortcomings to the critical apparatus: when it is a selection from the complete collation, it is an incomplete record of all the evidence which was available to the editor (see Section 1.3).

In addition, the printed critical apparatus is not a convenient data format to facilitate the study of a particular relationship among witnesses. In Calpurnius Flaccus, for example, the question was raised whether the editor Håkanson was right when he claimed that Pithoeus based his edition on manuscript N, or if manuscript N could have been copied from Pithoeus' edition³. The question is worth examining, since Pithoeus' comments on the Italian exemplar that he used does not allow us to identify a precise manuscript (see Section 5.1.2.6). In order to evaluate the relationship between N and Pithoeus, it would be necessary to search the apparatus for all instances where Pithoeus and N are in agreement in a reading which is not found in the other witnesses. It would also be necessary to search the apparatus for unique readings of both Pithoeus and N, in order to decide which one is the descendant of the other.

As a consequence, editors and readers would benefit from collation results in a format that can easily be shared, and enables scholars to follow and analyse one another's conclusions.

8.1.4 *Summary*

The editors who work with collated witnesses of a text have various needs. First, the combination of variants, annotations and paratextual material produce a large amount of complex collation data, which are difficult to read and to interpret. A good visualisation should therefore offer a way to check the collation data against the actual witnesses, whether they are manuscripts or printed editions, and to visualise not only variant readings but also all the paratextual elements associated with the readings. Second, being able to find common errors or unique errors in the collation results would therefore be especially useful for scholars preparing

³The question was raised by the librarian responsible for the Bongars collection in the Bürgerbibliothek of Bern, as I was enquiring if they knew anything about the original provenance of manuscript N.

a critical edition. And finally, scholars need to share their conclusions and the collation data that was used to reach those conclusions.

To this end, the editors should be able to interact with the collation to analyse readings and variants according to their editing method. Here we have considered the Neo-Lachmannian method of textual editing, which is still commonly followed in the editing of Classical texts, especially in order to evaluate the stemmatic weight of a witness. Collations could thus be filtered, so as to find patterns of agreements or disagreements between those witnesses, which can indicate how they are related to each other. Finally, scholars need to share their conclusions and make them reproducible or eventually disputable by others (see Section 1.3). Traditionally this is the purpose of the critical apparatus, however the apparatus may not always be a complete selection of variants. In some cases, it has been argued that the apparatus does not provide the full context for each witness, or is biased to confirm the editor's conclusions (see Section 1.3). But more importantly, a printed critical apparatus makes it very difficult to find quickly where a group of witnesses agree.

Visualisation and manipulation of collation results are essential in order to use collation for further research, such as studying the manuscript tradition and creating a stemma codicum. However, the existing visualisations are mostly linear: the reader must follow the text word by word and it is difficult to select only variants of interest, such as common readings or unique readings. Little is offered in terms of searching a collation for groups of witnesses which agree, or to find unique errors. A new perspective may be required in order to fulfil the needs of editors: the ability to filter the collation, and select agreements of witnesses or unique readings. In addition, the visualisations will usually show only plain text from the witnesses, and omit the paratextual elements that could be useful to an editor.

Therefore I would like to describe in this chapter two examples of visualisation in order to address those issues. In the next section, I will focus on the issue of visualising paratextual elements in a collation table. I will describe a table format developed to include more than only textual readings, using CollateX JSON input and output (see Section 7.1). The last section of the chapter will be devoted to the description of an interface that allows for scholars to interact with the collation results in order to apply Lachmann's method. In both the interface and the collation tables that will be described below, I have used the property **t** and **n** of CollateX's JSON tokens, as described in Section 4.4, in order to visualise as variants only the lexical differences, and to ignore the orthographic and word division differences. In practice two tokens are not considered as variant if their original forms **t** are

8.2. Table format: fixed visualisation

W1	dives iure	erit	pater	praemio	petit
W2	dives iure	egit	pater	praemio	petit
W3	dives iure	agit	pauper	praemio	petit
W4	dives iure	egit	pauper	praemio	petit
W5	dives iure	erit	pater	premio	petit

Figure 8.1: An example of a CollateX Collation table. Created from the CollateX demo <https://collatex.net/demo/> (June 18, 2017).

different, but their normalised forms **n** are similar. The issue of normalisation is discussed more in detail below in Section 8.6.

8.2 Table format: fixed visualisation

Previously, we have discussed the existing visualisations and outputs offered by the various collation tools (see Section 2.6.4 and Chapter 7). While the earlier tools and the Classical Text Editor would focus on getting an output that imitates a critical apparatus, most recent tools prefer to show variant texts in parallel (Juxta, TUSTEP), in the form of a graph (CollateX, TRAViz) or as a collation table (CollateX, iAligner). Among the possible outputs from collation tools, I have chosen to focus on the collation table format.

The collation table is a visualisation which is user-friendly even for scholars who do not work with CollateX or any computer-supported collation program, and it is quite similar to the format of a manual collation. Macé et al. (2015, 333) show two examples of manual collations in table format, where each column represents a different witness. Several examples of collation tables from CollateX were also presented in Chapter 2. The table typically represents each witness on a separate line or column, with their text aligned when it matches, and blank spaces inserted where a part of the text is missing in a witness. In its most simple form, the table will show only plain text from the witnesses. Enhancements have been proposed to improve the basic table output of CollateX, for instance with colours to indicate the places where a variation occurs: in CollateX online demo, the readings which are part of a variant location are shown in grey, while the readings which are shared by all witnesses are highlighted in green (see figure 8.1).

Taking in account the scholarly needs identified above, I will first review existing examples of collation tables from various projects, and I will then show a new

8.2. Table format: fixed visualisation

Version 1: MS-HRC-SB-5-9-1	In	the		beginning	.	
Version 2: MS-HRC-SB-5-10	At	In	the	outset	beginning	.
Version 3: MS-Spectrum	At		the	outset		.
Version 4: MS-WU-MSS008-3-71	In	the		beginning	.	
Version 5: MS-1958	In	the		beginning	.	

Figure 8.3: A collation table from the Beckett Archive Project (Van Hulle, Neyt, and Nixon 2014).

Single view	Compare	Tree view	Table view	
h080g/add0	reek	,		fringed with oaks and the wild willow, ran N oiseless ly on, betw een the pione ers
h080d	reek	,		fringed with oaks and the wild willow, ran N oiseless ly on, betw een the pione ers
h080l	reek		n	oiseless ran betw ixt the pione ers
h080m	reek		n	oiseless ran betw ixt the pione ers
h080i/base	reek	,	n	oiseless ran betw ixt the Pione ers
h080i/rdg2 h0.	reek	,	n	oiseless ran betw ixt the Pione ers
h080f	reek			fringed with oaks and the wild willow ran N oiseless ly on, betw een the pione ers
h080h	reek	,	n	oiseless ran betw ixt the pione ers
h080a	reek	,		fringed with oaks and the wild willow, ran N oiseless ly near, betw een our travell ers
h080g/base	reek	,		fringed with oaks and the wild willow, ran N oiseless ly on, betw een the pione ers
h080k	reek	,	n	oiseless ran betw ixt the pione ers
h080i/rdg1	reek	,	That	n oiseless ran betw ixt the Pione ers A

Figure 8.4: An example of a Compare collation table (Schmidt and Eggert 2015).

and Eggert 2015). Figure 8.4 shows an example from the Table view of Harpur’s ‘The Creek of the Four Graves’.

Unlike the other collation tables, this visualisation depends on a base text, and variant readings are highlighted at the letter level, so that users can see at once where the difference is located. The differences with respect to a base text (‘h080i/base’ on the fifth line) are highlighted in blue. The downside of this visualisation is that words are often split in two or more columns because of the letter-level alignment and highlighting. As a result, the word ‘Noiselessly’ here is separated across three columns in several witnesses, which can make it difficult to read (see figure 8.4). The Compare tables also include page numbers of manuscripts and editions, which appear as variants in the collation table, because the project requires the transcription of page numbers as part of a witness’ text⁵.

⁵See the Ecdosis page about the transcription policy for pages, columns, and lines: <http://ecdosis.net/main/node/23> (Accessed September 4, 2017).

8.2. Table format: fixed visualisation

8.2.1.4 iAligner

iAligner also offers a collation table visualisation, containing additional information embedded in the table with the help of colours. In the example of figure 8.5, readings may be highlighted in red, dark green, and light green. The readings highlighted in red represent the unique readings of a variant location. Light green readings represent readings similar across several witnesses at a variant location (Yousef, Palladino, and Crane 2017). This visualisation is an interesting progress towards visualising groupings of witnesses and unique errors according to Lachmann's method. However, the table may need to add more shades of green in case there is a location with variants that are present in more than two groups of manuscripts.

Figure 8.5 shows a collation table with four rows of Greek text. Each row contains several words and punctuation marks. The cells are color-coded: red for unique readings, dark green for readings similar across several witnesses, and light green for readings consistent across witnesses. The text in the table is as follows:

πρῶτος	δὲ	Δημόκριτος	πολύπειρος	άνηρ	συνεῖδεν	ὅτι	προμήκης	ἔστιν	ἡ	γῆ	ἡμίολιον	τὸ	μήκος	τοῦ	πλάτους	ἔχουσα	.
πρῶτος	δὲ	Δημόκριτος	πολύπειρος	άνηρ	συνεῖδεν	ὅτι	προμήκης	ἔστιν	ἡ	γῆ	ἡμίολιον	τὸ	μήκος	τοῦ	πλάτους	ἔχουσα	.
πρῶτος	δὲ	Δημόκριτος	πολύπειρος	άνηρ	συνεῖδεν	ὅτι	προμήκης	ἔστιν	ἡ	γῆ	ἡμίολιον	τὸ	μήκος	τοῦ	πλάτους	ἔχουσα	.
πρῶτος	δὲ	Δημόκριτος	πολύπειρος	άνηρ	συνεῖδεν	ὅτι	προμήκης	ἔστιν	ἡ	γῆ	ἡμίολιον	τὸ	μήκος	τοῦ	πλάτους	ἔχουσα	.

Figure 8.5: A collation table from iAligner (Yousef, Palladino, and Crane 2017, §5).

8.2.1.5 Stemmaweb

In Stemmaweb, the tool called Stexaminer makes it possible to examine variants against a particular stemma. The Stexaminer's purpose is to help scholars to determine if the textual evidence supports the stemma. To this aim, the stemma is accompanied by a collation table which displays the entire text, with colours to indicate how well the variants fit the stemma. If a row of the collation table is in green, this means that the variant readings are genealogically consistent, according to the stemma. If the row is not green, it indicates a conflict. The colour red signals a reading which must have originated independently from two or more witnesses; the colour yellow indicates a possible reversion, that is a reading which may have been corrected by scribal conjecture (Stexaminer Documentation)⁶. The collation table of the Stexaminer therefore lets users test the validity of a stemma hypothesis.

8.2.1.6 Summary

Those examples of collation tables suggest that a basic table, such as the one of the CollateX demo, is not the best solution for scholars to visualise their collated

⁶<https://stemmaweb.net/stemmaweb/stexaminer/6C65C98E-B150-11E1-95DC-A5115FAAC71C/help> (Accessed November 23, 2017).

8.2. Table format: fixed visualisation

The screenshot displays the Stexaminer interface for the 'Chronicle of Matthew'. It features a table of variant readings with columns for variant number, manuscript identifier, and the reading itself. Some readings are highlighted in yellow, green, or red. Below the table is a stemma diagram showing the relationships between the manuscripts. To the right, there are 'Aggregate text statistics' and an 'Analysis options' button. The statistics include: Total number of variant locations analyzed: 1862; Number of fully genealogical locations: 1058; Number of readings that conflict with the stemma: 44; and Genealogical reading transitions by relationship type.

Figure 8.6: Example of a collation table in Stemmaweb's Stexaminer. Retrieved from <https://stemmaweb.net/stemmaweb/stexaminer/6AB3650E-12AA-11E2-8D75-ABBB4ACE906A/0> (November 23, 2017).

text. It is not enough to highlight the readings in a variant location, and for this reason each of the projects examined has worked on improving the design of the collation table according to their requirements. Some projects have incorporated paratextual elements from the transcriptions, such as additions and deletions in the BDMP project, or page numbers in Charles Harpur edition. Other projects have focused on grouping manuscripts with the help of colours, for instance to gather manuscripts according to a shared orthography in the Digital Mishnah project. The tables from iAligner are the closest solution that would help scholars to find the variants relevant to applying Lachmann's method, and the tables from Stemmaweb are useful to check a stemma hypothesis that has already been created (either manually or with the help of genetic software).

However, other elements are still missing from those helpful visualisations. Most of them include little, if any, of the paratextual elements described above such as the changes of folia or columns and lines, and other types of editorial annotations. There are no indications, for instance, of unclear readings in the witnesses. The other issue of these tables is their linearity: users need to scroll through the entire text in order to see all the variants, when they may need to see only a specific selection of readings. In the next section, I explore how the table visualisation may be improved.

8.2. Table format: fixed visualisation

B1	B2	C1	C2	LH	M1	M2	N1	N2	P1594	ID
excerpta	excerpta	excerpta	excerpta		contra matrem	contra matrem	contra matrem	contra matrem	contra matrem	1
					contra matronam	contra matronam	contra matronam	contra matronam	contra matronam	16
					pro milite	pro milite	pro milite	pro milite	pro milite	24
⊙	⊙	⊙	⊙		o	o	o	o	o	65
										67
Note: Unknown abbreviation. Normalized form supplied by Lehnert	148r:8	82r:18	82r:18		2r:7	2r:7	244v:28	244v:28		2
148r:8	verginus	verginus	verginus	Verginius	virginus	virginus	virginus	virginus	Virginus	76
					pro parricida	pro parricida	pro parricida	pro parricida	pro parricida	84
pater	pater	pater	pater	patiar includi	paterer	paterer	paterer	paterer	patiar includi	124
est	est	est	est	es	es	es	es	es	es	127

Figure 8.7: A collation table for Calpurnius Flaccus, with additional information about readings.

8.2.2 Collation Tables for Calpurnius Flaccus

This section describes an example of a collation table for the *Declamations* of Calpurnius Flaccus, which includes more elements such as folio and line numbers for every reading, and editorial comments as well. The table also uses red and green colours to organise readings, but not in the same way as in iAligner tables. The colours are used here to separate errors from true readings, according to one version of the text considered reliable (such as a critical edition). On the other hand, the unique errors or groups of variants, which were in red and green in iAligner, are discovered through an interactive interface described later in Section 8.4. Therefore, the tables will not contain a sequence of text, but rather a selection of variant readings, obtained with the interface.

8.2.2.1 Table visualisation

Figure 8.7 shows a collation table for Calpurnius Flaccus. I have created this table by comparing the agreements of normalised readings among the different witnesses, with the help of the Jupyter notebook described later (p. 263 below). The table of figure 8.7 is thus only a selection of the complete collation table.

There are four manuscripts in this collation: B, C, M and N. Each manuscript is divided into two witnesses according to the different hands which wrote the text. For example, B1 is the first hand of manuscript B and B2 is the second hand who made corrections to the text of the first hand. At the time, manuscript A was omitted from this visualisation, because of its damaged and incomplete state (see p. 295 below).

There are also two editions in the collation. The *editio princeps* of Pierre Pithoeus, in the reprinted version of 1594. The second edition is the critical edition of Calpurnius Flaccus published by Lennart Håkanson in 1978. The last column, ID, represents

8.2. Table format: fixed visualisation

a way to identify rows in the table. The rows of the full table start from 0 (the first words in the text) to 1338 (the last words of the *explicit*). The table shows the list of variant readings shared by four witnesses (B1, B2, C1 and C2) against five others (M1, M2, N1, N2, P1594). Since the table represents only a selection of variant readings, the ID number gives an idea of where each reading appears in the text, and how it is related to the other variants in the table. There are a few items highlighted in the table of figure 8.7:

1. The (i) symbol next to a reading: on click, it can reveal or hide editorial comments that were made during the transcription, especially regarding problematic passages. Here the comment is related to an unknown abbreviation that was not resolved with certainty by the editors of Calpurnius.
2. The (:) symbol in the ID column: on click, it will reveal or hide locations of the readings for each witness in the row.
3. The location is in the form of ‘folio number:line number’. The unknown abbreviation mentioned previously on p. 194 appears in folio 148r, line 8, of manuscript B. Since there is a digital facsimile available for manuscript B, the location will also link to the image of the page.
4. Green and red colours: the coloured lines next to a reading show agreement (green) or disagreement (red) with the same reading of a base text, chosen among the witnesses. Here the base text is the text printed in the edition of Håkanson. As a result, the readings Håkanson rejected because he considered them to be errors are shown in red, while the readings he accepted as true are shown in green. The pattern of colours would of course be different if another text, such as Pithoeus’s edition, had been selected as the base text. The purpose of the colours is to detect relationships between witnesses, according to the (Neo)-Lachmannian method of text editing.

The collation table is presented in a column format, instead of showing each witness on a line, as in the other tables described above. The vertical visualisation has two advantages: first, it makes it easier to read through a list of disconnected variants, whereas the horizontal format seems to encourage a reader to read line by line, each witness separately. In addition, the vertical format has turned out to be interesting because it can represent texts both in Latin or in Hebrew. There is no need to worry about the direction of writing (right-to-left or left-to-right) in a vertical table. This can facilitate the adaptation of this table format to textual traditions in other languages (see Section 8.6.2 below).

8.2.2.2 Code description

The HTML file containing this example of a collation table is included in a Github repository with other digital materials (see also Appendix B.4). This HTML collation table format was created with the help of my colleague Ginestra Ferraro, a designer working for King's Digital Lab (KDL). To be properly displayed, the HTML file should be placed in the same directory as three other folders. One folder contains the fonts: Font Awesome was used for the icons in the table, such as the (i) sign which indicate a note⁷. Another folder contains the Cascading Style Sheets (CSS) to apply to the HTML documents, and the last folder contains javascript instructions with jQuery⁸.

The jQuery instructions are necessary to switch notes and folio numbers from hidden to visible. Each cell of the HTML table (<td>) contains at least one paragraph <p> which is either empty or has a reading to display. In addition, each cell which is not empty contains a second paragraph with the attribute `class='expandable-row hidden more-info'`, where the information about page and line numbers are stored. Some cells also have notes, which are stored in another paragraph with the attribute `class='expandable hidden more-info'`. Finally, the (i) and (:) signs are included in a link (so as to make them clickable) with respectively the attribute `class='expander right'` (for notes) and `class='expander-row right'` (for folio numbers, which must be expanded for the whole row). In the javascript file `config.js`, two functions will remove hidden attributes from the class of the paragraphs or add it back if it was absent from the class when the link was clicked.

For instance, the second function in figure 8.8 is binding the element with a class `expander-row` (the `:` symbols) to an action performed on click. The symbols are within a paragraph, within a cell, within a row, so that on click of the symbol (`$(this)`) the function must go three steps up in the hierarchy to get to the whole row (this is the role of the `parent()` functions). Then it must find all elements in the row which have the attribute `expandable-row` (`find('.expandable-row')`) and apply two actions: the first is the `slideToggle()` function which displays or hides the elements with a sliding motion. The second is to remove the hidden class. The CSS is instructing objects with a hidden class not to be displayed on screen. This prevents a large table with many elements from being loaded too slowly in the browser. However,

⁷<http://fontawesome.io/> (Accessed June 22, 2017). Font Awesome is an open source set of icons which can be customised with CSS.

⁸jQuery is a fast, small, and feature-rich JavaScript library that makes it easier to process HTML documents and to make those documents work across different browsers. <https://jquery.com/> (Accessed June 22, 2017).

```

// Expandable notes
$('.expander').bind("click", function() {
  $(this).parent().next('.expandable').slideToggle(400).removeClass("hidden");
  //$(this).parent().parent().find('.expandable').slideToggle(400).removeClass("hidden");
  // $("i", this).toggleClass("fa-caret-down fa-caret-right");
  return false;
});

// Expandable row info (folio:line)
$('.expander-row').bind("click", function() {
  $(this).parent().parent().parent().find('.expandable-row').slideToggle(400).removeClass("hidden");
  // $("i", this).toggleClass("fa-caret-down fa-caret-right");
  return false;
});

```

Figure 8.8: Javascript for expanding or hiding paragraphs.

when the hidden class is removed, the object is then properly displayed on screen. The behavior that adds the class back on click is determined by the `toggleClass()` function, which in this case is located in the file `jquery-2.2.1.min.js`.

8.2.3 Conclusion

There is still scope for improving existing collation table formats, as we have seen with the example of Calpurnius Flaccus. It is possible to integrate more information into the collation easily not only with colours, but also with information such as the location of a word in the manuscript, or editorial comments, that can be revealed on a click. Those paratextual elements are important elements in text collation and there is no reason to discard them in a visualisation that has been obtained through automated collation tools such as CollateX. As shown in the collation table, the use of a few symbols allows to make those elements easily available without overcrowding the results or slowing the charging of the page in a browser. Through the rest of the chapter, I will focus on the interface through which collation results may be updated and manipulated: I will first compare existing interfaces (Section 8.3), and then describe the tool that I have created in order to obtain collation tables such as the one presented above (Section 8.4).

8.3 Interactive interfaces

Not all collation tools offer an interface for users to interact with collation results, and the levels of interaction possible may vary. For instance, CollateX does not provide any form of interface beside a limited online demo. In Juxta, the interactions with the results are limited to adding comments, or switching the base text in the heat map visualisation. In the Ecdosis framework, it is not possible to modify the collation results, but only to search for readings in the text. Although adding comments or searching for a reading can be considered as interactions, the collation data behind these visualisations is still static: the user has no direct control over the collation results once the algorithm has been run on the transcriptions.

The effect is to give users an impression of finality and correctness (Andrews and van Zundert 2013), even though it is likely that users will need to correct and refine the alignment (see Section 2.4.4.3).

As Andrews and van Zundert (2013) argue, a dynamic interface is necessary to analyse collation results: it encourages scholars not to trust blindly the results of an algorithm, but to apply their own judgement regarding the correctness of the results; besides, it also encourages scholars to engage with collation results in order to conduct further research. The interactions available should include at least four options:

1. Annotating variants about their relationship;
2. Combining several readings into one reading;
3. Splitting one reading into several readings;
4. Correct or alter the alignment.

8.3.1 Examples

Andrews and van Zundert propose, as a solution to counter static visualisation, a JavaScript library to implement dynamic interactions within a variant graph visualisation. This library was adopted by the *Institut für Neutestamentliche Textforschung* (INTF) to work the *Editio Critica Maior* of the Greek New Testament and by *Stemmaweb* (Andrews and van Zundert 2013). Andrews and van Zundert's recommendations are also followed by *TRAViz*, with the possibility to split or merge the nodes of the variant graph (Jänicke et al. 2015, i87), and by *LERa* (Bremer et al. 2015), but without the option to comment on variant readings at the moment. However, although every interface offers 'splitting' and 'merging' readings, it should be noted that this means slightly different actions depending on each interface.

8.3.1.1 Stemmaweb

Stemmaweb's interface offers logged users the ability to 'view collation and edit relationships' for their own textual traditions. Regarding the annotation interactions, there are several options. It is possible to set a variant as lemma, and annotate variant readings by creating a relationship of type orthographic, grammatical, lexical, and so on (figure 8.9). These relationships are then visualised as coloured edges in the graph. In addition, it is possible to give the scope of the relationship as local

8.3. Interactive interfaces

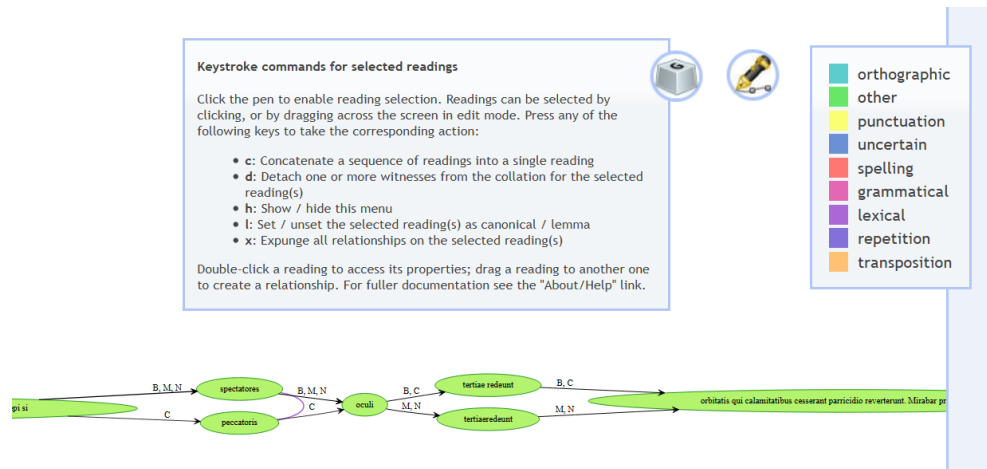


Figure 8.9: Stemmaweb interface for modifying and annotating collation results. Retrieved from my personal account (November 20, 2017).

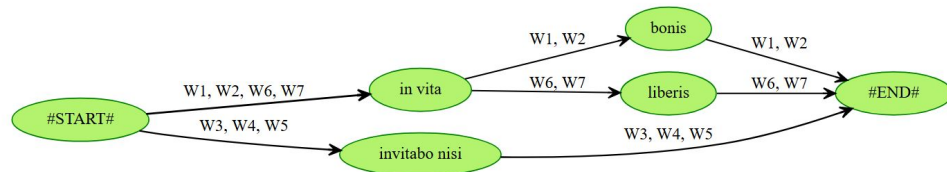


Figure 8.10: Stemmaweb graph example. Retrieved from my personal account (November 21, 2017).

or global, and to mark a relationship as stemmatically significant, and to include a comment in text format.

On the other hand, there are two options to modify the collation results: merging or splitting nodes in the graph. Two nodes can be merged if they share the same reading, for instance if a reading was not correctly aligned, or if two consecutive nodes are traversed by the same witnesses. A node of the graph can be split in two parallel readings, but not in two consecutive readings. For instance in figure 8.10, it is possible to split the reading *in vita* in two nodes in order to detach witnesses: one nodes for witnesses W1 and W2, and one node with witnesses W6 and W7. Then the readings *in vita bonis* (W1, W2) can be merged in one node. However, it is not possible to split the reading again, or to separate *invitabo* and *nisi* in two different nodes.

8.3.1.2 Collation Editor

The Greek New Testament edition project at the INTF became part of a larger collaboration with two other teams of researchers in Germany and in the UK: the International Greek New Testament Project (IGNTP) and the *Institut für Septuaginta- und biblische Textforschung* in Wuppertal (ISBTF). This international collaborative project required the development of a shared online environment, called the Workspace for Collaborative Editing, to edit the *Editio Critica Maior* of the Greek New Testament (Smith 2015; Houghton and Smith 2016)⁹. The Workspace for Collaborative Editing provides three interfaces, one of which is the Collation Editor.

The Collation Editor is designed for the purpose of collating the complex and abundant material of the Greek New Testament. There are several stages to the collation process: the first is to select a passage of text and a list of witnesses to be collated, and set the parameters of the collation algorithm; the second stage is to regularise the collation, for instance by removing insignificant orthographic variations. At this stage of regularisation, any reading can be annotated with a comment in plain text.

The next step is to 'Set Variants', and it is the stage when editors actually modify the collation results, by merging or splitting variation units. Unlike the Stemmaweb interface, the Collation Editor allows for a variation unit to be split into either readings or words. The latest makes it possible to separate the reading *invitabo nisi* into two separate words, *invitabo* and *nisi*. A variation unit split into readings can then be further split to detach specific witnesses.

Although the Collation Editor is a very detailed interface, it does not facilitate the visualisation of shared errors or the application of Lachmann's method. First, this interface is designed for preparing the collation data which will be analysed later, once each verse has been correctly collated. Second, the material of the Greek New Testament is too rich and complex to apply traditional editing methods, and inconsistent or contradictory data prevent editors from working with shared errors (Houghton and Smith 2016, 121).

8.3.1.3 TRAViz

Traviz proposed a new graph visualisation which is meant to improve the readability of the collation results, compared to the graphs of Stemmaweb (see also p. 108 above in Section 2.6.4). In terms of interactions to modify the collation results,

⁹<http://www.itsee.birmingham.ac.uk/> (Accessed November 20, 2017).

8.3. Interactive interfaces

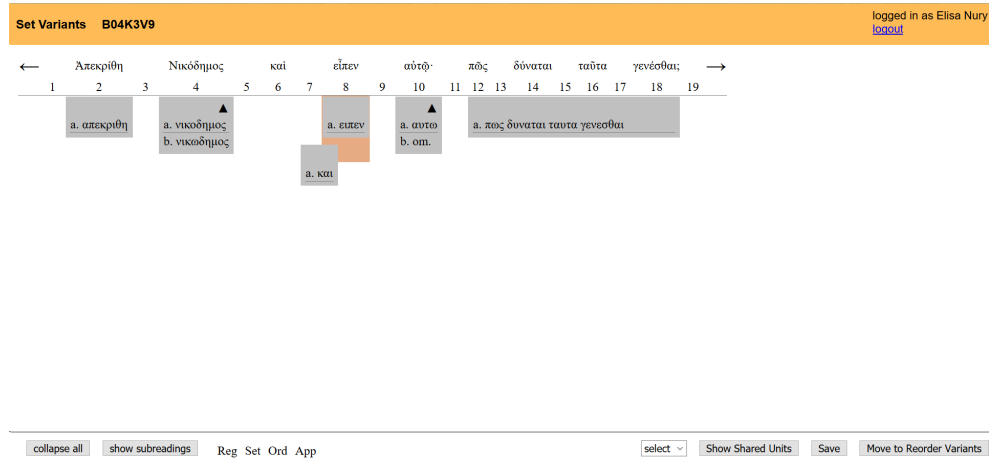


Figure 8.11: Modifying collation results with the Collation Editor. Retrieved from <http://www.itsee.birmingham.ac.uk/collation/> (November 20, 2017).

Calpurnius Flaccus - Sample

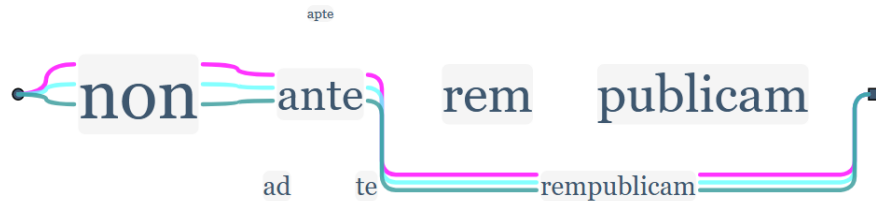


Figure 8.12: Example of a TRAViz graph interface.

TRAViz allows for splitting and merging nodes of the graph. One node can be split into parallel readings for each witness, as it was the case in Stemmaweb. Each node represents one word only, which was tokenised at whitespace separation. Nodes can be merged only if they are parallel readings at the same point of variation, which means that it is not possible to merge two consecutive readings. For instance the words *rem publicam* are divided in two nodes and cannot be merged. The word *republicam* can be merged with either *rem* or *publicam*, but not with both (figure 8.12).

In order to improve the graph and to hide orthographic variation such as *republicam* versus *rem publicam*, the user has to play around with the relative edit distance parameter in the JavaScript code, which will add fuzzy matching to the collation

algorithm. In that sense, this is not an interaction with the results, but with the algorithm, before collation results are displayed. For the small sample from Calpurnius, a relative edit distance of 0.4 will merge *publicam* and *republicam*, hiding the word division difference. It will also hide the difference between *ante* and *apte*, and this can then be corrected by splitting the node in two. This situation illustrates the issue of fuzzy matching: it may be very difficult to set the parameter correctly in order to hide real orthographic variation (*rem publicam* versus *republicam*), but not lexical variants (*ante* versus *apte*) as it was also the case with CTE's algorithm (see Section 7.3.3).

Hovering over a node in the graph highlights the witnesses passing through this node, while hiding the other witnesses. In figure 8.12 above, the graph shows only the witnesses passing through the node that contains the reading *republicam*. The graph shows only the connections of those specific witnesses, which is meant to help 'investigate the graph distribution of a subset of potentially similar [witnesses] and the exploration of the similarities and differences among them' (Jänicke et al. 2015, i88). Since this interaction happens when the mouse hovers over a node, its scope is limited. As soon as the user needs to scroll to see more text, the highlights will disappear. The visualisation is also limited for scholars who are looking for shared errors: the graph shows readings common to some witnesses, but not against others, and it makes it more difficult to recognise significant variants that can help to establish the witnesses' relationships.

8.3.1.4 Summary

These dynamic interfaces respond to the need to correct the collation results, and to annotating variants. All three offer the possibility to 'split and merge' nodes on the graph, although the range of actions available are different in each interface. The most complete is the Collation Editor, in which the collation results can be altered to fit exactly the need of editors. On the other hand, Stemmaweb provides annotations to describe how variant readings are related to each other with a list of categories.

However, it is possible still to add new interactions to a collation visualisation, and to encourage scholars to engage even more with the collation results. For instance, the application of Lachmann's method is still difficult. In addition, a dynamic interface could also solve the issue of linearity raised in the previous section. Instead of displaying the complete collation results, the interface would let users select only the variant locations relevant to their research question. This is what the interface for Calpurnius Flaccus tries to implement: to help scholars engage with collation in

8.3. Interactive interfaces



B1	B2	C1	C2	LH	M1	M2	N1	N2	P1594	ID
quinque cum tyranno	quinque cum tyranno	quinque cum tyranno	quinque cum tyranno	quinque cum tyranno	quinque cum tyranno	quinque cum tyranno	quinque cum tyranno	quinque cum tyranno	Quinque cum tyranno	6
				proximi						7
							familiae			8
proxime	proxime	proxime	proxime		proximae	proximae	proximae	proximae	proximae	9
familie	familie	familie	familie	familiae	familiae	familiae		familiae	familiae	10
puniantur	puniantur	puniantur	puniantur	puniantur	puniantur	puniantur	puniantur	puniantur	puniantur	11

Figure 8.13: Example of interface with widgets.

order to apply Lachmann’s method, and to make hypotheses about the witnesses relationship by finding shared errors or unique errors.

8.3.2 Interface for Calpurnius Flaccus

The interface that I have created is named PyCoviz, for Python Collation Visualisation. PyCoviz is meant to help scholars answer questions such as: What are the readings found in this group of witnesses, but not in that other group of witnesses? What are the readings common to any two given witnesses? What readings are unique to a particular witness? These questions are essential when trying to establish the relationships between witnesses (Robinson 1989b, 175). However, PyCoviz also integrates most of the necessary interactions discussed by Andrews and van Zundert (2013).

Figure 8.13 shows the main view of PyCoviz, which gathers all the interactions available in tabs: it is possible to view a collation extract, to modify and annotate the collation results, to compare witnesses in order to find shared readings or unique readings, to search for a particular reading, or to clarify a reading. Each one of these interactions will be described more in detail in Section 8.4.2.

8.3.2.1 Early Stages of Development

PyCoviz was created in the format of a Jupyter Notebook (see Section 8.3.2.2 below), but it was preceded by a couple of attempts that were ultimately discarded. It may be useful to briefly present those first attempts and the reasons why they were not adopted.

The first interface considered was a relational database. Relational databases have already been used in other projects with the purpose of comparing variant witnesses (Robinson 1989b; Dubuisson and Macé 2006). However, the relational database is not the best model suited to the JSON output from CollateX. It would

require the division of the collation results into a completely different structure made of multiple tables linked together, in order to include all the paratextual elements which appear in the JSON token objects. On the other hand, JSON is a language that follows the conventional data structures of programming languages such as Java, JavaScript, Python, etc. The JSON output from CollateX is therefore more easily manipulated in one of those languages. As a result, I turned to the Python programming language for a better way to handle the collation data and to create an interface.

Python is a high-level programming language which aims to imitate as much as possible natural English language. It is therefore a relatively easy to learn programming language. There is a large user community which can provide help on platforms such as [stackoverflow](https://stackoverflow.com)¹⁰. There is also a large number of third-party modules available, so that it is not necessary to code everything from scratch when a module is already available¹¹. In fact, CollateX is available as a module in Python, which would make it possible to integrate PyCoviz in a collation workflow entirely in Python¹².

The second interface was a Python script to be executed from the command line. The main issue of this command line interface was sharing it with other literary scholars who are not digital humanists but are working with traditional methods of editing. However, the purpose was to help such scholars apply the traditional Lachmannian method of editing to the results of automated collation tools. Sharing and reproducing results were precisely part of the needs of editors identified previously in this chapter.

For instance, I have collaborated with Andrea Balbo from the University of Turin in Italy, who is working on a new edition of Calpurnius Flaccus for the French collection of Classical texts *Les Belles Lettres*. This collaboration was fruitful as it demonstrated the utility of automated collation to a scholar who had no prior experience of digital humanities, and it helped me improve the interface of PyCoviz. With the command line script, we experienced a range of issues related to the operating system (Macintosh versus PC), such as the installation of Python and execution of the script in the command line, as well as character encoding. In addition, providing the adequate support from a distance was challenging, and I needed to supply a complete documentation anticipating everything that could go

¹⁰Stackoverflow is a platform where users can ask coding-related question to the community of developers: <https://stackoverflow.com> (Accessed June 22, 2017).

¹¹<https://www.python.org/about/> (Accessed June 22, 2017).

¹²<https://pypi.python.org/pypi/collatex> (Accessed June 22, 2017).

wrong. The conclusion was that I needed a more intuitive graphical user interface. Instead of creating one from scratch, I decided to use Jupyter Notebooks, which provide ready-made GUI elements and a comprehensive documentation with extensive support for installation on various operating systems.

8.3.2.2 Jupyter Notebook

A Jupyter Notebook is a document format that combines computer code with prose descriptions, and is accessible through a web browser¹³. Notebooks are especially designed to share or publish executable code, which makes it a well suited format for sharing the python script developed for collation visualisation (Kluyver et al. 2016; VanderPlas 2016). The combination of code and prose explanations should help to make the notebook accessible to scholars with little knowledge of coding or Python. Another advantage is that Jupyter Notebooks are well documented, with detailed explanations for installation, and there is a user community from which to get help. In addition, Jupyter Notebooks are becoming popular tools among digital humanists: the notebooks were used for instance by Underwood and Sellers (2015) to study the pace of change in literary standards¹⁴, by the organisers of a CollateX workshop in November 2016¹⁵, by van Zundert (2016) to experiment with ‘Slow Programming and Close Reading’¹⁶, and by Karsdorp (2017) to teach an introduction to Python¹⁷.

A Jupyter Notebook is made of a series of cells, stored in JSON format. There are text cells which contain explanations for users, and code cells. The code cells are numbered from one to thirty, in square brackets. When referring to code cells in the rest of this chapter, I will therefore use the following format: PyCoviz [n]. When the code cells are run and their code is executed more than once, that number might increase. So it should be understood that the numbers quoted in this chapter are the original numbers, before the code is executed more than once.

A Jupyter Notebook integrates elements of GUI with widgets. Widgets are components of a user interface such as buttons, checkboxes or text areas, which allows for user interactions. The JSON results from CollateX are uploaded in PyCoviz and

¹³The interface was previously called IPython notebook, but as many components were not necessarily specific to the Python language, the project was renamed Jupyter since version 4.0 in 2015: <https://blog.jupyter.org/2015/04/15/the-big-split/> (Accessed June 23, 2017).

¹⁴<https://github.com/tedunderwood/paceofchange> (Accessed June 23, 2017).

¹⁵<http://nbviewer.jupyter.org/github/DiXiT-eu/collatex-tutorial/blob/master/INTRO.ipynb> (Accessed June 23, 2017).

¹⁶<https://github.com/jorisvanzundert/reynaert-as-graph> (Accessed June 23, 2017).

¹⁷<http://www.karsdorp.io/python-course/> (Accessed June 23, 2017).

transformed into the Python data format. Then the collation data can be manipulated through various interactions: first, a few functions allow to modify or correct the collation, if the current alignment is not satisfying. Second, the collation can be filtered in order to find agreements between a group of selected witnesses. It is possible to search the collation results and to clarify a reading by displaying all its properties. Finally, some functions let users save the new collation after it has been modified, or save the tables obtained by filtering the collation for agreements. The next section will review the precise description of these interactions.

8.4 PyCoviz: A Python Interactive Interface

In this section, I will describe the interface of PyCoviz in detail (see Appendix B.5 for the files). This section will be divided into four parts: in the first part, the structure of the JSON output from CollateX is outlined, since it may be helpful to understand how it can be modified later with the widgets (Section 8.4.1).

The second part lists all possible interactions available to manipulate the collation results (Section 8.4.2). The interactive aspect of the widgets, and how to use them, as well as the code behind the interactions will both be examined. The code is important for several reasons. The first one is reproducibility, a topic already discussed p. 244 above: as researchers from all kind of fields write code in order to create research outputs, it is necessary for that code to be published, and understood by other researchers (Kluyver et al. 2016). Another reason is that coding is part of a researcher's activities, and therefore it needs to be recognised as true scholarship (van Zundert and Dekker 2017). For code to be criticised properly, it is not only necessary to publish it, but also to comment it: it is not always straightforward to understand what the code does just by reading it. Programmers need to make explicit the assumptions built into code.

In the third part, the Notebook PyCoviz will be applied to actual research questions about the text of Calpurnius Flaccus (Section 8.5). It will demonstrate how the widgets can be used in practice to solve problems faced by editors. The fourth part, finally, will discuss the issues of PyCoviz and its possible improvements (Section 8.6).

PyCoviz was last updated and tested with version 3.6.7 of Python, as well as the packages Jupyter Notebook v5.5.0 and Ipywidgets v7.2.1.

8.4. PyCoviz: A Python Interactive Interface

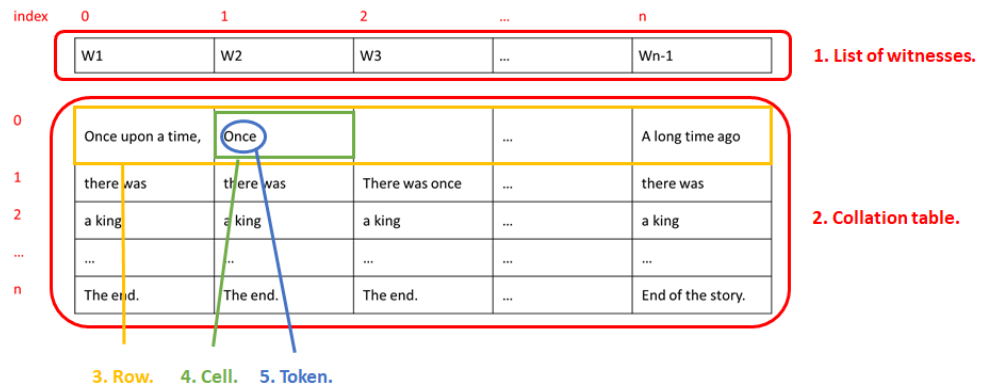


Figure 8.14: Structure of CollateX JSON results.

8.4.1 Understanding the Collation Data Structure

Before starting with the collation manipulation and visualisation, it may be helpful to understand the structure of the data. CollateX output is in JSON format, because this output would retain the additional properties of tokens included in the JSON output (see Section 7.1.2). CollateX JSON results are a complex combination of two simple data structures: arrays, and objects (see also Section 7.1.3)¹⁸. Arrays are an ordered sequence of items: for instance a list of witness sigla. Objects are made of pairs of name and value. For instance, the witness B1 is made of two pairs of name and value: the 'id' with a value of 'B1', and a 'text' which value is a list of tokens.

In figure 8.14, the various elements of the collation results are emphasised: the collation is a JSON object containing two items, (1) a list of witnesses and (2) a list of rows which form the collation table of the aligned text versions. The list of witnesses can be considered as the header for the table. In the case of Calpurnius, the list of witnesses has ten items which are the sigla of each witness.

Each row of the collation table contains in turn a list of cells, one cell for each witness present in the collation results. The cells are made of a list of tokens, a list which can be empty if a witness does not have text that aligns with the other witnesses' text at this point in the collation. Depending on the CollateX output format, tokens can be separated, or joined into segments (see also Section 7.1.4). If tokens are separated, it means that initially there is only one token per cell (although this can be changed in PyCoviz by moving tokens, see below Section 8.4.2.2). If aligned tokens are joined into segments, there can be more than one token per cell. The aligned tokens may be variant: for instance in figure 8.14 'Once upon a time' (4

¹⁸In Python, arrays are called lists, and objects are called dictionaries.

8.4. PyCoviz: A Python Interactive Interface

tokens) is aligned with ‘Once’ (1 token) and ‘A long time ago’ (4 tokens). The aligned tokens may be identical, such as ‘a king’ (2 tokens). However a cell can also contain a very large passage of text with a high number of tokens such as an entire chapter, either because there was no variant in this particular chapter, or because it was missing from one witness.

Tokens are objects with a series of pairs of name and value, such as **t** (the exact word from the manuscript), **n** (a normalised version), and so on. Here is the full hierarchical structure of the collation table:

- the table is a list of rows;
 - a row is a list of cells;
 - * a cell is a list of tokens;
 - a token is an object with the following items:
 - t** exact word as it appears in the witness;
 - n** optional normalised version of the word;
 - locus** optional exact location of the word in the manuscript or edition, a folio or page number followed by a line number;
 - note** optional comment;
 - decl** optional declamation number;
 - link** optional link to the page of a digital facsimile, where the token appears.

The index is the position of an item in a list, starting from zero to n. For the list of ten witnesses in the example of Calpurnius, witness B1 has index zero, while witness P1594 has index nine. The two objects of the collation results, the list of witnesses and the collation table, are not explicitly linked to each other. Instead, witnesses are matched to their respective column of the collation table through their index: the cells with index zero in the table correspond to the text of B1, while the cells with index nine correspond to the text of Pithoeus’ edition. The table is organised by rows instead of columns. This means that to access the text of witness B1, it is necessary to go through each row of the table and select the first cell (the cell with index zero).

The order of the rows is important, because it follows the order of the text. Row 0 has the first word(s) of the text, whereas the last row n has also the last word(s) of

8.4. PyCoviz: A Python Interactive Interface

the text. The row's index will be called ID number, and will be used later for variant location identification, for instance in order to add or delete rows in the table (it is the same number which appears in the HTML collation table described p. 251 above).

8.4.1.1 PyCoviz Structure

The Jupyter Notebook is divided in three sections. The first section takes care of importing the necessary Python modules (PyCoviz [1]) and loading the collation data (PyCoviz [2]). There are a few modules which come automatically with a Python distribution, and modules for the widgets which must be installed¹⁹. The collation data is loaded in two variables: `witnesses` for the list of witnesses, and `collation` for the collation table. In addition, the variable `base_text` is defined with 'LH', the siglum for Håkanson's edition which is used to visualise errors and true readings. The base text variable can be set as an empty string in case the user does not wish to see errors and true readings. The next section, from PyCoviz [3] to [18], regroups a series of functions which are needed for the last section, dedicated to exploring the collation results with widgets. There are widgets for updating the collation result, in PyCoviz [19] to [26], to find agreements among witnesses in PyCoviz [27], to search for and clarify tokens, respectively in PyCoviz [28] and [29]. Finally, a last widget gathers all interactions at the end of the notebook in PyCoviz [30]. We will now examine each widget and interaction available in the notebook.

8.4.2 *Manipulating the Collation Results*

The interactions available in PyCoviz are divided into five categories, as we have seen in figure 8.13 above:

- View an extract from the collation table;
- Modify the collation results;
- Find agreements between witnesses in order to locate shared errors;
- Search for a reading;
- Clarify a reading.

Each of these interactions will be described in the following sections.

¹⁹https://ipywidgets.readthedocs.io/en/stable/user_install.html (Accessed June 23, 2017).

8.4. PyCoviz: A Python Interactive Interface

B1	B2	C1	C2	LH	M1	M2	N1	N2	P1594	ID
quinque cum tyranno	quinque cum tyranno	quinque cum tyranno	quinque cum tyranno	quinque cum tyranno	quinque cum tyranno	quinque cum tyranno	quinque cum tyranno	quinque cum tyranno	Quinque cum tyranno	6
				proximi						7
							familiae			8
proximę	proximę	proximę	proximę		proximae	proximae	proximae	proximae	proximae	9
familię	familię	familię	familię	familiae	familiae	familiae		familiae	familiae	10
puniantur	puniantur	puniantur	puniantur	puniantur	puniantur	puniantur	puniantur	puniantur	puniantur	11

Figure 8.15: Widgets for the selection of a table extract.

8.4.2.1 View of a Collation Table Extract

Before analysing the collation results, it is necessary to visualise them first, and check whether the alignment is accurate or if it must be adjusted. The first interaction lets users select an extract from the collation table (figure 8.15).

Users can select the start and end of the extract by entering numbers in the two text boxes ‘From’ and ‘To’, and the table will automatically be displayed. These numbers are rows ID numbers, or index: for instance, requesting a table from 6 to 11 will show the first sentence of the first *Declamation* from Calpurnius Flaccus. The cells of the table are coloured in red and green again, according to their relationship to the reading of the base text.

The collation table displayed here is simpler than the HTML table described above: it shows only the original forms of tokens, but there are no hidden paratextual elements. It would be complex to integrate customised CSS and Javascript styling to the outputs displayed in a Jupyter notebook. However, this output is perfectly valid for the purpose of checking the alignment. If needed, the additional elements can be visualised with the widget to clarify a token (see Section 8.4.2.5 below).

Code The code executed to transform a collation table into HTML for display is the function `table_to_html` in `PyCoviz` [8]. The code for the table extract selection is located in `Pycoviz` [19]. Thanks to the `interact` function, the table is automatically updated each time a new value is entered in one of the two text boxes ‘From’ and ‘To’. These boxes are widgets which can accept only a limited kind of input: they are bounded to integer numbers. This means that some mechanism is already built in the widget to prevent users from entering wrong values, such as letters or other symbols. In addition, a minimum and maximum value are specified so that it is not possible for users to request rows that do not exist. The maximum is not a fixed number, but a flexible value which is dynamically calculated based on the length of the collation table. This ensures that `PyCoviz` can be reused with a collation of any

given size. Finally, an error message will be displayed if the user selects a wrong combination of ID numbers, e.g. if the second number ‘to’ is lower or equal to the number ‘from’, which returns an empty table.

By visualising the collation results in the table extract, users can spot errors in the alignment. In the example of figure 8.15 above, the reading of Håkanson (LH) in row seven, *proximi*, is not correctly aligned with the other witnesses. It should match *proxime* and *proximae* on row nine, and the subsequently empty row seven should be deleted. In order to correct the alignment, other widgets can be used to move tokens up or down in the collation table or to add and delete rows.

8.4.2.2 Modifications

Corrections and update to the collation table are likely to be necessary, because the alignment algorithm is not perfect. As we have seen in a previous chapter, the algorithm is based on a heuristic method which searches for an acceptable alignment, even if it is not the best alignment (see Section 2.4.4.3). It means that the algorithm makes sometimes arbitrary decisions which may not correspond to what an editor would have chosen. This is why it is crucial to review the collation result and eventually correct it, before starting to analyse it. In PyCoviz, there are four types of interactions related to updating and correcting the collation data:

- Move tokens up or down;
- Add or delete rows;
- Add or delete notes;
- Save the new collation.

Figure 8.16 shows the summary of widgets related to the modification of the collation results. For each modification, the user must choose which part of the collation will be affected, for instance by selecting a row ID or a witness siglum, and press a button that will apply the modification.



8.4. PyCoviz: A Python Interactive Interface

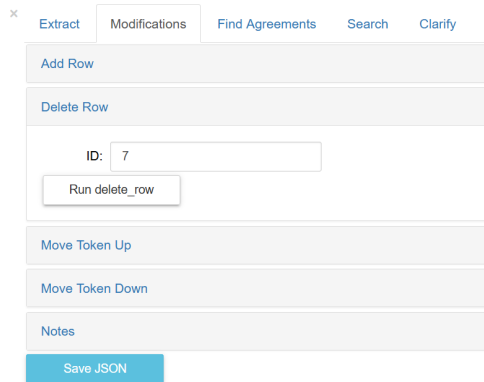


Figure 8.16: Summary of modifications available in PyCoviz.

The next two widgets let users add or delete a row (see figure 8.16 for the ‘delete row’ widget). The user should select a row by entering an ID number in the ‘ID’ text box. Then it is possible to either delete the selected row or add a new empty row after by clicking on a button²⁰. The widget will only delete rows that are empty, as a protection against inadvertently deleting the text of the witnesses.

Code The code for the widgets is in PyCoviz [21] and [22], for adding and deleting rows respectively. The code executed to actually add or delete rows are the functions `add_row_after` and `delete_row` in PyCoviz [14]. Unlike the collation extract widget, the addition or deletion of a row is not performed automatically. Instead, the argument `__manual=True` in the `interact` function ensures that users must first click on a button before the code is executed. The button is created automatically with the option `__manual=True` and cannot be modified: the button label will always be “Run” followed by the name of the Python function.



The next two interactions let users move tokens in the collation table, up or down to the same witness in the previous or next row.

Tokens can be moved only one at a time. In order to preserve the correct text sequence of the witnesses, the last token in a cell can be moved to the first place

²⁰Technically, it is not possible to add a row directly before the first row. If needed, the user should add a row after row 0, and then move tokens down (instead of adding a row before and then moves token up).

8.4. PyCoviz: A Python Interactive Interface

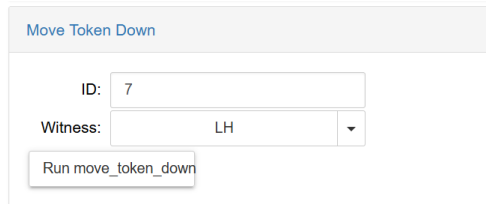


Figure 8.17: Move one token down.

in the next cell, or the first token in a cell can be moved to the last position of the previous cell. This ensures that for instance in witness N1, *familiae* in row eight cannot be moved below *proximae* of row nine, otherwise it would modify the word order of N1 (see figure 8.15 above).

Code The widget in PyCoviz [23] is for moving tokens down, and in PyCoviz [24] to move tokens up. The functions executed are `move_token_down` and `move_token_up` in PyCoviz [13]. The code makes use of the index of the rows (row ID) in order to move tokens up or down the next row (row ID plus or minus one).

For instance, the collation extract that we have seen in figure 8.15 above required correction, in order to align the token *proximi* of LH with the other witnesses. To correct the alignment of *proximi* in figure 8.15, a user need to select both the correct ID number (here seven), and the witness in which the token appears (here LH) in the widget shown in figure 8.17. Then clicking on the button ‘Run `move_token_down`’, the reading *proximi* will be moved to row eight. However this problem is not resolved, since *proximi* is now aligned with *familiae* in N1. Therefore, the user needs to select again the ID number eight, witness LH, and move the token down a second time so that it would be properly aligned with the reading *proximae* in the other witnesses.

Here is another example to illustrate a situation when a user may wish to add a new row, and move a token up. In figure 8.18 below, the editor might want to separate the reading *luxuriosum ob amorem* in LH, row 913, into two separate readings: *ob amorem* would be aligned with *ab amore* of the other witnesses, whereas the conjecture *luxuriosum* adopted by (Håkanson 1978, 27.13) would not be aligned with other readings in existing witnesses²¹.

A new row needs to be created before row 913. As we have seen above, a new empty

²¹The conjecture is actually proposed by Pithoeus in his critical apparatus, but he does not print it in the text: http://www.e-rara.ch/gep_g/content/pageview/1099006 (Accessed June 23, 2017).

8.4. PyCoviz: A Python Interactive Interface

From: 913
To: 915

B1	B2	C1	C2	LH	M1	M2	N1	N2	P1594	ID
ab amore	ab amore	ab amore	ab amore	luxuriosum ob amorem	ab amore	ab amore	ab amore	ab amore	ab amore	9
meretricis abdicavit ad	meretricis abdicavit ad	meretricis abdicavit ad	meretricis abdicavit ad	meretricis abdicavit ad	meretricis abdicavit ad	meretricis abdicavit ad	meretricis abdicavit ad	meretricis abdicavit ad	meretricis abdicavit ad	1
se ad	se ad	se ad	se ad	se ad	se ad	se ad	se ad	se ad	se ad	3
					...					9
										1
										5

Figure 8.18: An alignment that needs to be updated.

row will be inserted after the one selected. Therefore, to add a row before 913, the user needs to add a new row after row 912. Then the new empty row will become row 913, and the reading *luxuriosum ob amorem* will become row 914. When this is done, it is possible to move one token up from row 914 to row 913 in witness LH.



Notes are editorial comments, which were made during the transcription of the witnesses. These notes belong to the category of paratextual elements which are recorded during the collation process (see Section 8.1.1 above) and which an editor may need to have at hand while analysing the collation results. In PyCoviz, Notes can also be added to a specific token or deleted in PyCoviz during the analysis of the collation results, for instance to record comments arising from the comparison of variant readings, or the reason for an editorial conjecture (see figure 8.19).

▼ Notes

Witness: LH

ID: 8

Token posit... 0

Note: Conjecture (cf. Cicero Inv. II 144)]

Add note Delete note

Figure 8.19: Adding a note to a token.

A note can only be attached to one token. As a consequence users must not only indicate a witness and a row ID, but also the token position. The token position is necessary so that if the same word appears twice in the cell, the note will be attached to the right one. The position can be found using the ‘Clarify’ widget (see p. 281). Clicking on the ‘Add note’ button will add the comment to the ‘note’

8.4. PyCoviz: A Python Interactive Interface

property of the token. If the token had already a note, the new note will be added after the pre-existing one. On the other hand, clicking on the ‘Delete note’ button will eliminate the ‘note’ property entirely, including any pre-existing note.

The reason why notes apply only to one token at a time is because of the structure of the collation data: adding notes only inside a token respects the structure of CollateX JSON data and therefore it allows for consistency across the collation files which are loaded into the Jupyter Notebook. A comment could obviously be applicable to several tokens. In this case the best solution for now is to repeat the same note for each token concerned, although it is redundant. In Calpurnius, it is often the case that the same note is repeated between the same tokens of the first and second hand of a witness. For instance, the unknown abbreviation in B (f. 148r) and C (f. 82r) (see Section 6.2.3.1) receives a note to explain that the abbreviation is unknown and that the normalised form *contra* was supplied by the editor Lehnert. This note is the same for this token in the four witnesses B1, B2, C1, and C2.

Code The widgets are created in PyCoviz [25], whereas the code executed to add or delete notes are the functions `add_note` and `del_note` in PyCoviz [15]. The `add_note` function let user add as many notes as needed to one token, by creating the note feature, or appending the string of characters from the ‘Note’ text box to an existing note. It is recommended to end each note with a dot, so as to separate different notes. In PyCoviz, notes are only unstructured plain text which refer to one token only, and users must add manually any detail needed, such as the editor responsible for the note or the date when it was added. Ideally, the token to which the note is attached should be chosen through an ID number instead of a string of characters, as there could be twice the same word in a single cell. However, this would make reuse more difficult, as an ID property is not mandatory in CollateX’s tokens.

The conjecture *proximi* was proposed by Håkanson by comparing the text of Calpurnius to that of Cicero’s *De Inventione*, in paragraph 144 of the second book. Since Cicero’s text support the conjecture of *proximi*, the reference can be added as a note to this token to justify Håkanson’s choice.

The addition or deletion of notes show that indeed no part of the collation results is definitively fixed by CollateX’s algorithm. On the contrary, the editor is free to update any of the existing token properties, to add more information when needed, and correct or refine the alignment according to their own judgement. It is possible to correct any of the other token’s properties such as *t* and *n*, or to add new

8.4. PyCoviz: A Python Interactive Interface

properties. This was not implemented in PyCoviz, but new widgets could easily be added. However, I decided to limit the number of updates available in widgets to the essential ones, so that the notebook does not become too complex.



The Jupyter Notebook does not store any modification to the collation results. The Python data must be transformed back into JSON data and saved in a file, or the changes that were introduced in the collation will be lost once the Notebook is closed, or reloaded. For this reason, there is a widget for saving the modifications in the collation. It is the 'Save JSON' button which will save the Python data back in a JSON file (see figure 8.16 above).

Code The save button is created in PyCoviz [26], and the code executed is the functions `save_json` in PyCoviz [17]. When the button is clicked, a new JSON file is created, the date and time is appended to the filename, so that each file has a unique name and versions are available in chronological order. It is also possible to set up a fixed path, and every time the button is clicked this would save the results in the same file and overwrite the previous results. The advantage of overwriting the results is that there is no need to constantly reload a new collation file (in PyCoviz [2]) in order to have the most up-to-date results.



There are several issues related to the modifications described above, regarding the Rows ID, moving tokens, and reproducing the modifications. In its current form, the ID column is not a perfect solution to refer to rows in the table. The ID is simply the position of the row in the table: row 0 is the first row, row 1 the next, and so on. While it is useful to know the position of a row to add an empty row after, or to move a token to the previous or next row, it also means that rows do not have a fixed reference ID. For instance, the reading of *proximi*, which was moved to row nine in the first example, would then be in row 8 after the elimination of the empty row seven. This can lead to confusion after the collation table is updated, and the ID number then changes for all following rows after an addition or a deletion. The solution would be to add a fixed ID to each row of the table, in a new column. It

8.4. PyCoviz: A Python Interactive Interface

was not implemented in PyCoviz, because it would introduce a modification to the CollateX output format, which could complicate the reuse of data.

Moving tokens one at a time is rather tedious, and even more so when many modifications are needed. In this case, the solution could involve marking the cells that needs a token moved up, for instance with checkboxes, and then apply the `move_token_up` function to all marked cells. However, this solution was not implemented with widgets in the notebook, because the HTML display of the collation table does not allow checkboxes to be included inside at the time of writing. However, from a coding point of view, it would be quite straightforward to apply the function `move_token_up` to a list of cell defined by their row ID and witness.

In addition, updating the collation results manually with the widgets is not a good solution since it makes reproducibility more difficult (Marwick 2015). This currently is an issue of PyCoviz, which could be improved for instance by creating a database of all corrections to the collation results: actions would be recorded in a file every time a button was pressed (which function was executed, with which arguments, on which collation results and when).

After the collation has been corrected, and the alignment is now considered satisfying, users can start to study the collation to find groups of witnesses which agree together in errors, or to find unique errors to a witness. This can be done with the series of widgets described in the next section.

8.4.2.3 Find Agreements

The Agreements widget is central to PyCoviz. This widget lets users filter the collation results to find readings shared by a group of selected witness, and not by other witnesses. There are five elements in the Agreements widget: (1) and (2) dropdown menus to select witnesses, (3) the collation table, (4) a textbox and (5) a save button (figure 8.20).

In the first menu (1), the user selects a list of witnesses to see where they agree together. In the second menu (2), the user can select another list of witnesses. The resulting collation table (3) will show readings where the first witnesses agree together and not with the witnesses in the second list. By default, the second list will contain all the witnesses that are not selected in the first list. The total number of rows is displayed at the bottom of the table, so that it gives an approximation of the number of variants. It is an approximation because transpositions, for instance,

8.4. PyCoviz: A Python Interactive Interface

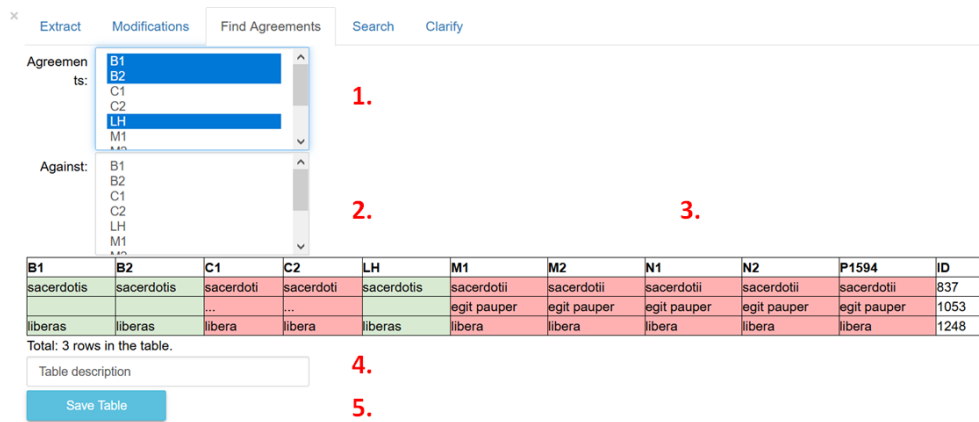


Figure 8.20: The Agreements widget.

appear as two or more rows but usually are considered only one variant.

The collation table can be saved in the HTML format, along with a short description (4), by clicking on the ‘save Table’ button (5). The description is necessary to keep track of how the table was created. Without a description, it may be difficult to remember which witnesses were selected for the comparison, and the HTML table may become meaningless. When the button ‘Save Table’ is clicked, the basic HTML table of the notebook is transformed into the more complex HTML format described previously in this chapter (Section 8.2.2).

Code The Agreements widget is encoded in PyCoviz [27]. The code executed to compare the witnesses is a series of functions: `compare_witnesses` in PyCoviz [6], `compare_multiple_cells_norm` and `compare_cell_norm` in PyCoviz [4] which is calling the function `cell_to_string_norm` in PyCoviz [3] to transform a list of tokens into a string of characters.

Let us examine briefly these functions, starting from the most simple. In PyCoviz [3], the two functions take as input a cell from the collation table, that is a list of tokens, and transform it into a string of characters. `cell_to_string` uses the original form of the tokens, and `cell_to_string_norm` uses the normalised forms *n* when available, or the original form *t* if there is no *n*. In addition, the function does not introduce blank spaces in between tokens. In effect, this means that word division is not taken into account when comparing normalised form. The input of these function is a cell from the collation table, for instance this one from manuscript C:

```
[ 'decl': '24', 'locus': '86v:7', 't': 'eius', 'link': 'https://digi.vatlib.it/view/MSS_
```


`Chig.H.VIII.261/0182', 'decl': '24', 'locus': '86v:7', 't': 'dem&', 'n': 'demet', 'link': 'https://digi.vatlib.it/view/MSS_Chig.H.VIII.261/0182']`

When passed as input to the functions, the outputs will be the following:

Cell to string: eius dem&

Cell to string norm: eiusdemet

The next step in PyCoviz [4] is to compare two or more cells, by comparing their strings of characters generated by the functions in PyCoviz [3]. The function returns 'True' if the strings of characters are equivalent, and 'False' if the strings are different. Let us compare the previous cell of manuscript C with the cell of Håkanson's edition at the same place.

Input:

Cell 1 (C) eius dem& (original) — eiusdemet (normalised)

Cell 2 (LH) eiusdem et (original) — eiusdemet (normalised)

Output:

Compare cell: False

Compare cell norm: True

Their original forms are different because of the word division and the special character '&', so that the function `compare_cell` returns False. But their normalised forms are equivalent: the special character was removed during the transformation of the TEI to JSON (see Section 7.1.3), and the white spaces were removed by the function `cell_to_string_norm`. Therefore, the function `compare_cell_norm` returns True.

The functions `compare_multiple_cells` and `compare_multiple_cells_norm` are very similar, except that they can compare more than two cells. These functions compare each cells to the next in the list, until they find two cells that are different, and return False. If the function reaches the end of the list without finding a difference, then all cells are equivalent and it will return True. For instance, we can compare all the cells in row 31 (figure 8.21). The normalised forms are all equivalent, so that the function `compare_multiple_cell_norm` returns True. But when comparing original forms, the function `compare_multiple_cell` will find a difference between *germanię* (C2) and *Germaniae* (LH). Therefore it will stop the comparison and return False.

8.4. PyCoviz: A Python Interactive Interface

Example: comparing multiple cells

Input

B1	B2	C1	C2	LH	M1	M2	N1	N2	P1594	ID
sunt germaniē	sunt germaniē	sunt germaniē	sunt germaniē	sunt Germaniæ	sunt germaniæ	sunt germaniæ	sunt germaniæ	sunt germaniæ	sunt Germaniæ	31
vultus et flava	vultus et flava	vultus et flava	vultus et flava	vultus et flava	vultus et flava	vultus et flava	vultus & flava	vultus & flava	vultus & flava	
proceritas	proceritas	proceritas	proceritas	proceritas	proceritas	proceritas	proceritas	proceritas	proceritas	

Output

```
compare multiple cell:      False
compare multiple cell norm: True
```

Figure 8.21: PyCoviz [4] - compare multiple cells.

Example: comparing witnesses

Input

B1	B2	C1	C2	LH	M1	M2	N1	N2	P1594	ID
incipiunt ex	incipiunt ex	incipiunt ex	incipiunt ex		incipiunt ex	incipiunt ex	incipiunt ex	incipiunt ex	incipiunt ex	0
calpurnio flacco	calpurnio flacco	calpurnio flacco	calpurnio flacco		calpurnio flacco	calpurnio flacco	calpurnio flacco	calpurnio flacco	Calpurnio Flacco	
excerptę	excerptę	excerptę	excerptę		excerptę	excerptę	excerptę	excerptę	excerptę	1
excerpta	excerpta	excerpta	excerpta							
decem	decem	x	x						x	2
rhetorum	rhetorum	rhetorum	rhetorum						Rhetorum	3
minorum	minorum	minorum	minorum						minorum	

```
First group = ['B1', 'B2']
Second group = ['N1', 'N2']
```

Output

B1	B2	C1	C2	LH	M1	M2	N1	N2	P1594	ID
excerpta	excerpta	excerpta	excerpta							1
decem	decem	x	x						x	2
rhetorum minorum	rhetorum minorum	rhetorum minorum	rhetorum minorum						Rhetorum minorum	3

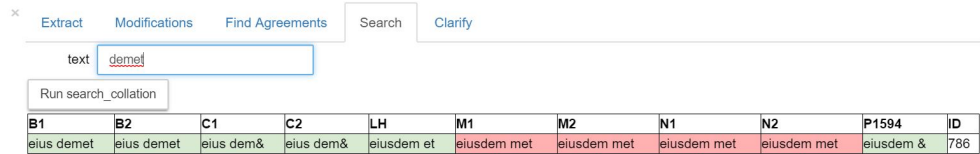
Figure 8.22: PyCoviz [6] - compare witnesses.

In PyCoviz [6] the function `compare_witnesses` goes through an entire collation table and divides each row into two groups of cells. The first group corresponds to witnesses that should agree together and have the same tokens, while the second group corresponds to the other witnesses from the second dropdown menu (or all others by default) which should all be different from the witnesses in the first group. If the two conditions are met, the row is added to the filtered table that will be displayed.

The function `compare_witnesses` takes as input three arguments: a collation table, and two lists of witnesses. For instance, in Appendix B.4, the arguments are a collation table which contains only the incipit of Calpurnius Flaccus (`collation[0:4]`), and two groups of witnesses. The output is a filtered table. It contains only the rows where the witnesses of the first group (B1 and B2) agree together against the witnesses of the second group (N1 and N2).

The notebook's setting is to compare normalised forms with this function `com-`

8.4. PyCoviz: A Python Interactive Interface



B1	B2	C1	C2	LH	M1	M2	N1	N2	P1594	ID
eius demet	eius demet	eius dem&	eius dem&	eiusdem et	eiusdem met	eiusdem met	eiusdem met	eiusdem met	eiusdem &	766

Figure 8.23: The comparison of normalised forms ignores word separation.

pare_witnesses so that orthographic variations and other accidentals are ignored. As a result, the reading *eius dem&* in C is considered equivalent to the reading *eiusdem et* of Håkanson, as we have seen above. This is the reason why it does not appear as an error in red in figure 8.23. As we will see in Section 8.6, it may not be as easy as it seems to compare orthographic variants.

Finally, the table can be saved in HTML format. The code executed is the function `table_to_html_fancy` in PyCoviz [9]. The table is saved in the folder called `alignment-tables`, where the CSS and Javascript necessary to a proper display are also stored (see Section 8.2.2 the description of the HTML tables). The code executed to save a collation table into HTML is the function `save_table` in PyCoviz [18]. The function will take a template prepared for storing results, `template.html` located in the folder `alignment-table`. Then it will insert the HTML table and the description in their predefined place with the help of regular expressions.



There are three different possible use of the Agreements widgets: to find unique readings, to compare two witnesses, or to compare multiple witnesses.

Unique Readings First, selecting only one witness in the first dropdown menu will show the unique readings of the witness in question, when all other witnesses are different from it. Selecting only LH in the first widget would display the readings that Håkanson chose to adopt and which have no manuscript evidence, i.e., conjectures. Interestingly, this will also show a few silent orthographic emendations made by Håkanson, because I have not normalised his readings in the transcription.

It is worth noting here the distinction between unique errors and unique readings. When a manuscript is selected, for instance B1, it will be compared to the base text of Håkanson's edition. As a result, only unique errors according to Håkanson will be displayed. However, another editor might disagree with Håkanson's decisions. In

8.4. PyCoviz: A Python Interactive Interface

The screenshot shows the 'Find Agreements' interface. It has a search bar and a 'Clarify' button. Below are two lists: 'Agreements' and 'Against'. The 'Agreements' list contains C2, LH, M1, M2, N1, N2, and P1594. The 'Against' list contains C2, LH, M1, M2, N1, N2, and P1594. Below the lists is a table with columns B1, B2, C1, C2, LH, M1, M2, N1, N2, P1594, and ID. The table has three rows. Below the table, it says 'Total: 3 rows in the table.' and 'M versus B, C, N, Pitheous'. There is a 'Save Table' button.

B1	B2	C1	C2	LH	M1	M2	N1	N2	P1594	ID
liberis	liberis			bonis	bonis	bonis	liberis	liberis	liberis	860
filia	filia	filia	filia	filia	filiar	filiar	filia	filia	filia	892
								933

Figure 8.24: Comparison of manuscript M against the other manuscripts and Pitheous.

order to see the unique readings of B1, it is necessary to compare B1 only to other manuscripts by excluding the witness LH from the comparison (see Section 8.4.2.3 Comparison of Multiple Witnesses below).

Comparison of Two Witnesses The second way to use the Agreements widget is for the comparison of two witnesses. In this case, the user can select only one witness in each list. This will show the differences between the two witnesses: selecting M1 in the first list and M2 in the second list, for instance, will display the readings where the second hand corrected the first hand in manuscript M. M is actually the manuscript with the smaller number of corrections, whereas B is the manuscript with most corrections (see Section 5.1.2.3), and this can be visualised thanks to the comparisons of first hands against second hands for each witness. The table of M1 versus M2 has 21 rows, whereas B1 versus B2 returns a hundred and ninety-five rows. In addition, it may also be interesting to compare different editions, and see where editors disagree.

Comparison of Several Witnesses Finally, it is possible to select in the first list a group of witnesses that agree together against another group of witnesses. The witnesses in the second group do not necessarily agree together, they are merely different from the witnesses in the first group.

Figure 8.24 shows the unique readings of manuscript M. The table has three rows where both M1 and M2 have the same reading, and B1, B2, C1, C2, N1, N2, and P1594 have a different reading. LH was omitted from the comparison in order to show unique readings, and not only unique errors. It should also be noted that this table shows the unique readings of both M1 and M2. In order to have a really

8.4. PyCoviz: A Python Interactive Interface

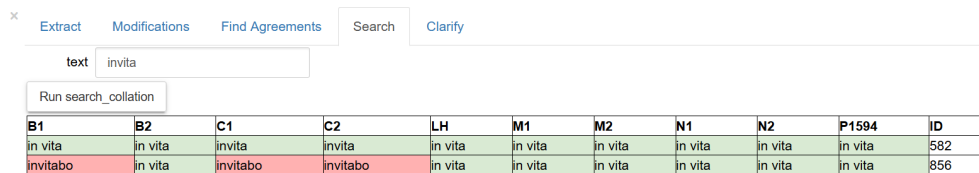


Figure 8.25: The Search widget.

complete picture of unique readings in manuscript M, the unique readings of M1, and then of M2, should also be considered. Unless otherwise specified, the siglum of a manuscript without a hand number refers to both hands (e.g., manuscript M stands for both M1 and M2).

8.4.2.4 Search the Collation

It may happen that the user wishes to find a particular reading, but does not know in which row of the collation it appears, especially since the row IDs change after rows are added or deleted. PyCoviz provides a very simple search widget (figure 8.25).

The widget is located in PyCoviz [28]. The code executed to search the collation for a string of characters is the function `search` in PyCoviz [16]. The widget takes a string of character as input and will search through each cell of each row, for both original or normalised readings. Since all the tokens of a cell are transformed into a string, it means that a search for several words will be successful only if the entire string of character is present in the same cell. If the words are divided into two cells, there will be no result. Thus, searching for the string *in vita bonis* will not yield any result, although it is present in the text (see Section 8.5.1 below). However, searching for *in vita* will show a result of two rows, the second of which also contains the reading *invitabo* that was aligned with *in vita bonis*. The search will return any row where the words are present in at least one witness, independently of the other witnesses' readings. If the text was not found, the function will print an error message.

8.4.2.5 Clarify a reading

The Jupyter notebook displays collation tables in basic HTML, which does not include the hidden paratextual elements available in the collation table described p. 251 above, such as notes or location in the manuscripts. In order to see this information, as well as the normalised form of a token, it is possible to clarify a reading by selecting a row ID and a witness. This will show all the properties of each token in the cell, separated by a comma, and including the link to the digital image when available.

8.4. PyCoviz: A Python Interactive Interface

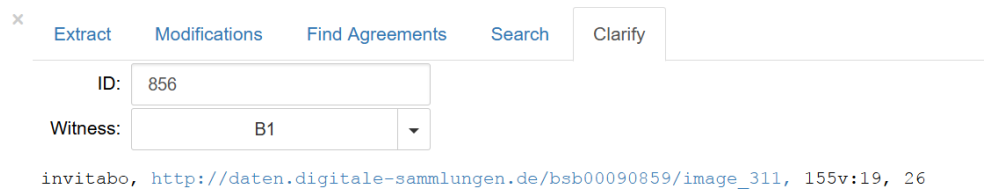


Figure 8.26: The Clarify widget.

Example: print info

Without the function `print_info`, a cell of the collation table is difficult to understand:

```
[{'decl': '0', 'locus': '147r:21', 't': 'incipiunt', 'link': 'http://daten.digitale-sammlungen.de/bsb00090859/image_294'}, {'decl': '0', 'locus': '147r:21', 't': 'ex', 'link': 'http://daten.digitale-sammlungen.de/bsb00090859/image_294'}, {'decl': '0', 'locus': '147r:21', 't': 'calpurnio', 'link': 'http://daten.digitale-sammlungen.de/bsb00090859/image_294'}, {'decl': '0', 'locus': '147r:21', 't': 'flacco', 'link': 'http://daten.digitale-sammlungen.de/bsb00090859/image_294'}, {'decl': '0', 'locus': '147r:21', 't': 'excerptę', 'n': 'excerptę', 'link': 'http://daten.digitale-sammlungen.de/bsb00090859/image_294'}]
```

Input

```
rowID = 0  
witness = B1
```

Output

```
0 : 0, 147r:21, incipiunt, http://daten.digitale-sammlungen.de/bsb00090859/image_294  
1 : 0, 147r:21, ex, http://daten.digitale-sammlungen.de/bsb00090859/image_294  
2 : 0, 147r:21, calpurnio, http://daten.digitale-sammlungen.de/bsb00090859/image_294  
3 : 0, 147r:21, flacco, http://daten.digitale-sammlungen.de/bsb00090859/image_294  
4 : 0, 147r:21, excerptę, excerptę, http://daten.digitale-sammlungen.de/bsb00090859/image_294
```

Figure 8.27: PyCoviz [11] - print info.

The widget is located in PyCoviz [29]. The code executed to display the tokens features is the function `print__info` in PyCoviz [11]. This function takes as input a cell from the collation table, that is expressed as a row ID and a witness. It prints all the information attached to each token of the cell in a legible way. First, the position of the token is printed, as it may be useful to add a note to this token. Then each feature of the token is printed.

8.4.3 Non-Interactive Functions

In the course of preparing this visualisation in PyCoviz, I have created several more functions, which have not been made interactive through the use of widgets, but are nonetheless available to more advanced users. For instance, in PyCoviz [5] the function `find_agreements` may be executed to find the agreements of a group of witnesses, but not against another group. This function shows a table that include the agreements of a group of witnesses when at least one of the other witnesses is different, but it does not matter which one. The many rows where all witnesses agree together are therefore not displayed.

For instance, in figure 8.28, the input is a collation table which contains only the text of the third declamation, and a list of witnesses that agree together (B1, B2). They agree together against all others in row 78, or only against P1594 in row 69, or

8.4. PyCoviz: A Python Interactive Interface

Example: find agreements

Input

```
collation table = collation[66:82] (text of declamation 3)
List of witnesses = ['B1', 'B2']
```

Output

B1	B2	C1	C2	LH	M1	M2	N1	N2	P1594	ID
o	o	o	o		o	o	o	o		67
iam	iam	iam	iam	iam	iam	iam	iam	iam		69
credo	credo	credo	credo	Crede	credo	credo	credo	credo	Crede	72
verginus	verginus	verginus	verginus	Verginius	virginus	virginus	virginus	virginus	Virginus	76
gaudium	gaudium	gladium	gladium	gladium	gladium	gladium	gladium	gladium	gladium	78
tinxisti	tinxisti	cinxisti	cinxisti	tinxisti	tinxisti	tinxisti	tinxisti	tinxisti	tinxisti	80

Figure 8.28: PyCoviz [5] - Find agreements.

Example: view variants

Input

```
collation table = collation[66:82] (text of declamation 3)
```

Output

B1	B2	C1	C2	LH	M1	M2	N1	N2	P1594	ID
o	o	o	o		o	o	o	o		67
iam	iam	iam	iam	iam	iam	iam	iam	iam		69
necat	negat	negat	negat	negat	negat	negat	negat	negat	negat	71
credo	credo	credo	credo	Crede	credo	credo	credo	credo	Crede	72
iudicasse	iudicasset	indicasse	indicasse	iudicasset	iudicasset	iudicasset	iudicasset	iudicasset	iudicasset	74
verginus	verginus	verginus	verginus	Verginius	virginus	virginus	virginus	virginus	Virginus	76
gaudium	gaudium	gladium	gladium	gladium	gladium	gladium	gladium	gladium	gladium	78
tinxisti	tinxisti	cinxisti	cinxisti	tinxisti	tinxisti	tinxisti	tinxisti	tinxisti	tinxisti	80

Figure 8.29: PyCoviz [7] - View variants.

only against C1 and C2 in row 80. It would not be possible to obtain this table with the function `compare_witnesses`.

In PyCoviz [7], the function `view_variants` returns a collation table with all rows where there is at least one variant. The result is in fact the list of all variant locations in the entire collation table. Figure 8.29 shows the list of all variants in declamation 3.

Finally, there is the function `print_witnesses_text` that shows an extract from the collation table in a horizontal text format in PyCoviz [10] (figure 8.30).

The function `get_pos` in PyCoviz [12] is not really necessary for users who decide to use only the interactive widgets. I had included this function in the early stage of development, so that I could combine several collation tables: for instance I experimented with comparing N1 **OR** N2 against P1594 (and not N1 **AND** N2). In that case, I would need to combine three different tables:

8.5. PyCoviz Applied to Calpurnius Flaccus

```
print_witnesses_text(collation[287:292])
```

```
spectatorem tertia orbitas, B1  
spectatorem tertia orbitas, B2  
spectatorem tertia orbitas, C1  
spectatorem tertia orbitas, C2  
spectatorem tertia orbitas, LH  
spectatorem tertia orbitas, M1  
spectatorem tertia orbitas, M2  
tertia orbitas spectatorem, N1  
orbitas tertia spectatorem, N2  
orbitas tertia spectatorem, P1594
```

Figure 8.30: PyCoviz [10] - A short passage in text format.

1. The agreement between N1 and P1594
2. The agreement between N2 and P1594
3. The agreement between N1N2 and P1594

Once the tables are combined, the function `get_pos` would be used as the key to sort the final table according to the position of the row in the complete collation table. As a result the rows appear from the lowest number to the highest, and keep the order of the text. Here is a usage example:

```
print_collation(sorted(P1594_vs_N1orN2, key=get_pos))
```

where `P1594_vs_N1orN2` would be the collation table that combines the three listed above. Although this function is not currently used in the notebook, I left it for more experienced users.

8.5 PyCoviz Applied to Calpurnius Flaccus

To illustrate how PyCoviz can help editors, the textual tradition of Calpurnius Flaccus will be corrected and analysed with the widgets described above. First, the modifications to the collation results will be described, and then we will see two examples of analysis: we will analyse how the edition of Pithoeus relates to the other manuscripts, then we will examine the corrections of manuscript B and see how they may be related to manuscript N.

8.5.1 Updates in Calpurnius Flaccus

In some cases, the collation results from CollateX were clearly incorrect: for instance, *proximi*, a conjecture by Håkanson (1978, 1.2), was not aligned with the reading *proximae* present in the other witnesses. The token was thus moved in

8.5. PyCoviz Applied to Calpurnius Flaccus

B1	B2	C1	C2	LH	M1	M2	N1	N2	P1594	ID
invitabo	in vita	invitabo	invitabo	in vita	in vita	in vita	in vita	in vita	in vita	847
nisi	liberis	nisi	nisi	bonis	bonis	bonis	liberis	liberis	liberis	848

Figure 8.31: Example of corrected alignment.

order to be at the right place. The alignment could also be refined, even if it is not actually wrong: the reading *luxuriosum ob amorem* in Håkanson (see Section 8.4.2.2 above) could not be considered as wrong from the point of view of the algorithm, but is not entirely satisfying from the editor's point of view.

Other situations are less obvious, and the final alignment might be debatable. For instance, the reading *invitabo nisi* in witnesses B1 (f. 155v), C1 and C2 (f. 87r) is closely related to *in vita bonis* found in M1 and M2 (f. 13v). The variant appears to originate mainly from a word division issue: it seems thus that the readings might belong to the same cell. However, witnesses N1, N2, B2 and LH read *in vita liberis*. Therefore, *bonis* was ultimately aligned with *liberis* and *nisi*, even if that means that the letters *-bo* in *invitabo* are not aligned anymore with *bo-* in *bonis* (see figure 8.31). This alignment allows to show the agreement of the witnesses with the reading *in vita*, however another user might decide that the whole group of words should be aligned together: *invitabo nisi* aligned with *in vita bonis* and *in vita liberis*. Although transpositions are usually considered as one variant and could be merged in a single row, I chose to keep the original output so that it could be turned again into a variant graph, which is a better visualisation for transpositions than a table.

Within the Declamations of Calpurnius Flaccus, a total of 171 corrections have been made to the alignment provided by CollateX: these corrections include moving 157 tokens, adding two rows and deleting twelve. Out of the 68,914 tokens from ten witnesses, only 157 were not properly aligned by the algorithm, which represent around 0.23 percent of the total number of tokens. Even considering that some errors or mismatches in the alignment escaped correction, the percentage is unlikely to get higher than 0.5. This low percentage may attest to CollateX's efficiency. However, the ten witnesses of this case study represent a small textual tradition, especially since four of the ten witnesses are actually artificially created by attributing the status of witness to corrections of second hands and thus are very similar to the first hand. In addition, there are few instances of transposition in Calpurnius, and the transpositions that are found usually involve no more than two or three

words. Therefore it is likely that, for texts with a more complex tradition, a higher percentage of errors may arise.

8.5.2 Compare witnesses

8.5.2.1 *Editio Princeps* vs. Manuscripts

As we have seen in Section 5.1.2.6, Pithoeus notes in his critical apparatus that he has based his edition partly on the damaged manuscript A, and on another manuscript from Italy. However, Pithoeus does not give more precision about this Italian manuscript, and C is the only manuscript that is known to have been in Italy. In addition, manuscripts B, M and N are written in italics, an indication of a potential Italian origin. Pithoeus, M and N also have in common the numbering of declamations, and the indications *pro/contra* which explain who the declamator is defending or accusing in the discourse.

With the Agreements widget, we can compare how Pithoeus' edition relates to each manuscript by comparing them together against the other manuscripts. Pithoeus is compared with subsequently B, C, M and N, against the others. We can then see the readings P has in common with each manuscript, and especially if they share common errors. If one witness has significantly more errors in common with Pithoeus, it would be an indication that this may be Pithoeus' Italian exemplar.

The first figure 8.32 shows the agreement of P1594 with manuscript B (that is B1 and B2), against C, M and N. P1594 and B have only two readings in common, *rutili* and *liberas*, which are both considered the correct reading by Håkanson. The comparison of P1594 and C against BMN yields four readings, of which only one is an error according to Håkanson (figure 8.33). However, this reading appears at the very beginning of the text: it is in row 2, which is part of the *incipit*. It is considered an error only because Håkanson did not print the incipit in his critical text. In fact, it is more likely that the error lies with M and N, which both omitted the second part of the *incipit* that was also present in manuscript A: *excerpta X (decem B) rhetorum minorum*, 'extracts of the ten minor rhetors'). P1594 and M have no readings in common.

On the other hand, P1594 and N have seven readings in common, four of which are errors (the first four lines in figure 8.34 represent two transpositions). Since Pithoeus has more readings in common with N against the other manuscripts, Håkanson judged that N was the Italian manuscript of Pithoeus. However, it is interesting to review the unique readings of N that Håkanson (1978, XIII) claims

8.5. PyCoviz Applied to Calpurnius Flaccus

B1	B2	C1	C2	LH	M1	M2	N1	N2	P1594	ID
rutili	rutili	rutili	rutili	rutili	rutili	rutili	rutili	rutili	Rutili	30
liberas	liberas	libera	libera	liberas	libera	libera	libera	libera	liberas	1239

Figure 8.32: Pithoeus and B against CMN.

Pithoeus has adopted:

- *inquit erant* (LH, 7:19)
- *scio me* (LH, 19:4)
- *quacumque* (LH, 20:11)
- *vel* (LH, 22:4)
- *es* (LH, 22:20)
- *dicit* and *remittitur* (LH, 24:11)
- *in vita liberis* (LH, 25:17)
- *pauper* (LH, 31:2)

Three of those readings are present in figure 8.34: *inquit erant*, *scio me* (two words inversions of which only *erant* and *me* are visible in the table), and *dicit*. The readings *vel*, *remittitur* and *in vita liberis* do not appear in figure 8.34 because they are also present in B2, which can be explained by the fact that the readings of B2 have likely been copied from N (see Section 8.5.2.2 below). The readings *pauper* and *quacumque* are not present in figure 8.34 because they are also shared by manuscript M.

In his list, Håkanson omits three unique readings of N which are actually adopted by Pithoeus: *huiusmodi*, *miseræ* and *reddit*. The last line of the table in figure 8.34 is a reading from the *explicit*, which makes more sense in context: N has the reading *finis*, and P1594 has the reading *explicit*. This reading appears in a previous row that is not in the table, because it is not an agreement of P1594 and N. However both N and P1594 omit the last part of the *explicit* (*explicitæ ex Calpurnio Flacco excerptæ*).

Although the analysis of the shared errors of Pithoeus' edition shows that the closest manuscript is N, it is worth checking the readings quoted by Håkanson. There are

8.5. PyCoviz Applied to Calpurnius Flaccus

B1	B2	C1	C2	LH	M1	M2	N1	N2	P1594	ID
decem	decem	x	x						x	2
quedam	quedam				quaedam	quaedam	quaedam	quaedam		11
si	si	sic	sic	sic	si	si	si	si	sic	203
oculum	oculum	oculorum	oculorum	oculorum	oculum	oculum	oculum	oculum	oculorum	1145

Figure 8.33: Pithoeus and C against BMN.

B1	B2	C1	C2	LH	M1	M2	N1	N2	P1594	ID
erant	erant	erant	erant	erant	erant	erant				205
							erant	erant	erant	207
me	me	me	me	me	me	me				614
							me	me	me	616
dixit	dixit	dixit	dixit	dicit	dixit	dixit	dicit	dicit	dicit	810
eiusmodi	eiusmodi	eiusmodi	eiusmodi	eiusmodi	eiusmodi	eiusmodi	huiusmodi	huiusmodi	huiusmodi	971
miserere	miserere	miserere	miserere	miserere	miserere	miserere	miserere	miserere	miserere	1017
reddidit	reddidit	reddidit	reddidit	reddidit	reddidit	reddidit	reddidit	reddidit	reddidit	1334
explicitae ex calpurnio flacco excerptae	explicitae ex calpurnio flacco excerptae	explicitae ex calpurnio flacco excerptae	explicitae ex calpurnio flacco excerptae		explicitae ex calpurnio flacco excerptae	explicitae ex calpurnio flacco excerptae				1356

Figure 8.34: Pithoeus and N against BCM.

small discrepancies between Håkanson's list and the evidence from the witnesses. The last step in this analysis is to consider if Pithoeus did in fact use N as an exemplar, or if it is possible that manuscript N was copied from Pithoeus. In this case, the unique errors from both witnesses must be examined.

To find unique errors, witnesses are selected only from the first dropdown menu. There is one unique reading shared by N1 and N2, which could have been easily corrected: *at ego dico meus es* (N) versus *est* in other witnesses ('But I say, he is / you are mine'). There is also only one unique error of N2: *eris aevum* instead of *eris in aevum* ('you will be for the rest of your life'). Again, this could easily have been corrected. All unique errors of N1 have been corrected by N2, so that it is difficult to draw any conclusion from these errors. However it can be noted that one error of N1 *tam malae* (f. 247v:7) was corrected to *male* which is a reading present in B, C1 and M but not in P1594. On the other hand, there are forty-two unique errors in P1594. Six of these are word inversions; one is part of the explicit, and is an error only because Håkanson did not print the explicit in his text; finally, two are different readings from the 1580 first print of the Pithoeus' edition, and may have been misprinted in the 1594 edition. With this information at hand, it seems more plausible that Pithoeus used N2 as his exemplar and corrected a few errors. The copyist of N is less likely to have been able to correct all the errors (or conjectures) introduced in P1594.

8.5. PyCoviz Applied to Calpurnius Flaccus

8.5.2.2 Corrections of the second hand in Manuscript B

As noted in Section 5.1.2.3 above, the corrections of the second hand in manuscript B, B2, have been likely copied from manuscript N (Håkanson 1978, XI; Lehnert 1903, XIII). In that case again, it is possible to use the Agreements widget to compare the corrections of B2 against the other witnesses. First, comparing the readings of B1 against B2 will show the list of corrections by B2, a table with 197 rows: from this we can see for instance that B2 has added the *pro/contra* notes that introduce most discourses and which are present in M, N, and Pithoeus edition (see Section 5.1.2.4). This means that either M, N, or Pithoeus, are the likely source of B2's corrections. In fact, the corrections in B2 share 135 readings with MNP1594 against C; but C and B2 share only one reading against MNP1594.

Therefore, we need to analyse the corrections of B2 and see if it is possible to find agreements with one particular witness. The large majority of B2's corrections are in fact also common to M, N and Pithoeus (comparing the readings that are common to B2, M, N, P1594 against B1 shows a table of 175 rows). These variant readings cannot help to decide which one is the source of B2's correction, but we can focus on the 22 other corrections of B2.

- Eight are unique readings of B2, some of which may have been misread from N, e.g., *inultam* instead of *multam* (see Håkanson 1978, XII). It is also possible that the correction of *commenta* into *comuenta* came from *conventa* in N, especially since it is followed by *est* one of the four readings in figure 8.35.
- Four readings are common to B2, M and N1, where the correction of N2 correspond to B1 (see figure 8.35 below). Because of the reading *patruo* in particular, and the absence of lacuna after *calpurnius*, it seems that there has been an exchange of readings between B and N. Three of these four readings are also shared by P1594.
- Four readings are common to B2, N and P1594, but not to M. Three of these readings are errors which convinced Håkanson that M could not be the source of B2's corrections (see Håkanson 1978, XII).
- Two readings are common to B2, M and N, but not to P1594.
- Four are more isolated cases: the correction *arcem* (f. 151v:10) corresponds to manuscript C but not to *arce* in M, N, and P1594; *addictus* (f. 151v:12) corresponds to M, N2 and P1594, but not to N1 (which reads *abdicator*); the

8.5. PyCoviz Applied to Calpurnius Flaccus

B1	B2	C1	C2	LH	M1	M2	N1	N2	P1594	ID
indicare	vindicare	indicare	indicare	indicare	vindicare	vindicare	vindicare	indicare	indicare	171
calpurnius	abdicatorum patrum liberans	calpurnius ...	calpurnius ...	Abdicatorum patrum liberans	abdicatorum patrum liberans	abdicatorum patrum liberans	abdicatorum patrum liberans	calpurnius	Abdicatorum patrum liberans	441
patruo	avo	avo	avo	avo	avo	avo	avo	patruo	avo	725
es	est	es	es	es	est	est	est	es	est	1119

Figure 8.35: Four readings are maybe exchanged between B and N.

last ones are orthographic differences (*immolanda* on f. 158v:21, which is written *imolanda* in M1 and *ephebo* on f. 158r:23 written *ephoebo* in M1).

The fact that the three witnesses M, N and P1594 are closely related makes it more difficult to identify with certainty the source of B2's corrections. From M, N and P1594, there is not one witness which shares a correction with B2 and against all the others. The examination of the 22 readings listed above shows that B2 has slightly more in common with manuscript N: for instance there are twelve shared readings, of which seven are errors (table 8.1 below). The three errors which are absent from M, in particular, make B2 closer to N. This link between B2 and N is reinforced by the likely exchange of readings (see figure 8.35, and Section 5.1.2.5 for the comparison of the hands which made corrections in manuscript B and N).

	M1	M2	N1	N2	P1594
B2 Shared Readings	7 (+2 orth.)	9	12	9 (+4) [†]	10
B2 Shared errors	2	4	7	5 (+2) [†]	4

[†] (when N2 is likely different only because copied from B1)

Table 8.1: Relations of B2 with M, N, and P1594.

8.5.3 Conclusion

The section demonstrated how it is possible to use the PyCoviz notebook for the correction, analysis and visualisation of the collation results of Calpurnius Flaccus. The JSON output from CollateX was corrected, using the widgets. Although most errors of alignment were obvious, it was necessary also to make decisions regarding more difficult cases such as for the variants *invitabo nisi/in vita bonis/in vita liberis*, or to decide how to visualise transpositions.

Then the collation was analysed with the Agreements widget: we examined two questions of the tradition, namely the relationship between the editio princeps of Pithoeus with the other manuscripts, and the origin of the corrections in B2. We

demonstrated how to make Håkanson's research reproducible, using his edition as a base text in order to examine shared errors. Although the tables produced with the widgets did not bring a new conclusion regarding the tradition of Calpurnius Flaccus, it made it possible to spot small errors in Håkanson's argument. By choosing to view shared readings instead of shared errors, users can make sure that they have the full evidence for making a decision, instead of relying only on Håkanson's classification of errors and true readings.

8.6 Discussion

8.6.1 Issue of normalisation

The collation tables in PyCoviz are all made by comparing normalised forms of tokens whenever possible. In practice, the orthographical differences are thus excluded from the collation tables. However useful it may seem, the use of normalised forms for the analysis of collation results may be problematic in certain circumstances. Comparing only normalised forms may hide variant readings that could be considered significant to an editor.

For instance, in folio 83r, manuscript C reads *liniamentis*, while the other witnesses read *lineamentis*. This is a purely orthographic difference, and as such, does not appear in the collation table as a place of variation since the comparison is done on the normalised tokens. A user of the Jupyter Notebook, selecting witnesses in order to find their (dis)agreements, would not see this row in the results for any combination of groups of witnesses, as it does not contain any other variant. However, Håkanson (1978, 7) included this orthographic difference in his critical apparatus. Håkanson therefore considered this difference to be somehow significant, but in PyCoviz it would be nearly invisible. For this reason, the notebook also provides a set of functions to compare readings in their original forms, i.e., using tokens (t) instead of their normalised forms (n).

Nevertheless, this approach is not a satisfying solution, because CollateX's results used here have consecutive matching tokens joined into segments. As a consequence, some large chunks of texts are combined into a single cell of the table when there is no difference between normalised forms. And if there is an orthographic difference, it becomes hard to spot it in the middle of a long block of text: the word *liniamentis* mentioned above appears in the middle of a 37-word reading that shows other orthographic variations (figure 8.36). Comparing orthographic differences is therefore difficult.

B1	B2	C1	C2	LH	M1	M2	N1	N2	P1594	ID
nunquam	nunquam	nunquam	nunquam	Numquam	numquam	numquam	nunquam	nunquam	Nunquam	
iudices contra	iudices contra	iudices contra	iudices contra	iudices contra	iudices contra	iudices contra	iudices contra	iudices contra	iudices contra	
istum tutior	istum tutior	istum tutior	istum tutior	istum tutior	istum tutior	istum tutior	istum tutior	istum tutior	istum tutior	
veni quicquid	veni quicquid	veni quicquid	veni quicquid	veni quicquid	veni quicquid	veni quicquid	veni quicquid	veni quicquid	veni quicquid	
aufferi potuit	aufferi potuit	aufferi potuit	aufferi potuit	aufferi potuit	aufferi potuit	aufferi potuit	aufferi potuit	aufferi potuit	aufferi potuit	
amisi soli	amisi soli	amisi soli	amisi soli	amisi Soli	amisi soli	amisi soli	amisi soli	amisi soli	amisi soli	
omnium torti	omnium torti	omnium torti	omnium torti	omnium torti	omnium torti	omnium torti	omnium torti	omnium torti	omnium torti	
sunt donec	sunt donec	sunt donec	sunt donec	sunt donec	sunt donec	sunt donec	sunt donec	sunt donec	sunt donec	
mentirentur ita	mentirentur ita	mentirentur ita	mentirentur ita	mentirentur Ita	mentirentur ita	mentirentur ita	mentirentur ita	mentirentur ita	mentirentur Ita	
laniatos	laniatos	laniatos	laniatos	laniatos	laniatos	laniatos	laniatos	laniatos	laniatos	
miseros ita	miseros ita	miseros ita	miseros ita	miseros ita	miseros ita	miseros ita	miseros ita	miseros ita	miseros ita	
confusis	confusis	confusis	confusis	confusis	confusis	confusis	confusis	confusis	confusis	
lineamentis	lineamentis	lineamentis	lineamentis	lineamentis	lineamentis	lineamentis	lineamentis	lineamentis	lineamentis	
proici iussit ut	proici iussit ut	proici iussit ut	proici iussit ut	proici iussit ut	proici iussit ut	proici iussit ut	proici iussit ut	proici iussit ut	proici iussit ut	
iam nec pater	iam nec pater	iam nec pater	iam nec pater	iam nec pater	iam nec pater	iam nec pater	iam nec pater	iam nec pater	iam nec pater	
posset	posset	posset	posset	posset	posset	posset	posset	posset	posset	
agnoscere	agnoscere	agnoscere	agnoscere	agnoscere	agnoscere	agnoscere	agnoscere	agnoscere	agnoscere	
miramini si ab	miramini si ab	miramini si ab	miramini si ab	miramini si ab	miramini si ab	miramini si ab	miramini si ab	miramini si ab	Miramini si ab	
hostibus	hostibus	hostibus	hostibus	hostibus	hostibus	hostibus	hostibus	hostibus	hostibus	
sepulti sunt	sepulti sunt	sepulti sunt	sepulti sunt	hostibus sepulti	hostibus sepulti	hostibus sepulti	hostibus sepulti	hostibus sepulti	hostibus sepulti	
nullos	nullos	nullos	nullos	sunt nullos	sunt nullos	sunt nullos	sunt nullos	sunt nullos	sunt nullos	

Figure 8.36: The reading *liniamentis* in the collation results.

CollateX also provides results with consecutive matching tokens not joined into segments (see Section 7.1). In this case, however, the information that some groups of tokens should be considered together is lost, especially when one token of one witness matches with several tokens in another witness. It is a fairly common situation in Latin, since texts used to be written in the *scriptio continua* style, without word division (see Section 6.2.3.2). Inconsistencies in word division are quite frequent in Calpurnius: *republicam* versus *rem publicam*, *contradicit* versus *contra dicit*, and so on. Some cases are more complex than just difference in word division, such as *verberantibus* in B1 corrected into *verbera cibus* by B2 (f. 148v), where one word matches two different words of another witness. In fact, these situations were so frequent that they were not normalised in the TEI transcriptions. Instead, it was decided to compare normalised forms without spaces in between words, so that word division would not be considered a variant: PyCoviz will consider for instance that witnesses with the readings *eius demet* (BC) and *eiusdem et* (LH, P1594) agree together.

There are so many instances of one-to-many matching tokens that correcting CollateX results, when matching tokens are not joined into segments, would not be worth the effort. The two examples of figure 8.37 show how the alignment of tokens is more accurate when tokens are joined into segments (figure 8.37(a)) rather than separated (figure 8.37(b)). There are two differences in word division, between *ante/apte* and *ad te*, and between *republicam* and *rem publicam*. In the first example, the tokens are correctly aligned. However, when the output has separated tokens, the alignment is much less precise: instead of aligning *ad te* with *ante*, *ad* is aligned with *ante*, and *te* is in the following row. The reading *republicam* is misaligned with *te*, instead of either *rem* or *publicam*.

8.6. Discussion

B1	B2	C1	C2	LH	M1	M2	N1	N2	P1594	ID
gloria non	gloria non	gloria non	gloria non	gloria Non	gloria non	gloria non	gloria non	gloria non	gloria Non	
ad te	apte	ad te	ad te	ante	ante	ante	ante	ante	ante	492
rem publicam	rem publicam	rem publicam	rem publicam	rem publicam	rem publicam	rem publicam	republicam	republicam	Rempublicam	493

(a) Joint Tokens.

non	non	non	non	Non	non	non	non	non	Non	2584
ad	apte	ad	ad	ante	ante	ante	ante	ante	ante	2585
te		te	te				republicam	republicam	Rempublicam	2586
rem	rem	rem	rem	rem	rem	rem				2587
publicam	publicam	publicam	publicam	publicam	publicam	publicam				2588

(b) Separated Tokens.

Figure 8.37: Joint vs. separated tokens outputs.

Any attempt to toggle between the two results of CollateX, with and without joined segments, is bound to be difficult because of these many places where there is no one-to-one matching token. The solution would be to normalise tokens after collation. However, this solution may require a lot of efforts for traditions with countless orthographic variations, such as medieval texts for instance, for which normalisation would precisely improve the collation results. One solution may be an iterative process of collation and normalisation such as in the Collation Editor: ‘[t]he regularisation stage of the Workspace is iterative, with the editor making a series of regularisation and then re-collating the text. The process of regularisation often improves the results from CollateX and makes the remaining stages of the editing process more straightforward’ (Houghton and Smith 2016, 120). In the case of Calpurnius, I had included the normalisation directly into the transcriptions so as to be able to test the different tools with both normalised and non-normalised tokens. Since I needed to collate multiple times, the normalisations that I would include into a CollateX output would be lost for other tools.

8.6.2 Reuse

The notebook was designed with reuse in mind. Consequently, it should be able to run with any CollateX result that has at least tokens (t). All other features of the tokens are optional. It is not necessary either to choose a base text and view readings as errors or true readings. There is no limitation regarding the number of witnesses present in the collation, however the number of witnesses that can be comfortably visualised is limited by the size of the computer screen.

A sample of collation kindly provided by Hayim Lapin, the editor of the Digital Mishnah project, was used to verify the possible reuse of PyCoviz with a different CollateX output. The example of figure 8.38 shows a selection of agreements between some of the sixteen witnesses from the sample. No base text is selected, and readings are not marked with green or red. Although the JSON sample has a different format from my own JSON collation, and it does not contain notes,

P00001.xml	P00002.x ml	S00483.x ml	S00651.x ml	S01520.x ml	S03524.x ml	S04589.x ml	S04624.x ml	S07106.x ml	S07204.x ml	S07319.x ml	S07326.x ml	S08174.x ml	Vat115.x ml	Vat117.x ml	Vilna.x ml	I D
יתן	יתן	יתן	יתן	יתננה		יתן		יתן	יסול	יתן	יתן	יתן	יתן		יתן	1 1#
יתן	יתן	יתן	יתננה	יתננה				יתן	יסול	יתן	יתן	ייתן	יתן	יתן	יתן	2 3#
דבר	דבר	דבר	דבר	דבר		ל		דבר	דבר	דבר	דבר	דבר	דבר	דבר	דבר	4 6#

Figure 8.38: Example of a collation table from a Digital Mishnah sample.

location, links or declamation numbers, this does not prevent from using the interactions available in PyCoviz. From this experience, it appeared that there is an advantage to displaying witnesses in columns, instead of rows like all the other table visualisations (see Section 8.2 above). The advantage is that there is no need to worry about the direction of writing. A collation table organised in columns can display both left-to-right (Latin) or right-to-left text (Hebrew).

In its current state, PyCoviz still requires users to be comfortable with a minimum of code, in order to change some variables such as the input collation file or the base text. It is also helpful to know about CollateX and its JSON output formats. If PyCoviz was to become truly user-friendly for all scholars and not only Digital Humanists, modifications could be implemented, for instance adding widgets for the choice of input file and base text, and adapting button labels ('Save JSON' could be changed to 'Save corrections'). To be truly user-friendly for non DH users, however, and avoid all mention of JSON altogether, it would mean a deeper transformation, starting already from the stage when the transcription is transformed with XSLT. It would also mean an integration with a user-friendly CollateX interface, such as the Workspace for collaborative Editing. At this point it would probably not stay in a Jupyter notebook, but it would become something entirely different.

8.6.3 Version Control and Reproducibility

The adoption of a Jupyter notebook has advantages: it provides a user interface with interactive widgets, and explanations along with the code. It is a great tool to share code and to make it available for other scholars to examine and review, and eventually adapt it to their own needs. Nonetheless, there are a few inconvenient aspects of coding with a Jupyter Notebook. Version control is notably challenging because the Notebook is saved in JSON format, which is not practical for visualising changes in the code or in the code's output. The presentation of code in blocks, without line numbers, can make it difficult to refer to a precise portion of code. Finally, it may be complicated to keep track of which version of the code produced which collation tables, based on which collation file (obtained from which version of CollateX). For instance, there were overall not many mistakes in the CollateX

alignment, but there were many more mistakes in previous collations obtained with former versions of CollateX. It is not an issue related to Jupyter notebooks only, but to any output obtained with computational method. This kind of information is crucial for the reproducibility of materials obtained with computational methods, as well as for quoting those results in publications. At the moment, there is no way to record this information automatically in PyCoviz. However, a possible solution to this issue could be to use a Python module such as ReciPy: every time a Python script is run, ReciPy records in a database the input and output files, as well as the version of the code²².

8.6.4 Further Improvements

In its current state, PyCoviz is still a prototype with limitations. This section summarises the various improvements discussed in this chapter, and which could be implemented in PyCoviz in the future. Some improvements are technical enhancements, such as:

- Adding unique IDs to each row of the collation table;
- Moving large amounts of tokens at once;
- Improving the search results;
- Saving files via a dialog box;
- Using ReciPy to keep track of the input and output files, and the version of the code.

Other improvements are related to the visualisation method, such as letting users choose a lemma for each row in the table, or visualising uncertainty. Editors have to choose, at each variant location, the reading that will be printed in the critical text. However, this decision may be subject to doubt, often expressed in the critical apparatus. Håkanson (1978) did so in several occasions, for instance with expressions such as *fortasse* ('maybe', p. 9), *lacunam suspicor* ('I suspect a lacuna', p. 9), *fort. recte* ('perhaps rightly', p. 26). Håkanson also expressed doubts in his preface with question marks next to supposed true readings. Visualising uncertainty could

²²<https://github.com/recipy/recipy> (Accessed September 4, 2017).

be achieved by introducing a new colour beside red and green, to distinguish a different level of certainty²³.

Dealing with incomplete witnesses, such as manuscript A in Calpurnius Flaccus, is another issue that requires changes to the Agreements widget. The problem was that the collation results did not make a difference between the different reasons why the text is missing from the results: there are lacunae due to illegible text in A, and there are lacunae due to missing folia. In other witnesses, there are empty cells when the text is absent for this witness but not in others: for instance, declamation 45 is present only in manuscript C and in Håkanson's edition, but not in other witnesses, although there is no damage to the manuscript, and the text is perfectly legible.

The lacunae in A were represented by empty cells in the collation table. However, this would skew the results when looking for agreements. For instance, looking for agreement between A and other witnesses would return the rows where A is actually non-existent, and other witnesses happen to have an empty cell. The same problem appears when comparing other witnesses against A: the search would yield many results of agreements against A, when the witnesses agree together and A has an empty cell. This does not mean that the text of A was really different. We don't know because the manuscript evidence simply does not exist for A at this point of the text.

What should happen when a user is searching for agreements? Should the search stop when the last token of A is reached? This would need an adaptation of the Agreement widget and the collation results. First, the comparison process should ignore a witness after its last token was reached. Secondly, the collation table should also indicate explicitly the state of a witness, whether the manuscript is extant or not.

8.7 Conclusion

In summary, we have identified three specific needs of editors and readers who wish to study collation results: visualising paratextual elements, discovering common errors or unique errors in order to apply Lachmann's method, and sharing their conclusions with others who should be able to reproduce the reasoning.

²³Green and red were used for this demonstration, as they are colours commonly associated to the idea of true or false. However, it may not be the best set of colours for an inclusive design that should also take into account users with colour-blindness issues.

The visualisation of paratextual elements was implemented in the HTML collation tables. Editorial notes or comments related to tokens can be displayed, as well as the location of tokens in their respective witnesses. When available, a link to a digital facsimile allows for others to check in the manuscripts if the transcription is correct, or to see difficult passages such as lacunae directly in the source material. Although these paratextual elements are not visible in the collation tables of the Jupyter notebook, they are always accessible through the widget for clarifying a reading.

The application of Lachmann's method to the collation results is made possible via the use of colours, and the Agreements widget. First, the use of red and green colours serves to distinguish between errors and true readings, according to a selected base text. In parallel, the use of the Agreements widget let users filter the collation results in order to find readings common to a group of witnesses or readings unique to one witness. These readings can then be analysed to make inferences about the witnesses relationships. However, it is important to keep in mind as well how the comparison is performed in the Notebook to find shared readings, and how normalisation impacts the collation tables obtained when searching for agreements of witnesses. The comparison of normalised forms, for instance, means that only substantial differences will be displayed, but not orthographic differences or changes in word division. Errors and true readings should also be carefully considered as the product of an editor's judgement on the text, in this case the decisions of Håkanson.

In this case we have been able to reproduce Håkanson's conclusions, but it is possible that other editors will reach different conclusions. By sharing the JSON collation file and using the PyCoviz notebook, users are able to share their results and let other scholars reproduce the tables that stand behind their edition. However, users need to be able to document which table is created from which collation file, and with which version of the Python code. It should also be noted that critical decisions are integrated into the code: for instance the decision about what makes a significant variant and what can be ignored (such as word division), or the choice of the base text which will generate the distinction between errors in red and true readings in green. The code is more than just technical, it requires also the editor's point of view. As a critical edition is considered to be an argument about a text, it is possible to view the PyCoviz interface and its code as an argument about the process of editing. For instance, the inclusion of a base text in the comparison, even as an optional parameter, makes an argument about how to edit the text: it may be a valid argument for texts such as the *Declamations*, which are edited following

the Neo-Lachmannian method, but less so for texts studied from the perspective of genetic criticism, such as Beckett's works.

The case study of Calpurnius Flaccus was based on a particular collation format obtained with an automated collation tool, CollateX. Nevertheless, the methodology behind the creation of this visualisation is focusing on an editor's needs, rather than the method behind the creation of the collation file. Therefore, this kind of visualisation should be applicable as well to collation prepared manually and in different digital formats. For example, the same visualisations could be obtained from an edition encoded in TEI P5 with `<app>` elements to encode variants.

Conclusion

IN this section, I wanted to test different collation tools, and determine which were most useful in the creation of a digital critical edition. I transcribed the witnesses of the Declamations of Calpurnius Flaccus, and used the XML transcriptions to generate the inputs for automated collation in different formats. I then compared the three tools CollateX, Juxta and the Classical Text Editor. The outcomes were twofold: first, the selection of CollateX as the most convenient tool, and second, the creation of a visualisation tool to help editors analyse the collation results.

While testing the different tools, it became apparent that there were many possible combinations of input format, collation tool, algorithm options and output format, all of which would influence the quality of the collation results with regard to accuracy, visualisation and ease of further processing. As a whole, CollateX and Juxta's results were similarly accurate, although a few combinations of input and output formats could produce useless results in CollateX. CTE was less accurate at first, but later modifications to the algorithm improved the results. Juxta's visualisations were the most user-friendly, and CTE's results more suitable for the preparation of a printed edition. CollateX's results were more flexible for further processing, which made it more interesting.

The conclusion of this comparison is that editors need to know in advance what they wish to do in order to choose the best tool. However, this can be hampered by the constant evolution of collation tools: all three tools evolved during this dissertation, and at least five new tools became available between 2014 and 2017. This highlights the need for tool criticism to help scholars select the most appropriate tool. The framework proposed in Chapter 2 was therefore a useful guidance in order to compare the three collation tools, and to better understand the potential of other recent tools, such as LERA for instance.

Comparing the various output formats available, the CollateX JSON output was chosen in order to create a new visualisation tool with Python: PyCoviz. The advan-

tage of CollateX's JSON output was the possibility to include additional information associated to words in the text (such as their position in the manuscript, or an editorial comment), after the transformation of words into tokens. The JSON output was also easily manipulated with the Python script, in order to filter the collation results and visualise sets of variants particularly useful to editors, such as the agreements of groups of witnesses in order to establish the relationships between witnesses.

General Conclusion

THE purpose of the thesis was to study the application of automated collation to a Latin text, and examine both the theoretical and practical aspects. I set out to answer questions such as: what is automated collation? How does it differ from the traditional method? What may discourage scholars from adopting it? What are the existing collation tools, and how are they different from each other? How can their results help scholars in the creation of a digital critical edition?



In the first part of the thesis, I have approached the topic from a theoretical perspective. In Chapter 1, I started with a presentation of traditional manual collation: its definition, purpose, and its methodology. The important point highlighted in this chapter was that collation is a critical activity. Moreover, what is recorded in a collation should not be dependent on the medium (paper versus electronic files).

Then in Chapter 2, I have presented automated collation in detail: how its name, definition and purpose have evolved over time, as researchers distanced themselves from the manual method and the goal of a printed edition, turning instead to the alignment algorithms and output formats flexible enough for different purposes. The methodology was also examined, especially with respect to the main differences between manual and automated collation, i.e., transcription and the use of a base text.

A survey of collation tools revealed that researchers are still actively updating existing tools and creating new ones. As a result, it appeared that users would greatly benefit from a framework to compare and criticise tools, in order to choose the most adapted to their own needs. Therefore I selected a set of criteria to

compare collation tools, inspired by the field of tool criticism (Chambers et al. 2017) and previous surveys (such as Huculak and Richardson 2013). The purpose of these criteria was not to make a definitive judgement about the tools, but to provide the information necessary to select an appropriate tool in a given context. For instance, I have chosen CollateX, but others have preferred a tool with a more user-friendly interface, or wanted a different algorithm. In practice, these criteria have proven useful to understand the new tools which were created in the years after I started this dissertation, and to compare collation tools (see Chapter 7).

In Chapter 3, I have focused more particularly on the issues of transcription, which is a necessary step for automated collation, but is also often advanced as an argument against it. The comparison of transcription with collation, thanks to an extended model of transcription (Huitfeldt, Marcoux, and Sperberg-McQueen 2008), showed that transcription and manual collation are very similar activities. It also highlighted a more subtle difference between how scholars think of manual collation (a database of variant readings) and automated collation (alignment of texts with the insertion of gaps). This conceptual difference may drive some scholars away from automated collation, as they do not believe it will respond to their needs. The analysis of transcription issues also led to other questions such as what makes a witness, and how text is processed into tokens, the units of textual comparison.

Finally, inspired by my experience with CollateX, I have proposed to model readings, a technical term of textual criticism which refers to the content of the text at a given point, as tokens with various properties (Chapter 4). The purpose of this model is to facilitate the selection of a relevant set of variants, depending on the context of the textual tradition and the purpose of the edition. The properties **t** and **n** of CollateX tokens are a promising mechanism to discriminate between substantive and accidental variation for instance. As only simple theoretical examples were considered in this chapter, it would be interesting to test further the application of this model on a real text and with additional properties besides **t** and **n**. While I wanted to experiment with the text of Calpurnius Flaccus, I have faced visualisation issues (see Chapter 8). Another question raised by this model was the representation of tokens, and collation, for non-textual elements. The concept of tokens could be applied in the future for instance to mathematical diagrams, as the collation of diagrams is desirable to improve our knowledge of the traditions of scientific texts (Saito 2006).



In the second part of the thesis, I took the text of the Latin author Calpurnius Flaccus as a case study to test different collation tools and identify how their results can concretely help creating a digital scholarly edition. In Chapter 5, I described the tradition of the *Declamations* of Calpurnius, and the method I followed from transcription to collation, and visualisation of the collation results.

The transcription encoding with TEI XML was described in Chapter 6, whereas in Chapter 7, I compared the tools CollateX, Juxta and the Classical Text Editor (CTE). The accuracy of the collation results was not significantly affected by the three collation algorithms, except for the first test with CTE which was markedly less accurate than the others. This chapter highlighted how each tool has evolved even as I was working, which illustrates well the need to be able to compare tools with the criteria proposed in Chapter 2. The workflow issue was also mentioned: since the workflow can be divided between multiple tasks (such as transcription, collation, visualisation) that must be performed with different tools, a seamless workflow is important in the context of digital editing (Barabucci and Fischer 2017).

Each tool examined had different advantages: CTE would be more relevant to scholars typesetting a printed edition, which was not the focus of this thesis. Juxta had the most user-friendly graphical interface that provided great visualisations. On the other hand, Juxta had some limitations for scholars in the process of editing a text, since the collation results are not easily manipulated within Juxta. CollateX was the preferred tool for this thesis, because of the flexibility of its input and output. As a result, I have used CollateX JSON output to explore the collation of Calpurnius Flaccus in the last chapter.

Scholars have expressed doubt about the ability of collation tools to support traditional textual criticism (see Chapter 3). In Chapter 8, I began by assessing the needs of editors, and then I described a visualisation tool that I have created, showing how CollateX's results can help the analysis of Calpurnius Flaccus. It was used for instance to explore the relationships between witnesses such as the *editio princeps* and the manuscripts, or the origins of corrections by a second hand in manuscript B. Although the case study did not yield new insights into the tradition of Calpurnius Flaccus, it helped reproducing Håkanson's conclusion. The purpose was not to create a critical edition, but to exemplify how the digital visualisation can help

editors in their task.

Chapter 8 underlined the importance of taking into account the traditional Neo-Lachmannian method in order to create dynamic visualisations. This chapter also demonstrated how a digital methodology did not deprive editors from applying their individual judgement. However, users of computer programmes should be aware of the implicit decisions which are embedded in the code of such programmes. Finally, the chapter also discussed the importance of good practices for digital scholarship which should allow for sharing, reproducing, and reusing research.



In conclusion, I would like to come back to the claim that automated collation has changed our understanding of collation. As we have discussed previously, the change from manual collation to automated collation led scholars to think of collation not as a database of variants but more as the alignment of textual versions. This has in turn broadened the scope of automated collation to include not only the search for variant readings, but also instances of text reuse (Yousef and Palladino 2016). The new methodology of transcription followed by automated collation may not imply less work than the traditional method, but its results provide much more flexibility with reuse and visualisation options (Houghton and Smith 2016). This was exemplified with the tool created for this dissertation.

PART III

APPENDICES

THIS Appendix to the Theory part contains two items: a list of automated collation tools, which was discussed especially in Chapter 2, and the file for the collation model of Chapter 3.

A.1 List of tools – Automated Collation

Here is a list of all collation tools that I have found in scholarly literature, most of which are discussed in Chapter 2. They are presented in chronological order, usually according to the first available publication which describes a working program. For some programs, it can be difficult to pinpoint a date of creation, such as for TUSTEP: in this case I have chosen the date of publication of the first edition created with the help of TUSTEP.

A list of sources is included, where other publications referring to the program may appear. When available, the name of the program is also added, as well as short history (who created the program, to edit which text, and so on). Finally, there may be a comment on the program's status, mentioning for instance if the program is still in development, is open source or under a commercial license, and links to pages where the program can be obtained for the most recent tools.

This list of automated collation tools already contains over 25 programs, created between 1960 and 2016, and it is likely not exhaustive. At least eleven of these tools are available today. Some programs have been much more successful than others, as the sources can attest, but the continuous efforts from scholars all over the world shows that, despite its issues and criticism, automated collation is an attractive solution to the comparison of multiple textual witnesses during the critical editing of a work.

1962 Dearing (1), Methods of Textual Editing.

Sources: Dearing 1962, Dearing 1970.

History: the program was created in the spring of 1962 by computer scientist Ronald Bland, for an IBM 7090 (Dearing 1962, 1). He was helped by Dearing's friend Charles Hobb, and Dearing admits that neither was paid for their work (Dearing 1962, 18). This is the first known description of a collation program in a scholarly publication. It was created to collate the work of John Dryden, an English poet and playwright of the 17th century.

1967 Maretti and Zarri, Collatio Codicum: An Exercise in COMIT Programming.

Sources: Maretti and Zarri 1967, Boretti 2009.

History: in this contribution, Maretti and Zarri (1967) do not give any explanation on the program's history. According to Boretti (2009, 11), a student of the University of Pisa, Zarri used this program to collate a manuscript of the Latin poet Catullus.

1968 Froger, La critique des textes et son automatisation.

Sources: Froger 1966, Froger 1968, Froger 1970, Gilbert 1973, Duplacy and Huret 1977, Hockey 1980.

History: Froger started to work on automated collation in 1960 already. The collation program was created by Ms. Renaud at the Company Bull G. E. and was used to collate a short Latin text of the 6th Century AD (Froger 1966). At the time, there was no other existing program for automated collation. By 1966, however, technological progress had already rendered the program obsolete (Froger 1966). While the 1968 essay is the most important contribution of Froger, his article of 1966 is much more detailed on how the program actually works.

1968 Silva and Bellamy, Some Procedures and Programmes for Processing Language Data.

Sources: Gilbert 1973, Hockey 1980, Schmidt 2013.

Name: EDIT.

1970 Dearing (2), Computer Aids to Editing the text of Dryden.

Sources: Dearing 1970, Gilbert 1973, Stringer and Vilberg 1987, Hockey 1980, Shillingsburg 1996.

History: this is a new version of the program developed in 1962. Because of a change in computer material on the campus of the University of California where Dearing was working, a new version was required to accommodate the new machines. It was written in Fortran by Richard Bandat (Dearing 1970, 260).

1970 Gibson and Petty, Project OCCULT: The Ordered Computer Collation of Unprepared Literary Text.

Sources: Gibson and Petty 1970, Widmann 1971a, Gilbert 1973, Hockey 1980, Marín 1991.

Name: OCCULT.

History: the work on a collation program started in 1964 to collate the works from 19th century American writers with more accuracy than when it was done by scholars, or more often by graduate students (Gibson and Petty 1970, 280). The program was designed to collate prose texts and was first tested with two very different versions of *Daisy Miller*, a short novel by Henry James, which served as an extreme case study to check the program's accuracy. However, the results were not satisfying, in part because the communication between literary scholars and computer programmers was not always successful. In a second attempt, Petty wrote himself the program, to be make sure that the decisions they made as textual scholars 'would not be circumvented when they were translated into machine code' (Gibson and Petty 1970, 288).

1970 Cabaniss, Using the Computer for Text Collation.

Sources: Cabaniss 1970, Widmann 1971a, Gilbert 1973, Hockey 1980, Marín 1991.

History: along with OCCULT (Gibson and Petty 1970), it was one of the first program to propose full text collation without limitation of the text's length, and not only line-by-line collation as was done for poetry (Cabaniss 1970, 1). The program was not designed for a specific textual tradition, but it was tested with extracts from the French translation of *De Proprietatibus Rerum*, a Latin text written around 1230 by the Franciscan friar Bartholomeus Anglicus and translated in French in 1372 (Cabaniss 1970, 10).

1971 Widmann, The computer in historical collation: use of the IBM360/75 in collating multiple editions of A Midsummer Night's Dream.

Sources: Widmann 1971a, Widmann 1971b, Gilbert 1973, Hockey 1980.

Name: FORMAT.

History: Widmann adopted an automated collation workflow in order to prepare a new Variorum Edition of Shakespeare's *A Midsummer Night's Dream*, and to compare 80 to 120 printed editions of the play (Widmann 1971b, 57).

1972 Tübingen System von Textverarbeitungs-Programmen.

Sources: Ott 1989, Ott 1991, Kopp, Küster, and Ott 2000, Raabe 2008a, Boretti 2009, Huculak and Richardson 2013, Andrews 2014b, Nyhan and Flinn 2016

ITUG website¹, TUSTEP website².

Name: TUSTEP, TXSTEP.

Status: the program is still in development, and is available to download from the TUSTEP website. At the time of writing, the latest versions are 2017 for TUSTEP, and February 2017 for TXSTEP. Since 2011, TUSTEP is open source and under a revised BSD license.

History: the Tübingen System of Text-Processing Programs was the result of a series of projects carried out at in the University of Tübingen, Germany. Its conception started shortly after the *Zentrum für Datenverarbeitung* (ZDV) was created in 1966, and was further developed after a center for processing literary documents (LDDV) was created in 1970 and lead by Wilhelm Ott (Ott 1991). The first edition using TUSTEP was published in 1972, and the program earned its surname ‘TUSTEP’ in 1978 (ITUG website). In 2010, Tobias Ott presented a new XML version, TXSTEP, intended to make the program more accessible to a wider audience through an English interface, and the addition of XML files among the possible input formats (TUSTEP website).

1973 Gilbert, Automatic Collation: A Technique for Medieval Texts.

Sources: Gilbert 1973, Gilbert 1974, Gilbert 1979, Hockey 1980, Marín 1991.

Name: COLLATE.

History: the program was designed to prepare a critical edition of a medieval prose text with a limited manuscript tradition, the *Quaestiones super libros Metaphysicae* of Johannes Buridanus, at the University of Manitoba (Gilbert 1973, 144). The program had to deal with a medieval commentary, which raised specific challenges: in particular a technical and repetitive vocabulary, and the recursive format of questions followed by answers (Gilbert 1973, 245). COLLATE was an entire editing system comprising several programs in a modular structure, only part of which was dedicated to collation Gilbert (1974).

1978 Shillingsburg, Computer Assistance to Scholarly Editing.

Sources: Shillingsburg 1978, Shillingsburg 1980, Marín 1991, Siemens 1994, Shillingsburg 1996, Raabe 2008a.

Name: PC-CASE, MAC-CASE.

History: PC-CASE is an integrated system of editing with a computer, designed for ‘very long prose texts for which there are up to eight variant texts’ (Shillingsburg 1996, 144). There was also a hypertext version for a Macintosh

¹<http://www.itug.de/> (Accessed May 24, 2017).

²http://www.tustep.uni-tuebingen.de/tustep_eng.html (Accessed May 24, 2017).

operating system, MAC-CASE (Shillingsburg 1996, 139). The program was created by Susann Folett and Russel Kegley for the Thackeray edition at the Mississippi State University (Shillingsburg 1978, 453).

1987 Stringer and Vilberg, The Donne Variorum Textual Collation Program.

Sources: Stringer and Vilberg 1987, Greetham 1994, Siemens 1994, Shillingsburg 1996, Raabe 2008a, Huculak and Richardson 2013, Donne Variorum website³.

Name: Coll60, DV-COLL.

Status: the program is still available to download from the Donne Variorum website, but it runs on Windows operating systems up to Windows 7 only. The software is freely available under the GNU General Public License.

History: the tool was originally created for the Donne Variorum project, by the computer support personnel at The University of Southern Mississippi and other consultants (Donne Variorum website). John Donne was an English poet of the seventeenth century, and his work has a large tradition: about 87 manuscript sources and close to 400 print versions of individual poems (Stringer and Vilberg 1987). Stringer and Vilberg benefited from the work of Vinton Dearing and his colleagues, who shared with them the code of two collation programs (Stringer and Vilberg 1987, 85). DV-COLL is the most recent version of the program, which was adapted from its predecessor Coll60 (see the Donne Variorum website for more details).

1989 Robinson, The Collation and Textual Criticism of Icelandic Manuscripts (part 1 and 2).

Sources: Robinson 1989a, Robinson 1989b, Hilton 1992, Robinson 1994, Hockey 2000, Reeve 2000, Robinson 2007b, Raabe 2008a, Boretti 2009, Robinson 2009, Andrews 2014b.

Name: Collate (0-1-2).

History: the history of Collate and its following developments are described extensively in a blog post by Peter Robinson (2007b). The first version (Collate 0) was created to collate and edit forty-four manuscripts of an Old Norse poem, the *Svipdagsmal* saga, as part of his PhD thesis (Robinson 1989a). This first program was inspired by Gilbert's work on collation and in particular her use of the computer to further manipulate collation results (Robinson 1991, 87, note 24). Later Robinson updated the program to a new version released in 1991 (Collate 1), with a graphical user interface that would make the program available to other researchers (Robinson 1994). A new release

³<http://donnevariorum.tamu.edu/toolsandresources/collation-software/> (Accessed May 19, 2017).

of Collate in 1996 was labeled ‘Collate 2’ (see also the history of collate in Section 2.3). Chaucer’s *Wife of Bath* prologue from the *Canterbury Tales* as well as Dante’s *Monarchia*, among other texts, were edited with Collate (Robinson 2009). Robinson then started to design a new version that would handle XML (Robinson 2007b), but this finally became a new collation tool, CollateX, which will be described later.

1989 Cannon and Oakman, Interactive Collation on a Micro-Computer: the URICA! Approach.

Sources: Cannon and Oakman 1989, Marín 1991, Hilton 1992, Shillingsburg 1996.

Name: URICA!, URICA! II.

History: the impulse to create a new collation tool was given by Father Green at St. Bonaventure University (New York) in 1984, when he started a new edition of Duns Scotus, a Scottish theologian of the 13th century. He had at his disposal one printed text and ten manuscripts in Latin that needed to be transcribed, and then collated, and he turned for help to the University of South Carolina (Cannon and Oakman 1989). The URICA! System was then adopted by other scholars and it was installed in thirty-five different sites according to Hilton (1992). While the first version was available on PC, the second one is designed for Macintosh and brings improvements on several points such as the interface or the possibility to perform semi-automatic collation with well integrated user interactions (Hilton 1992).

1989 Marín and García, Requisitos para la edición crítica informatizada: UNITE. Conjunto de programas para la Unificación automática de Textos. Versión para microordenadores SUN.

Sources: Marín and García 1989, Marín 1991, Siemens 1994, Shillingsburg 1996.

Name: UNITE.

History: UNITE is a set of tools for editors to collate up to 6 versions of a text (Marín 1991, 110). As a test case, it was used on the *Libro de Alexandre*, a Castilian poem of the 13th century. However, UNITE was not designed for a specific textual tradition (Marín 1991, 109).

1994 Horton, Sequence Comparison and Old-Spelling Texts.

Sources: Horton 1994.

History: Horton created two collation programs, in order to test algorithms. Horton compared the efficiency of different algorithms, collating two version of Shakespeare’s play *Love’s Labours Lost*.

2000 Salemans, *Building Stemmas with the Computer in a Cladistic, Neo-Lachmannian, way: the Case of Fourteen Text Versions of *Lanseloet van Denemerken*.*

Sources: Salemans 2000.

History: in the context of his PhD dissertation, Salemans created a suite of tools to build a stemma for the text of *Lanseloet van Denemerken*. Among these tools seems to have been a collation program. The tools are described in appendix A on the CD-ROM which accompanies the thesis (Salemans 2000, 106).

2002 Urbina et al., *Critical Editing in the Digital Age: Informatics and Humanities Research.*

Sources: Urbina et al. 2002, Monroy et al. 2002.

Name: Collator.

History: The Collator program was created at the Cervantes Project digital library, for an electronic Variorum Edition of *Don Quixote de la Mancha* (Urbina et al. 2002; Monroy et al. 2002). It is a part of a larger editing system with several modules, the Multi-Variant Editor for Documents (MVED).

2005 Juxta.

Sources: Raabe 2008c, Raabe 2008a, Boretti 2009, Huculak and Richardson 2013, TEI Consortium eds. 2013, Prebor 2013, Kingsley 2014, Wheelles and Jensen 2014, Zeevaert 2015, Yousef, Palladino, and Crane 2017, Juxta Software website⁴, Juxta Commons website⁵, Juxta Editions website⁶.

Name: Juxta (desktop application), Juxta (Web Service), Juxta Commons, Juxta Editions.

Status: the desktop application and web service on which Juxta Commons is built are both open source on Github⁷.

History: Juxta was originally developed in 2005 as a desktop application by the Applied Research in Patacriticism (ARP) group⁸ at the University of Virginia (Wheelles and Jensen 2014). The project was then taken up by the scholarly organisation NINES⁹ and the company Performant Software, and in 2012 the Juxta Commons interface was released. In 2015, a new interface

⁴<http://www.juxtasoftware.org> (Accessed May 19, 2017).

⁵<http://juxtacommons.org/> (Accessed May 19, 2017).

⁶<http://juxtaeditions.com/> (Accessed May 19, 2017).

⁷<https://github.com/performant-software/juxta-desktop> and <https://github.com/performant-software/juxta-service> (Accessed May 19, 2017).

⁸ARP was a Digital Humanities lab run by Jerome McGann and Johanna Drucker at the University of Virginia: https://en.wikipedia.org/wiki/Applied_Research_in_Patacriticism (Accessed May 14, 2017).

⁹See <http://www.nines.org> (Accessed May 19, 2017).

called Juxta Editions was released, with different subscription plans (a basic free account, or paid subscriptions).

2009 Andrews, Prolegomena to a critical edition of the Chronicle of Matthew of Edessa, with a Discussion of Computer-Aided Methods Used to Edit the Text.

Sources: Andrews 2009, Huculak and Richardson 2013.

Name: Encritic, ncritic, Text::TEI::Collate.

Status: open source and available on Github¹⁰, as well as a Perl CPAN module¹¹.

History: Andrews created the program Encritic for her PhD thesis, to edit an Armenian medieval text (Andrews 2009).

2009 Schmidt and Colomb, A data structure for representing multi-version texts online.

Sources: Schmidt and Colomb 2009, Schmidt 2009, Schmidt 2010, Dekker and Middell 2011, Schmidt 2013, Schmidt and Eggert 2015, Jänicke et al. 2015, Ecdosis website¹².

Name: nmerge, Compare.

Status: available from a Google Code repository¹³. The code is also available on the Ecdosis Github page¹⁴.

History: The tool was created especially in reaction to the use of embedded markup such as XML TEI, which makes it difficult to deal with overlapping hierarchies (Schmidt and Colomb 2009; Schmidt 2010). First created at the Loyola University Chicago's Center for Textual Studies and Digital Humanities, for the HRIT project (Huculak and Richardson 2013), nmerge is supported by the AustESE project and is now hosted on Ecdosis¹⁵. It was used to create the Charles Harpur Critical Archive (Schmidt and Eggert 2015).

2009 TEI Comparator.

Sources: Cummings and Mittelbach 2011, Cummings 2013, Huculak and Richardson 2013.

¹⁰<https://github.com/tla/ncritic> (Accessed May 22, 2017).

¹¹<https://metacpan.org/pod/Text::TEI::Collate> (Accessed May 22, 2017).

¹²<http://ecdosis.net/main/> (Accessed June 13, 2017).

¹³The last update dates back from July 2011. <https://code.google.com/archive/p/multiversiondocs/> (Accessed May 29, 2017).

¹⁴<https://github.com/Ecdosis> (Accessed May 27, 2017).

¹⁵See also: <http://hritwiki.ctsdh.luc.edu/home> for HRIT (Accessed May 29, 2017). AustESE is the Australian Electronic Scholarly Editing: <http://www.itee.uq.edu.au/eresearch/projects/austese/> (Accessed May 29, 2017). Ecdosis is 'is a set of tools for digitising cultural heritage texts for the Web and ebooks': <http://ecdosis.net/> (Accessed May 27, 2017).

Name: TEI Comparator.

Status: available from the TEI comparator website¹⁶, under GNU General Public License version 3.0.

History: The TEI Comparator was developed at Oxford University by James Cummings, Sebastian Rahtz, and Arno Mittelbach, for the Holinshed project. The purpose was to compare two versions, encoded in XML TEI files, of the *Chronicles of England, Scotland, and Ireland*, a collaborative work of historiography published in 1577 and 1587 (Cummings 2013). The TEI-Comparator was launched in 2009, and its primary goal was to create links between the two texts at paragraph level (Cummings and Mittelbach 2011).

2010 Roelli and Bachmann, Petrus Alfonsi or On the mutual benefit of traditional and computerised Stemmatology.

Sources: Roelli and Bachmann 2010, Roelli 2014.

Status: unpublished. May be available from Philipp Roelli upon request.

History: it was developed for the edition of Petrus Alfonsi's *Dialogus*, a medieval text with a large manuscript tradition (Roelli and Bachmann 2010). The program written in Perl is meant to be used directly with the phylogenetic software Phylip, to create a stemma (Roelli, private correspondence).

2011 Dekker and Middell, Computer-Supported Collation with CollateX: Managing Textual Variance in an Environment with Varying Requirements.

Sources: Dekker and Middell 2011, Raabe 2014, Dekker et al. 2015, Raabe 2015, Jänicke et al. 2015, Yousef, Palladino, and Crane 2017, CollateX website¹⁷.

Name: CollateX.

Status: Open source, the code is available on Github under the licence GNU General Public version 3¹⁸. Still under development, CollateX is version 1.7.1 at the time of writing.

History: the project of creating a successor for Peter Robinson's Collate started in 2010 (CollateX website). CollateX development was led by Ronald Dekker and Gregor Middell, and it was funded by Interedition, a COST Action promoting the interoperability of tools in the the field of scholarly digital editing¹⁹. Although the project started as rewriting a version for Collate, the new CollateX took a different approach and is now a separate tool. Collate was a complete system of scholarly editing, from collation of textual witnesses to

¹⁶<http://tei-comparator.sourceforge.net/> (Accessed May 29, 2017).

¹⁷<http://collatex.net> (Accessed May 29, 2017).

¹⁸<https://github.com/interedition/collatex/> (Accessed May 29, 2017).

¹⁹<http://interedition.eu/> (Accessed May 29, 2017).

the preparation of a critical text with apparatus, wrapped in a graphical user interface. On the other hand, CollateX focusses rather on the alignment of various versions of a text, and is built upon the interoperability principle, so as to be embedded easily within other software systems and to be flexible enough to deal with the challenges of different textual traditions (CollateX website). The prototype of CollateX, a Java based application, was presented in 2011 (Dekker and Middell 2011). In 2014, the first Python version was released with a new collation algorithm²⁰.

2015 Jänicke et al., TRAViz: A Visualization for Variant Graphs.

Sources: Jänicke, Geßner, and Marco 2014, Jänicke, Büchler, and Scheuermann 2014, Jänicke et al. 2015, Yousef, Palladino, and Crane 2017, TRAViz website²¹.

Name: Sentence Alignment Flows, TRAViz.

Status: open source, the code is available from Stefan Jänicke's Github repository²². It is licensed under the Fair Academic License (FAL).

History: the first version of the tool, created within the Digital Humanities project eTRACES, was called Sentence Alignment Flows, and it was designed to improve the readability of graph visualisations such as the ones produced by CollateX (Jänicke, Büchler, and Scheuermann 2014). According to TRAViz website, the tool can align multiple editions of a text and generate an interactive graph visualisation.

2015 Hagel, Classical Text Editor (CTE).

Sources: Hagel 2007, Boretti 2009, CTE help manual²³, CTE website²⁴.

Name: Classical Text Editor.

Status: under commercial license, a 30-day trial version can be downloaded. Version 9.03 at the time of writing.

History: Originally a tool for text editing in which collation was done by hand, version 9.0 of the CTE introduced the feature of automated collation in February 2015. Little information is available about how the collation works, since the software is under commercial license. Hagel declared that his algorithm is not meant to deal with large scale transpositions (private correspondence). On the other hand, 'what may be more innovative is its robustness against orthographic variation, which [was] achieved by integrating Levenshtein comparison with a longest-common-sequence comparison on

²⁰<https://pypi.python.org/pypi/collatex/2.1.2> (Accessed May 29, 2017).

²¹<http://www.traviz.vizcovery.org/index.html> (Accessed May 29, 2017).

²²<https://github.com/stjaenicke/TRAViz> (Accessed May 29, 2017).

²³The CTE help manual is accessible only from within the software.

²⁴<http://cte.oeaw.ac.at/?id0=main> (Accessed May 29, 2017).

the level of words. In this way, it may even correctly compare texts where all words show orthographical divergence (including different approaches to word separation at places)’ (private email from Stefan Hagel, 2017).

2015 Chaudhuri et al., Collation: Prabhed and its Predecessors.

Sources: Chaudhuri 2015.

Name: Prabhed.

History: The work on this collation program actually started around 2007, when Juxta was the most advanced program available. The development of the two previous collation programs (*Tafat* and *Pathantar*), leading to Prabhed, is described in Chaudhuri (2015). Prabhed and its predecessors were used to collate works of the Indian poet Rabindranath Tagore (1861 - 1941). However, at the time when the work started, no program offered a proper handling for the non-Latin fonts of the Bengali language, the script used by Tagore (Chaudhuri 2015, 99). The collation tool is integrated within the digital critical edition *Bichitra: Online Tagore Variorum*²⁵.

2015 SaDa—Semi-automatische Differenzanalyse von komplexen Textvarianten.

Sources: Bremer et al. 2015, né Gießler Medek et al. 2015, Schütz and Pöckelmann 2016, SaDa website²⁶.

Names: LERA and LAKomp.

Status: demo versions available online for LERA²⁷ and LAKomp²⁸.

History: SaDa is a project carried out between 2012 and 2015 at the Martin Luther University of Halle-Wittenberg. The aim of the project was the analysis of differences in texts with variants, and the visualisation and analysis of the results. The project focused on two kinds of texts: medieval or early-modern texts with a rich tradition not yet sorted out, or texts with complex adaptations which resulted very different versions that may have each their own reception and tradition (SaDa website). For these kinds of texts, two collation tools were developed, LERA and LAKomp. The latter is specifically designed for earlier languages (such as Early New High German) and provides annotation facility for part-of-speech tagging prior to collation.

2016 Yousef and Palladino, iAligner: A tool for syntax-based intra-language text alignment.

Sources: Yousef and Palladino 2016, Yousef, Palladino, and Crane 2017.

Name: iAligner.

²⁵http://bichitra.jdvu.ac.in/bichitra_collation_guide.php (Accessed May 29, 2017).

²⁶<http://www.informatik.uni-halle.de/sada> (Accessed August 1, 2017).

²⁷<https://lera.uzi.uni-halle.de/?lang=en> (Accessed August 1, 2017).

²⁸<https://lakomp.uzi.uni-halle.de/> (Accessed August 1, 2017).

Status: the code is available on Github²⁹, and a limited demo version is also accessible online³⁰.

History: this recent tool was presented as a poster at the fifth AIUCD Annual Conference in Venice. It was developed at the Digital Humanities department of the University of Leipzig. ‘The main aim of the tool is to facilitate various degrees of textual comparison: in critical editorial practice, it allows the detection of manuscript variants across several witnesses, including non-literal variants in instances of textual re-use’ (Yousef and Palladino 2016).

2016 Schubert et al., eComparatio - Editionsvergleich.

Sources: Schubert et al. 2016, Yousef, Palladino, and Crane 2017.

Name: eComparatio.

History: presented at the DH2016 conference, this tool is designed to compare various editions of ancient texts and create a born-digital critical apparatus. It is primarily a tool for visualisation (Schubert et al. 2016). eComparatio is a project developed in cooperation between the Chair for Ancient History of the University of Leipzig and the ICE (Interdisciplinary Center of E-Humanities in History and Social Sciences) at the University of Erfurt.

²⁹https://github.com/OpenGreekAndLatin/ILA_python (Accessed December 10, 2017).

³⁰<http://www.dh.uni-leipzig.de/tools/Alignment/> (Accessed July 31, 2017).

A.2 Collation Model

Here is the Alloy file of the collation model presented in Chapter 3. The file was used with the Alloy Analyser 4.2³¹ to generate instances of the model. It is available as well here: <https://github.com/enury/phd-automated-collation/blob/master/collation-model/collation.als>.

```

1 module transcription/collation
2 open util/relation as relation
3 //-----
4 //TYPE
5 sig Type {}
6 //-----
7 //TOKEN
8 sig Token {
9     rtt: Reading->Type,
10    d: one Document
11    }{
12    //changes to the initial readings model
13    //to prevent tokens that are not mapped to a type
14    //every token is a token in some reading of the document.
15    some r : Reading | this in elems[r.tokenseq]
16
17    //A token is in document D iff every reading that includes that token
18    //is a reading of document D.
19    all r : Reading | this in elems[r.tokenseq] implies r.doc = d
20    all r: Reading | this in elems [r.tokenseq] iff r.doc = d
21    all r: Reading | all t: Type | r->t in rtt iff this->t in r.tt
22 }
23 //-----
24 //DOCUMENT
25 sig Document {}
26 //-----
27 //READING
28 sig Reading {
29     //I added the "one"
30     doc: one Document,
31     tokenseq: seq Token,
32     tt: Token->Type,
33     //tt: Token->one Type //cf. Stackoverflow
34     typeseq: seq Type
35 }{
36     #tokenseq > 0
37     not (hasDups [tokenseq])
38     dom [tt] = elems [tokenseq]
39     function [tt, elems [tokenseq]]

```

³¹<http://alloytools.org/> (Accessed November 3, 2017).

A.2. Collation Model

```
40     typeseq = tokenseq.tt
41 }
42 //-----
43 //TRANSCRIPTION
44 //similarity of 2 documents
45 pred t_similar (e, t: Document) {
46     some r1, r2: Reading |
47     {r1.doc=e
48     r2.doc=t
49     r1.typeseq = r2.typeseq}
50 }
51 // similarity of 2 readings (added to the initial readings model)
52 pred r_similar (r1, r2: Reading) {
53     r1.typeseq = r2.typeseq
54 }
55 //-----
56 //COLLATION
57 /*A very simplified model: we have 3 distinct documents.
58 * A base-text, an exemplar being collated against the base-text, and the collation.
59 * If the base-text and the exemplar share the same reading, then we do not collate.
60 * If the base-text and the exemplar have a different reading, then the exemplar's
61 * reading is transcribed
62 *
63 * So when do we have a situation of a reading being collated?
64 * (1) if a reading of the base-text and exemplar are different
65 * (2) then the readings of the exemplar and collation are the same
66 * (3) else nothing is transcribed in the collation document.
67 */
68 pred collation (disj b, e, c: Document) {
69     some r1, r2, r3: Reading |
70     {
71     //reading 1 is in the base text
72     r1.doc=b
73
74     //reading 2 is in the exemplar
75     r2.doc=e
76
77     //reading 3 is in transcribed in the collation iff r1 (base) and r2 (exemplar)
78     //are different
79     not r_similar[r1, r2] implies {r3.doc=c and r_similar[r2, r3] and t_similar[c, e]}
80     else no r: Reading | r.doc=c and t_similar[b, e]
81     }
82 }
83 //-----
84 //COMMAND
85 run collation
```

Practice

B

THIS Appendix contains the electronic files discussed during the Practice part of this dissertation. They are available both attached to the dissertation, and online in a Github repository: <https://github.com/enury/phd-automated-collation>.

The files are divided into the different stages of collation with CollateX: (1) XML transcriptions of the witnesses of Calpurnius Flaccus; (2) XSLT file that transforms XML into JSON for input to CollateX; (3) corrected Collation results from CollateX; (4) Python interface for the correction and study of collation results; (5) HTML tables that were used for analysing the manuscript tradition of the *Declamations*. This Appendix provides, for each file, a reference to the chapters where they are discussed, a summary of the content of the files, and at least one sample figure.

B.1 XML Transcriptions

Reference The TEI XML transcriptions of Calpurnius Flaccus witnesses are described in Chapter 6. The witnesses include five manuscripts, ABCMN, the *editio princeps* by Pithoeus (1594), P1594, and the critical edition of Håkanson (1978), LH. The seven files include transcriptions for the five manuscripts and two editions, as well as a master file that gathers all witnesses: 1. A.xml; 2. B.xml; 3. C.xml; 4. M.xml; 5. N.xml; 6. LH.xml; 7. all_witnesses.xml. The XML files are available at <https://github.com/enury/phd-automated-collation/tree/master/XML>.

Summary The description of the transcription files in Chapter 6 is divided into structure and content. The structure is the division of the text in pages, declamations, lines and words. The content refers to the manuscripts encoding of abbreviations, orthographic regularisations, scribal corrections, unclear passages or lacunae, editorial comments, and so on. In addition, the encoding of the critical apparatus of Pithoeus and the encoding of the critical edition of Håkanson are outlined.

B.2. XSLT Transformation

```
<ab n="0" type="incipit">
<pb n="383" facs="http://www.e-rara.ch/gep_g/content/pageview/1098914"/>

<lb n="1" break="yes"/><chi rend="uppercase"><w>Incipiunt</w> <w>ex</w> <w>cal</w></pc>
<lb n="2" break="no"/><w>Flacco</w></hi> <w>excer</w></pc>
<lb n="3" break="no"/><w>ptae</w> <w>x</w></pc> <w>Rhetorum</w> <w>minorum</w></pc>
<lb n="4" break="no"/></ab><ab n="1" type="decl"><num>I</num> <w>tyrannicida</w> <w>Uxor</w> <w>tyrannicida</w></pc>
<lb n="5" break="yes"/><chi rend="uppercase"><w>quinque</w> <w>cum</w> <w>tyranno</w> <w>proximae</w> <w>familiae</w>
```

(a) XML transcription of the *incipit* in Pithoeus (1954).

```
<!-- master file that include all witnesses to be transformed into json -->

<xi:include href="B.xml"/>
<xi:include href="C.xml"/>
<xi:include href="M.xml"/>
<xi:include href="N.xml"/>
<xi:include href="P1594.xml"/><!-- Pithoeus -->
<xi:include href="LH.xml"/><!-- Hakanson -->

<!-- <xi:include href="A.xml"/> -->
```

(b) Content of the master file that gathers all transcriptions.

Sample The Appendix B.1 shows two examples from the XML transcription files.

B.2 XSLT Transformation

Reference The XSLT transformation is described in detail in Section 7.1.3, *From XML to JSON*. The transformation scenario was performed in oXygen 1.7, with XSLT and XPath 2.0, and the transformation engine Saxon-PE 9.6.0.7. The file `witnesses-to-json.xml` is available at: <https://github.com/enury/phd-automated-collation/tree/master/XSLT>.

Summary The XSLT transformation takes the XML transcription files and transforms them into the JSON input format for CollateX. The XSLT transforms each witness word-by-word into JSON tokens, which include the original form of the word, the location of the word in the witness and the number of the Declamation in which the word appear. In addition, the tokens may also include an editorial note, and a link to a digital facsimile. The transformation goes through two stages, first the TEI XML is transformed into another XML format, which is again transformed into JSON.

Sample The following figures show parts of the two stages in the XSLT code. The first figure B.2(a) shows how a witness is first transformed into a `<witness>` element. The second figure B.2(b) shows the template that transforms again a `<witness>` element into JSON format.

B.3. JSON Collation Results

```
<!-- create a witness -->
<witness id="{ $siglum }">

  <!-- pass through each declamation -->
  <xsl:for-each select="ancestor::tei:TEI/descendant::tei:ab">

    <!--<declamation n="{ @n }"-->

    <xsl:choose>
      <!-- for modern editors, apply templates with mode 'edd' -->
      <xsl:when test="{ $siglum = 'LH' }">
        <xsl:apply-templates mode="edd"/>
      </xsl:when>
      <!-- for manuscript witnesses and Pithoeus, apply templates with mode 'wit' -->
      <xsl:otherwise>
        <xsl:apply-templates mode="wit">
          <xsl:with-param name="phand" select="{ $vhand }" tunnel="yes"/>
        </xsl:apply-templates>
      </xsl:otherwise>
    </xsl:choose>

    <!--</declamation>-->

  </xsl:for-each>
</witness>
```

(a) XSLT witness template - first stage.

```
<xsl:template match="witness">
  <!-- Open JSON object for witness -->
  <xsl:text/>{&#xA;"id" : "<xsl:value-of select="@id"/>",&#xA;"tokens" : [&#xA;<xsl:text/>
  <!-- transform tokens into JSON -->
  <xsl:apply-templates/>
  <!-- close JSON witness -->
  <xsl:text>]&#xA;}</xsl:text>
  <!-- if it is not the last witness, add comma (following-sibling::witness)-->
  <xsl:if test="position() != last()"><xsl:text>,</xsl:text></xsl:if>
  <xsl:text>&#xA;</xsl:text>
</xsl:template>
```

(b) XSLT witness template - second stage.

Figure B.2: Example from the XSLT transformation files.

B.3 JSON Collation Results

Reference The JSON output of CollateX is described in Chapter 8, the visualisation of collation results, in Section 8.4.1 which describes the structure of CollateX output. The file 2018-03-02-BCMNPH.json contains the corrected version of Calpurnius' collation, available at: <https://github.com/enury/phd-automated-collation/tree/master/pycoviz/json-collations>.

Summary The JSON output of CollateX contains a list of witnesses, and a collation table. The manuscript witnesses are divided into hands (B1 and B2, for instance), and manuscript A was not included in this collation file. The collation table is a complex structure of nested JSON objects and array. Each column of the table represent the text of a witness, while each row represent the aligned readings

at one point in the text. Here is the structure of the collation table :

- The table is a list of rows: [row 1, row 2, ... row n].
- A row is a list of cells: [cell 1, cell 2, ... cell n].
- A cell is a list of tokens: [token 1, token 2, ... token n].
- A token is a dictionary with several entries, such as “t” or “n”, for instance: "t" : "foemina", "n" : "femina", "decl" : "1", "locus" : "244r:11" (Manuscript N).

Sample Figure B.3 shows the first cell of the first row in the table, which is part of the *incipit* of Calpurnius Flaccus. This sample was formatted for a better visualisation. However, the JSON output of CollateX is not formatted, and is quite difficult to study in JSON format. Instead, it should be visualised in some way, such as a collation table in HTML.

B.4 HTML Tables

Reference The HTML collation table visualisation is discussed in Section 8.2.2. The appendix also includes tables which were used as argument in Section 8.5 when analysing the tradition of Calpurnius Flaccus. The files include:

Template `template.html` is the file model for saving new tables created within PyCoviz (see p. 278).

Example `example.html` contains a sample table with all the elements available in the HTML tables of Calpurnius (see p. 251).

Collation and Variants the file `complete-collation-html` is the complete collation table of Calpurnius Flaccus, while `all-variants.html` is the list of all variants in the collation table.

Pithoeus Two files contain the tables that concern the relationship of Pithoeus edition with the other manuscripts, `P1594-vs-manuscripts.html` and `P1594-N-unique-errors.html` (see Section 8.5.2.1).

B2 Three files contain the tables about the origins of corrections by the second hand B2: `B2-vs-B1.html` is the table of all corrections in B2; the corrections

are divided between those common to B2, M, N and Pithoeus (B2MNP-vs-B1.html), and the other corrections are in B2-corrections.html (see Section 8.5.2.2).

The tables are available at <https://github.com/enury/phd-automated-collation/tree/master/pycoviz/alignment-tables>. In order to be viewed properly in the browser, the HTML tables should be saved in the same directory as the three folders `css`, `fonts`, and `js`.

Summary The HTML table is a visualisation of the JSON output from CollateX. The originality of this particular table visualisation is the addition of paratextual elements, such as notes or folio and line numbers, which can be shown or hidden by clicking on symbols. To be properly displayed, the HTML tables need to be stored in a the same directory as the three other folders which contain the fonts, the CSS files for styling, and the javascript instructions that enable the display of paratextual elements.

Sample Figure B.4 shows how the HTML collation table appears in the browser, while figure B.5 shows a sample of javascript code which makes this visualisation possible.

B.5 PyCoviz Jupyter Notebook

Reference The PyCoviz interface is extensively described in Section 8.4 *PyCoviz: A Python Interactive Interface*. The actual file is a Jupyter Notebook available at: <https://github.com/enury/phd-automated-collation/blob/master/pycoviz/interactive-collation.ipynb>.

Summary PyCoviz content is divided into three different sections. The first section imports the necessary Python modules and the JSON collation file to be visualised. The second section contains various functions which are essential to the interface, such as functions that compare tokens, or that transform the JSON into HTML. The third section contains the widgets that let users interact directly with the collation results, by correcting the alignment, selecting agreements between witnesses, or searching for and clarifying a reading.

Sample The following figure B.6 shows the Agreement widget available in PyCoviz Section 8.4.2.3). This widget lets users filter the collation results to find readings shared by a group of selected witness, and not by other witnesses. There are five

B.5. PyCoviz Jupyter Notebook

elements in the Agreements widget: (1) and (2) dropdown menus to select witnesses, (3) the collation table, (4) a textbox and (5) a save button. Figure B.7 shows the code which is described in Section 8.4.2.3, and that is used to select variant readings for the Agreement widget.

B.5. PyCoviz Jupyter Notebook

```
[
  [
    {
      "locus": "147r:21",
      "link": "http://daten.digitale-sammlungen.de/bsb00090859/image_294",
      "decl": "0",
      "t": "incipiunt"
    },
    {
      "locus": "147r:21",
      "link": "http://daten.digitale-sammlungen.de/bsb00090859/image_294",
      "decl": "0",
      "t": "ex"
    },
    {
      "locus": "147r:21",
      "link": "http://daten.digitale-sammlungen.de/bsb00090859/image_294",
      "decl": "0",
      "t": "calpurnio"
    },
    {
      "locus": "147r:21",
      "link": "http://daten.digitale-sammlungen.de/bsb00090859/image_294",
      "decl": "0",
      "t": "flacco"
    },
    {
      "n": "excerptae",
      "locus": "147r:21",
      "link": "http://daten.digitale-sammlungen.de/bsb00090859/image_294",
      "decl": "0",
      "t": "excerptę"
    }
  ],
]
```

Figure B.3: Example of CollateX JSON output.

B1	B2	C1	C2	LH	M1	M2	N1	N2	P1594	ID
excerpta	excerpta	excerpta	excerpta							1
					contra matrem	contra matrem	contra matrem	contra matrem	contra matrem	16
					contra matronam	contra matronam	contra matronam	contra matronam	contra matronam	24
					pro milite	pro milite	pro milite	pro milite	pro milite	65
					o	o	o	o		67
					2r:7	2r:7	244v:28	244v:28		
Note: Unknown abbreviation. Normalized form supplied by Lehnert.	148r:8	82r:18	82r:18							
148r:8										
virginus	virginus	virginus	virginus	Virginus	virginus	virginus	virginus	virginus	Virginus	76
					pro parriida	pro parriida	pro parriida	pro parriida	pro parriida	84

Figure B.4: HTML collation table visualised in the browser.

```
// Expandable notes
$( '.expander' ).bind( "click", function() {
  $( this ).parent().next( '.expandable' ).slideToggle( 400 ).removeClass( "hidden" );
  // $( this ).parent().parent().find( '.expandable' ).slideToggle( 400 ).removeClass( "hidden" );
  // $( "i", this ).toggleClass( "fa-caret-down fa-caret-right" );
  return false;
});

// Expandable row info (folio:line)
$( '.expander-row' ).bind( "click", function() {
  $( this ).parent().parent().parent().find( '.expandable-row' ).slideToggle( 400 ).removeClass( "hidden" );
  // $( "i", this ).toggleClass( "fa-caret-down fa-caret-right" );
  return false;
});
```

Figure B.5: Sample of javascript code for the HTML collation table.

B.5. PyCoviz Jupyter Notebook

Agreements: B1, B2, C1, C2, LH, M1, M2

Against: B1, B2, C1, C2, LH, M1, M2

B1	B2	C1	C2	LH	M1	M2	N1	N2	P1594	ID
sacerdotis	sacerdotis	sacerdoti	sacerdoti	sacerdotis	sacerdotii	sacerdotii	sacerdotii	sacerdotii	sacerdotii	837
liberas	liberas	libera	libera	liberas	libera	libera	libera	libera	libera	1053
liberas	liberas	libera	libera	liberas	libera	libera	libera	libera	libera	1248

Total: 3 rows in the table.

Table description

Save Table

Figure B.6: Agreements widget.

Find agreements 2 [↑](#)

```
In [6]: #This function is similar to the previous one:
#it returns rows of the collation table (table) where a list of x witnesses (witlist) agree together
r, but
#do not agree with the witnesses in a second list (nonwitlist).

#By default, the function will return the agreement of the x witnesses, against all the other witnesses.

def compare_witnesses(table, witlist, nonwitlist=[]):
    result_table = []

    #first list of x witnesses which agree together
    witindex = [witnesses.index(wit) for wit in witlist]
    #against all the other witnesses
    if not nonwitlist:
        nonwitindex = [witnesses.index(wit) for wit in witnesses if wit not in witlist]
    #except if a second list of y witnesses is specified
    else:
        nonwitindex = [witnesses.index(wit) for wit in nonwitlist]

    #go through the collation table, row by row
    #to find places where the x witnesses agree together against others
    for row in table:
        #get list of cell for the x witnesses
        cell_list = [row[i] for i in witindex]
        #there must be agreement of the x witnesses (normalised tokens)
        if compare_multiple_cell_norm(cell_list) is True:
            for i in nonwitindex:
                #if they agree with one of the other y witnesses
                if compare_cell_norm(row[witindex[0]], row[i]) is True:
                    #go to next row
                    break
            #but if they do not agree with any of the y witnesses
        else:
            #add row to the result
            result_table.append(row)
    return result_table
```

Figure B.7: Code that select variant readings for the Agreement widget.

Bibliography

- Aizpurua, Paul, ed. 2005. *Les plaidoyers imaginaires: extraits des déclamations*. Paris: Gallimard.
- Andrews, Tara L. 2009. "Prolegomena to a Critical Edition of the Chronicle of Matthew of Edessa, with a Discussion of Computer-Aided Methods Used to Edit the Text". PhD thesis, DPhil. University of Oxford.
- . 2012. "The Third Way: Philology and Critical Edition in the Digital Age". *Variants* 10:1–16.
 - . 2014a. "Analysis of Variation Significance in Artificial Traditions Using Stemmaweb". *Digital Scholarship in the Humanities*: 1–17. doi:10.1093/llc/fqu072.
 - . 2014b. "Digital Techniques for Critical Edition". In *Armenian Philology in the Modern Era: From Manuscript to Digital Text*, ed. by Michael E. Stone and Valentina Calzolari, 175–195. Leiden: Brill.
 - . 2015. "Tools for Digital Philology: Transcription". Visited on Apr. 29, 2015. <http://www.digitalbyzantinist.org/2015/04/transcribing-texts-with-t-pen.html>.
 - . 2017. "What We Talk About When We Talk About Collation". In *Advances in Digital Scholarly Editing*, ed. by Peter Boot et al., 231–234. Sidestone Press. doi:10.1080/0950238042000260351.
- Andrews, Tara L., and Caroline Macé. 2013. "Beyond the tree of texts: Building an empirical model of scribal variation through graph analysis of texts and stemmata". *Literary and Linguistic Computing* 28 (4): 504–521. doi:10.1093/llc/fqt032.
- Andrews, Tara L., and Joris van Zundert. 2013. "An Interactive Interface for Text Variant Graph Models". Digital Humanities Conference DH2013, University of Nebraska–Lincoln, 16–19 July 2013. Visited on May 24, 2017. <http://dh2013.unl.edu/abstracts/ab-379.html>.

- . 2014. “Apparatus vs . Graph – an Interface as Scholarly Argument”. Visited on Oct. 31, 2014. <http://interfacecritique.net/joris-van-zundert-tara-andrews-apparatus-vs-graph-an-interface-as-scholarly-argument/>.
- Arvanitopoulos Darginis, Nikolaos, and Sabine Süssstrunk. 2014. “Binarization-free Text Line Extraction For Historical Manuscripts”. Digital Humanities Conference DH2014, Lausanne, 7–12 July 2014. Visited on June 9, 2017. <http://dharchive.org/paper/DH2014/Paper-928.xml>.
- Avanzi, Girolamo. 1495. *In Val. Catullum et in Priapeias emendationes*. Venetiis: J. de Cereto de Tridino.
- Balbo, Andrea. 2012. “Applicazioni del fenomeno della parola-segnale ai Declamationum excerpta di Calpurnio Flacco”. In *Vestigia notitiae. Scritti in memoria di Michelangelo Giusta*, ed. by Edoardo Bona, Carlos Lévy, and Giuseppina Magnaldi, 187–192. Alessandria: Edizioni dell’Orso.
- Bamman, David, and Gregory Crane. 2011. “The Ancient Greek and Latin Dependency Treebanks”. *Language technology for cultural heritage: Selected papers from the LaTeCH Workshop Series*: 79–89.
- Banderier, Gilles. 2009. “Bâle et la famille Pithou : contribution à l’étude des rapports intellectuels entre Bâle et la France au XVIe siècle Bâle et la famille Pithou”. *Revue suisse d’histoire* 59:387–409.
- Barabucci, Gioele, and Franz Fischer. 2017. “The Formalization of textual criticism. Bridging the gap between automated collation and edited critical texts”. In *Advances in Digital Scholarly Editing*, ed. by Peter Boot et al., 47–53. Sidestone Press.
- Barbrook, Adrian C., et al. 1998. “The phylogeny of the *Canterbury Tales*”. *Nature* 394 (27 August): 839.
- Barney, S. A., et al., eds. 2006. *The Etymologies of Isidore of Seville*. Cambridge University Press.
- Bédier, Joseph. 1928. “La tradition manuscrite du Lai de l’Ombre. Réflexion sur l’art d’éditer les textes anciens”. *Romania* 54 (214): 161–196, 321–356.
- Beneš, Carrie E. 2003. “The Appearance and Spread of the E-Cedilla in Latin Bookhands”. *Manuscripta. A Journal for Manuscript Research* 43-44:1–44. doi:<https://doi.org/10.1484/J.MSS.2.300664>.
- Bernstein, Neil W. 2013. *Ethics, Identity, and Community in Later Roman Declamation*. Oxford University Press.
- Berti, Monica, Bridget Almas, and Gregory Crane. 2016. “The Leipzig Open Fragmentary Texts Series (LOFTS)”. *DHQ* 10 (2).

- Birnbaum, David J. 2012. "Editorial principles". Visited on Jan. 17, 2018. <http://suprasliensis.obdurodon.org/editorial-principles.html>.
- Blake, Norman, and Jacob Thaisen. 2004. "Spelling's Significance for Textual Studies". *Nordic Journal of English Studies* 3 (1): 93–108.
- Bleeker, Elli. 2017. "Mapping Invention in Writing: Digital Infrastructure and the Role of the Genetic Editor". PhD thesis, Universiteit Antwerpen.
- Bleeker, Elli, and Elena Spadini. 2016. "Code and Collation: training textual scholars". Visited on Aug. 28, 2017. <https://dixit.hypotheses.org/1127>.
- Bloomer, W. Martin. 1997. "Schooling in Persona: Imagination and Subordination in Roman Education". *Classical Antiquity* 16 (1): 57–78. doi:10.2307/25011054.
- . 2007. "Roman Declamation: The Elder Seneca and Quintilian". In *A Companion to Roman Rhetoric*, ed. by Jon Hall and William Dominik, 297–306. Blackwell Reference Online. doi:10.1002/9780470996485.ch22.
- Boretti, Margherita. 2009. "Teoria e prassi di filologia computazionale : aspetti, problemi, sperimentazioni. L'edizione critica del Viage al Purgatory di Ramon de Perellos per la produzione dell'edizione elettronica sul Web con PinakesText." PhD thesis, Università di Pisa.
- Boschetti, Federico. 2007. "Methods to extend Greek and Latin corpora with variants and conjectures : Mapping critical apparatuses onto reference text". In *Proceedings of the Corpus Linguistics Conference (CL2007)*, 1–11.
- Boschetti, Federico, et al. 2009. "Improving OCR Accuracy for Classical Critical Editions". In *Research and Advanced Technology for Digital Libraries. 13th European Conference, ECDL 2009, Corfu, Greece, September 27 - October 2, 2009. Proceedings*, ed. by Maristella Agosti et al., 156–167. Berlin, Heidelberg: Springer Berlin Heidelberg.
- Bourgain, Pascale, and Françoise Vielliard, eds. 2002. *Conseils pour l'édition des textes médiévaux. 3. Textes littéraires*. Paris: École nationale des Chartes.
- Bremer, Thomas, et al. 2015. "Zum Einsatz digitaler Methoden bei der Erstellung und Nutzung genetischer Editionen gedruckter Texte mit verschiedenen Fassungen". *editio* 29:29–51.
- Broude, R. 1991. "When Accidentals are Substantive: Applying Methodologies of Textual Criticism to Scholarly Editions of Music". *Text: Transactions of the Society for Textual Scholarship* 5 (1991): 105–20.
- Bryant, John, ed. 2018a. "Versions of Billy Budd: A Fluid-Text Edition". Visited on Feb. 10, 2018. <https://mel.hofstra.edu/versions-of-billy-budd.html>.

- , ed. 2018b. “Versions of Moby Dick: A Fluid-Text Edition”. Visited on Feb. 10, 2018. <https://mel.hofstra.edu/versions-of-moby-dick.html>.
- Bülow-Jacobsen, Adam. 1979. “Some Considerations on the Quality of Microfilms of Manuscripts”. *Cahiers de l’Institut du Moyen-Âge Grec et Latin* 30:91–104.
- Burman, Pieter, ed. 1720. *Declamationes 19 Majores Et Quae Ex 388 Supersunt 145 Minores, Et Calpurnii Flacci Declamationes Cum Notis Doctorum Virorum*. Du Vivié, Johannes.
- Burnard, Lou. 2014a. “Introduction”. In *What is the Text Encoding Initiative? How to add intelligent markup to digital resources*. Marseille: OpenEdition Press.
- . 2014b. “The TEI and XML”. Chap. 1 in *What is the Text Encoding Initiative? How to add intelligent markup to digital resources*. Marseille: OpenEdition Press.
- Busa, R. 1980. “The annals of humanities computing: The index Thomisticus”. *Computers and the Humanities* 14 (2): 83–90. doi:10.1007/BF02403798.
- Cabaniss, Margaret Scanlon. 1970. “Using the Computer for Text Collation”. *Computer Studies in the Humanities and Verbal Behaviour* 3:1–33.
- Campano, Giannantonio. 1495. *Omnia Campani opera*. Ed. by Michele Ferno. Venetiis: Bernardinum Vercellensem jussu domini Andreae Torresano de Assula.
- . 1707. *Jo. Antonii Campani Epistolae et Poemata, una cum vita Auctoris*. Ed. by Johann Burkhard Mencke.
- Cannon, Robert L. 1976. “OPCOL: An Optimal Text Collation Algorithm”. *Computers and the Humanities* 10 (1): 33–40. doi:10.1007/BF02399140.
- Cannon, Robert L., and Robert Oakman. 1989. “Interactive Collation on a Micro-Computer. The URICA! Approach”. *Computers and the Humanities* 23:469–472.
- Cardelle de Hartmann, Carmen, Darko Senekovic, and Thomas Ziegler. 2014. “Modes of variability: the textual transmission of Petrus Alfonsi’s Dialogus”. In *Petrus Alfonsi and His Dialogus. Background – Context - Reception*, ed. by Carmen Cardelle de Hartmann and Philip Roelli, 227–248. Firenze: SISMEL - Ed. del Galluzzo.
- Caton, Paul. 2009. “Lost in Transcription: Types, Tokens, and Modality in Document Representation”. In *Digital Humanities 2009. Conference Abstracts. University of Maryland, College Park June 22 – 25, 2009*, 80–82. Maryland Institute for Technology in the Humanities (MITH).
- . 2013. “On the term ‘text’ in digital humanities”. *Literary and Linguistic Computing* 28, no. 2 (): 209–220. doi:10.1093/lc/fqt001.

- Cayless, Hugh. 2015. "The TEI Critical Apparatus Module and Digital Critical Editions, or Why Your Digital Edition Should Have a Data Model". Visited on Aug. 28, 2015. <https://github.com/hcayless/appcrit/blob/master/docs/DLL-seminar-paper.md>.
- Cerquiglini, Bernard. 1989. *Éloge de la variante: Histoire critique de la philologie*. Aux travaux. Paris: Seuil.
- Chambers, Sally, et al. 2017. "Towards a tool and data criticism framework. A developer's and user's perspective". In *The abstracts of the DHBenelux 2017 - Wednesday 5 July 2017*, 72–74.
- Chaudhuri, Sukanta, ed. 2015. *Bichitra: The Making of an Online Tagore Variorum*. Springer.
- Chiesa, Paolo. 2002. *Elementi di critica testuale*. Testi e manuali per l'insegnamento universitario del latino. Bologna: Pàtron Editore.
- Cloppet, Florence, et al. 2016. "ICFHR2016 competition on the classification of medieval handwritings in Latin script". *Proceedings of International Conference on Frontiers in Handwriting Recognition, ICFHR*: 590–595. doi:10.1109/ICFHR.2016.0113.
- Colwell, Ernest Cadman, and Ernest W. Tune. 1964. "Variant readings : classification and use". *Journal of Biblical Literature* 83 (3): 253–261.
- Cortesi, M. 1994. "Un nuovo testimone delle 'Declamationes minores' pseudoquintilianee". In *Immagini del Medioevo. Saggi di cultura mediolatina*, 81–95. Spoleto.
- Courtney, Edward. 1967. "The Transmission of Juvenal's Text". *Bulletin of the Institute of Classical Studies* 14:38–50.
- Cozzo, Andrea. 2006. *La tribù degli antichisti. Un'etnografia ad opera di un suo membro*. Roma: Carocci.
- Craig-McFeely, Julia. 2008. "Digital Image Archive of Medieval Music: The evolution of a digital resource". *Digital Medievalist* 3.
- Crane, Gregory. 2014. "The Digital Loeb Classical Library - a view from Europe". Visited on Oct. 31, 2016. <http://sites.tufts.edu/perseusupdates/2014/09/22/the-digital-loeb-classical-library-a-view-from-europe/>.
- Crane, Gregory, et al. 2014. "Cataloging for a Billion Word Library of Greek and Latin". *Proceedings of the First International Conference on Digital Access to Textual Cultural Heritage - DATeCH '14*: 83–88. doi:10.1145/2595188.2595190.
- Cummings, James. 2013. "About the Project". Visited on Aug. 2, 2017. <http://www.cems.ox.ac.uk/holinshed/about.shtml>.

- Cummings, James, and Arno Mittelbach. 2011. "The Holinshed Project: Comparing and linking two editions of Holinshed's Chronicle". *International Journal of Humanities and Arts Computing* 4 (1-2): 39–53.
doi:<https://doi.org/10.3366/ijhac.2011.0006>.
- D'Avray, David L. 2012. "Contamination, Stemmatics and the Editing of Medieval Latin Texts". In *Ars Edendi Lecture Series*, ed. by Alessandra Bucossi and Erika Kihlman, 2:63–82. Stockholm: Stockholm University.
- Dahlström, Mats. 2000. "Drowning by Versions". *Human IT* 4:7–38.
- Dain, Alphonse. 1964. *Les Manuscrits*. Paris: Les Belles Lettres.
- Damerau, Fred J. 1964. "A technique for computer detection and correction of spelling errors". *Communications of the ACM* 7 (3): 171–176.
doi:[10.1145/363958.363994](https://doi.org/10.1145/363958.363994).
- Damon, Cynthia. 2016. "Beyond Variants: Some Digital Desiderata for the Critical Apparatus of Ancient Greek and Latin Texts". In *Digital Scholarly Editing. Theories and Practices*, ed. by Matthew J. Driscoll and Elena Pierazzo, 201–218. Open Book Publisher.
- Dane, Joseph A. 2003. "The Notion of Variant and the Zen of Collation". Chap. 4 in *The Myth of Print Culture: Essays on Evidence, Textuality and Bibliographical Method*, 88–113. University of Toronto Press.
- Dawe, R. D. 1964. *The Collation and Investigation of Manuscripts of Aeschylus*. Cambridge: Cambridge University Press.
- De Strycker, Émile. 1975. "Suggestions pratiques pour la collation des manuscrits d'un texte hagiographique grec à tradition riche". In *Corona gratiarum : Miscellanea patristica, historica et liturgica Eligio Dekkers O.S.B. 12 lustra complenti ablata*. Ed. by Eligius Dekkers, 2:345–366. Brugge : Sint Pietersabdej.
- De Young, Gregg. 2009. "Diagrams in ancient Egyptian geometry. Survey and assessment". *Historia Mathematica* 36 (4): 321–373.
doi:[10.1016/j.hm.2009.02.004](https://doi.org/10.1016/j.hm.2009.02.004).
- De Biasi, Pierre-Marc. 2000. *La génétique des textes*. Paris: Natan.
- . 2004. "Toward a science of literature: manuscript analysis and the genesis of the work". In *Genetic Criticism: Texts and Avant-textes*. Ed. by J. Jed Deppman, D. Ferrer, and M. Groden, 36–68. Philadelphia: University of Pennsylvania Press.
- Dearing, Vinton A. 1962. *Methods of Textual Editing. A paper delivered by Vinton A. Dearing at a Seminar on Bibliography held at the Clark Library, 12 May 1962*. Los Angeles: William Andrews Clark Memorial Library.

- . 1970. “Computer Aids to Editing the Text of Dryden”. In *Art and Error: Modern Textual Editing. Essays compiled and edited by Ronald Gottesman and Scott Bennett*. 254–278. London: Methuen / Co. Ltd.
- Dekker, Ronald Haentjens. 2014. “Computer automated collation with CollateX and Python”. Presentation given at DH Benelux in The Hague, 12-13 June 2014. Visited on Oct. 8, 2015. <http://dhbenelux.org/wp-content/uploads/2014/06/demo-haentjens-dekker.pdf>.
- Dekker, Ronald Haentjens, and Gregor Middell. 2011. “Computer-Supported Collation with CollateX: Managing Textual Variance in an Environment with Varying Requirements”. In *Supporting Digital Humanities, Copenhagen 17-18 November 2011: Conference Proceedings*. Copenhagen: Croatian National Corpus.
- Dekker, Ronald Haentjens, et al. 2015. “Computer-supported collation of modern manuscripts: CollateX and the Beckett Digital Manuscript Project”. *Literary and Linguistic Computing* 30 (3): 452–470.
- DeRose, Stephen J., et al. 1990. “What is Text, Really”. *Journal of Computing in Higher Education* 1 (2): 3–26.
- Dillen, Wout. 2015. ““(Hiatus in Ms.)”. Towards a TEI compliant typology of textual lacunae in Samuel Beckett’s manuscripts”. *Manuscritica. Revista de crítica genética* 28:65–73.
- Dillen, Wout, and Dirk Van Hulle. 2013–. “Lexicon of Scholarly Editing”. Visited on Oct. 31, 2016. uahost.uantwerpen.be/lse/index.php/lexicon/.
- Dinter, Martin T., Charles Guérin, and Marcos Martinho, eds. 2017. *Reading Roman Declamation - Calpurnius Flaccus*. De Gruyter.
- Driscoll, Matthew J. 2000. “Encoding Old Norse / Icelandic Primary Sources using TEI-Conformant SGML”. *Literary and Linguistic Computing* 15 (1): 81–91.
- . 2006. “Levels of transcription”. In *Electronic Textual Editing*, ed. by Lou Burnard, Katherine O’Brien O’Keeffe, and John Unsworth, 254–261. New York: Modern Language Association of America.
- . 2009. “Marking up abbreviations in Old Norse-Icelandic manuscripts”. In *Medieval texts – contemporary media: The art and science of editing in the digital age*, ed. by Maria Grazia Saibene and Marina Buzzoni, 13–34. Pavia: Ibis.

- Dubuisson, Marc, and Caroline Macé. 2006. "Handling a Large Manuscript Tradition with a Computer". In *The Evolution of Texts: Confronting Stemmatalogical and Genetical Methods. Proceedings of the International Workshop held in Louvain-la-Neuve on September 1–2, 2004*, ed. by Caroline Macé et al., 25–37. Pisa, Roma: Istituti Editoriali e Poligrafici Internazionali.
- Duplacy, Jean, and Éric Huret. 1977. "Classification des états d'un texte, mathématiques et informatique: repères historiques et recherches méthodologiques". *Revue d'histoire des textes* 5 (1975): 249–309.
- Eley, Penny, et al. 2005. "Partonopeus de Blois: transcriptions of all manuscripts and fragments". Visited on Feb. 6, 2018. <http://purl.ox.ac.uk/ota/2499>.
- Englert, W. 2003. *Lucretius: On the Nature of Things*. Classical Library. Newbury: Focus Publishing.
- Epp, Eldon Jay. 1993. "Towards the Clarification of the Term 'Textual Variant'". In *Studies in the Theory and Method of New Testament Textual Criticism*, ed. by Gordon D. Fee and Eldon Jay Epp, 47–61. Grand Rapids, Michigan: Eerdmans Publishing.
- . 2007. "It's All about Variants: A Variant-Conscious Approach to New Testament Textual Criticism". *Harvard Theological Review* 100 (3): 275–308. doi:10.1017/S0017816007001599.
- Fischer, Andreas, et al. 2009. "Automatic Transcription of Handwritten Medieval Documents". In *Proceedings of the 15th International Conference on Virtual Systems and Multimedia. 9-12 September 2009. Vienna, Austria*. Ed. by Robert Sablatnig, Martin Kampel, and Martin Lettner, 137–142. Los Alamitos, CA: IEEE Computer Society. doi:10.1109/VSMM.2009.26.
- Fischer, Bonifatius. 1970. "The Use of Computers in New Testament Studies, with Special Reference to Textual Criticism". *Journal of Theological Studies* 21 (2): 297–308.
- Fischer, Franz. 2012. "All Texts are Equal, but... Textual Plurality and the Critical Text in Digital Scholarly Editions". *Variants* 10:77–92.
- Flanders, Julia, and Fotis Jannidis. 2016. "Data Modeling". In *A New Companion to Digital Humanities*, ed. by Susan Schreibman, Ray Siemens, and John Unsworth, 229–237. John Wiley & Sons, Ltd. doi:10.1111/b.9781118680643.2016.00018.x.
- Flores, Enrico. 1998. *Elementa critici di critica del testo ed epistemologia*. Napoli: Loffredo.

- Flüeler, Christoph, and Dot Porter. 2015. "Digital Manuscripts as Critical Edition". Visited on Aug. 5, 2015. <http://schoenberginstitute.org/2015/06/30/digital-manuscripts-as-critical-edition/>.
- Forrin, Noah D, Tanya R Jonker, and Colin M MacLeod. 2014. "Production improves memory equivalently following elaborative vs non-elaborative processing." *Memory* 22 (5): 470–80. doi:10.1080/09658211.2013.798417.
- Franzini, Greta, Melissa Terras, and Simon Mahony. 2016. "A Catalogue of Digital Editions". In *Digital Scholarly Editing. Theories and Practices*, ed. by Matthew J. Driscoll and Elena Pierazzo, 161–182. Open Book Publisher.
- Froger, Jacques. 1966. "La collation des manuscrits à la machine électronique". *Bulletin d'information de l'Institut de Recherche et d'Histoire des Textes* 13:135–171. doi:10.3406/rht.1966.1035.
- . 1968. *La critique des textes et son automatisations*. Paris: Dunod.
- . 1970. "La critique des textes et l'ordinateur". *Vigiliae Christianae* 24 (3): 210–217.
- Gabler, Hans Walter. 2008. "Remarks on Collation". Visited on Aug. 25, 2015. https://www.academia.edu/167070/%7B%5C_%7DRemarks%7B%5C_%7DOn%7B%5C_%7DCollation%7B%5C_%7D.
- Gadda, C. E. 1983. *Racconto italiano di ignoto del novecento*. Ed. by Dante Isella. Torino: G. Einaudi.
- Ganascia, Jean-Gabriel. 2014. "MEDITE". Visited on July 17, 2017. <http://obvil.paris-sorbonne.fr/developpements/medite>.
- Gibbs, Fred, and Trevor Owens. 2012. "Building Better Digital Humanities Tools: Toward broader audiences and user-centered designs". *Digital Humanities Quarterly* 6 (2).
- Gibson, William M., and George R. Petty. 1970. "Project OCCULT: The Ordered Computer Collation of Unprepared Literary Text". In *Art and Error: Modern Textual Editing. Essays compiled and edited by Ronald Gottesman and Scott Bennett*. 279–300. London: Methuen / Co. Ltd.
- Gilbert, Penny. 1973. "Automatic Collation: A Technique for Medieval Texts". *Computers and the Humanities* 7 (3): 139–147.
- . 1974. "Using the Computer to Collate Medieval Latin Manuscripts". In *The Computer in Literary and Linguistic Studies*, ed. by Alan Jones and Robert F. Churchouse, 106–113. Cardiff: University of Wales Press.

- . 1979. “The Preparation of Prose-Text Editions with the COLLATE System”. In *La pratique des ordinateurs dans la critique des textes. Paris, 29-31 Mars 1978*, ed. by Jean Irigoien and G. P. Zarri, 245–254. Paris: Éditions du Centre National de la Recherche Scientifique.
- Greetham, David. 1994. *Textual Scholarship: An Introduction*. New York: Garland.
- . 2007. “What is Textual Scholarship?” In *A Companion to the History of the Book*, ed. by Simon Eliot and Jonathan Rose, 21–32. Blackwell Reference Online. doi:10.1111/b.9781405127653.2007.00003.x.
- . 2013. “A history of textual scholarship”. In *The Cambridge Companion to Textual Scholarship*, ed. by Neil Fraistat and Julia Flanders, 16–41. Cambridge University Press.
- Greg, Walter W. 1950. “The Rationale of Copy-text”. *Studies in Bibliography* 3:19–36.
- Gronovius, Johann Friedrich, ed. 1665. *M. Fabii Quintiliani Institutionum Oratoriarum Libri Duodecim : Accesserunt huic renovatae editioni Declamationes ... Cum Turnebi, Camerarii, Parei, Gronovii & Aliorum Notis ...* Hackius.
- Guffey, George Robert. 1968. “Standardization of Photographic Reproductions for Mechanical Collation”. *The Papers of the Bibliographical Society of America* 62 (2): 237–240.
- Gunderson, Erik. 2003. *Declamation, Paternity, And Roman Identity. Authority And The Rhetorical Self*. Cambridge: Cambridge University Press.
- Hagel, Stefan. 2007. “The Classical Text Editor. An attempt to provide for both printed and digital editions”. In *Digital Philology and Medieval Texts. Proceedings of the Arezzo Seminar 2006, 19-21 January*, ed. by Arianna Ciula and Francesco Stella, 77–84. Pisa: Olschki.
- Håkanson, Lennart. 1972. “Some critical remarks on Calpurnius Flaccus”. *Eranos* 70:59–71.
- . 1974. “Some more critical remarks on Calpurnius Flaccus”. *Eranos* 72:53–64.
- . 1976. “On two passages in Calpurnius Flaccus”. *Eranos* 74:67–68.
- . 1978. *Calpurnii Flacci Declamationum Excerpta*. Stutgardiae: Teubner.
- . 2014. “Der Satzrhythmus des 19 Grösseren Deklamationen und des Calpurnius Flaccus”. In *Unveröffentlichte Schrifte. Band 1: Studien zu den pseudoquintilianischen Declamationes maiores*. Ed. by Biagio Santorelli, 47–130. Berlin/Boston: De Gruyter.
- Halm, Karl F., et al. 1868. *Catalogus codicum latinorum Bibliothecae Regiae Monacensis*. Vol. I.1. Monachii : Bibliotheca regia. doi:urn:nbn:de:bvb:12-bsb00008254-3.

- Harkins, P. W. 1958. "Bisensory collation of MSS. A modern method for collating MSS." *Manuscripta* 2:162–166. doi:<https://doi.org/10.1484/J.MSS.3.81>.
- Haug, Dag. 2015. "Treebanks in historical linguistic research". In *Perspectives on Historical Syntax*, ed. by Carlotta Viti, 188–202. Amsterdam: Benjamins.
- Havet, Louis. 1911. *Manuel de critique verbale appliquée aux textes latins*. Paris: Librairie Hachette.
- Heikkilä, Tuomas. 2014. "The Possibilities and challenges of computer-assisted stemmatology: the example of Vita et miracula s. Symeonis Treverensis". In *The Analysis of Ancient and Medieval Texts and Manuscripts: Digital Methods*, ed. by Tara Andrews and Caroline Macé, 19–42. Turnhout: Brepols.
- Hilton, Michael L. 1992. "The URICA! II Interactive Collation System". *Computers and the Humanities* 26:139–144.
- Hockey, Susan. 1980. *A Guide to Computer Applications in the Humanities*. London: Duckworth.
- . 2000. *Electronic Texts in the Humanities : Principles and Practice*. Oxford: Oxford University Press.
- Hogeweg, P., and B. Hesper. 1984. "The alignment of sets of sequences and the construction of phyletic trees: An integrated method". *Journal of Molecular Evolution* 20 (2): 175–186. doi:[10.1007/BF02257378](https://doi.org/10.1007/BF02257378).
- Honkapohja, Alpo. 2013. "Manuscript abbreviations in Latin and English: History, typologies and how to tackle them in encoding". *Studies in Variation, Contacts and Change in English* 14.
- Horton, Thomas B. 1994. "Sequence Comparison and Old-Spelling Texts". In *Research in Humanities Computing 2*, ed. by Susan Hockey and Nancy Ide, 89–110. Oxford.
- Houghton, Hugh. 2013. "The Electronic Scriptorium: Markup for New Testament Manuscripts". In *Digital Humanities in Biblical, Jewish and Early Christian Studies*, ed. by Claire Clivaz, A Gregory, and D Hamidovic, 31–60. Leiden: Brill.
- Houghton, Hugh, Martin Sievers, and Catherine Smith. 2014. "The Workspace for Collaborative Editing". Digital Humanities Conference DH2014, Lausanne, 7–12 July 2014. Visited on Nov. 20, 2017. <http://dharchive.org/paper/DH2014/Paper-224.xml>.
- Houghton, Hugh, and Catherine Smith. 2016. "Digital Editing and the Greek New Testament". In *Ancient Worlds in Digital Culture*, ed. by Claire Clivaz, Paul Dilley, and David Hamidovic, 110–127. Brill.

- Huculak, J. Matthew, and Ashlin Richardson. 2013. "White Paper : A Survey of Current Collation Tools for The Modernist Versions". Visited on Aug. 25, 2015. <http://web.uvic.ca/~%7B%5C~%7B%7D%7Dmvp1922/wp-content/uploads/2013/10/WhitepaperFINAL.pdf>.
- Huelsenbeck, Bart. 2016. "Annotations to a Corpus of Latin Declamations: History, Function, and the Technique of Rhetorical Summary". *Lexis. Poetica, retorica e comunicazione nella tradizione classica* 34:357–382.
- Huitfeldt, Claus, Yves Marcoux, and C. M. Sperberg-McQueen. 2008. "What is transcription?" *Literary and Linguistic Computing* 23 (3): 295–310. doi:10.1093/lc/fqn013.
- The Interedition Development Group. 2013. "CollateX – Documentation". Visited on July 29, 2014. <http://collatex.net/doc/>.
- Jackson, Daniel. 2006. *Software Abstractions: Logic, Language, And Analysis*. Cambridge, Mass. ; London: MIT Press.
- Jakacki, Diane, and Katherine Faull. 2015. "Building Bridges to – Where? The Phenomenology of Undergraduate DH". Innovations in Digital Humanities Pedagogy: Local, National, International Training. A Mini-conference, and Member Meeting Sponsored by the International Digital Humanities Training Network. Visited on July 13, 2017. <http://dh2015.org/innovations-in-digital-humanities-pedagogy/>.
- Jänicke, Stefan, Marco Büchler, and Gerek Scheuermann. 2014. "Improving the Layout for Text Variant Graphs". In *VisLR: Visualization as Added Value in the Development, Use and Evaluation of Language Resources*, 41–48.
- Jänicke, Stefan, Annette Geßner, and B Marco. 2014. "5 Design Rules for Visualizing Text Variant Graphs". Digital Humanities Conference DH2014, Lausanne, 7–12 July 2014. Visited on Sept. 2, 2017. <http://dharchive.org/paper/DH2014/Paper-652.xml>.
- Jänicke, Stefan, and David Joseph Wrisley. 2017. "Visualizing Mouvance: Toward a visual analysis of variant medieval text traditions". *Digital Scholarship in the Humanities* 32 (September): ii106–ii123.
- Jänicke, Stefan, et al. 2015. "TRAViz: A Visualization for Variant Graphs". *Digital Scholarship in the Humanities* 30 (Issue suppl_1): i83–i99. doi:<https://doi.org/10.1093/lc/fqv049>.
- Jones, F. 1985. "Two notes on Calpurnius Flaccus". *Acta Classica* 28:89–90.
- Karsdorp, Folger. 2017. "Python Programming for the Humanities by Folger Karsdorp". Visited on Feb. 1, 2018. <http://www.karsdorp.io/python-course/>.

- Kaster, Robert. 2001. "Controlling Reason: Declamation in Rhetorical Education at Rome". In *Education in Greek and Roman Antiquity*, ed. by Yun Lee Too, 317–337. Brill.
- Kaster, Robert A. 1996. "Review of Lewis A. Sussman, *The Declamations of Calpurnius Flaccus*". *Classical Philology* 91 (1): 84–89.
- Kenny, Vincent, Matthew Nathal, and Spencer Saldana. 2014. "Heuristic algorithms". Visited on Oct. 20, 2018. https://optimization.mccormick.northwestern.edu/index.php/Heuristic_algorithms.
- Kestemont, Mike, and Dominique Stutzmann. 2017. "Script Identification in Medieval Latin Manuscripts Using Convolutional Neural Networks". Digital Humanities Conference DH2017, Montréal, Canada, 8-11 août 2017. Visited on Aug. 23, 2017. <https://dh2017.adho.org/abstracts/078/078.pdf>.
- Kingsley, Stephanie. 2014. "Work Flows and Wish Lists: Reflections on Juxta as an Editorial Tool". Visited on Apr. 10, 2017. <http://www.juxtaoftware.org/work-flows-and-wish-lists-reflections-on-juxta-as-an-editorial-tool/>.
- Kline, Mary-Jo. 1998. *A Guide to Documentary Editing*. 2nd ed. Baltimore: Johns Hopkins University Press.
- Kluyver, Thomas, et al. 2016. "Jupyter Notebooks — a publishing format for reproducible computational workflows". In *Positioning and Power in Academic Publishing: Players, Agents and Agendas*, ed. by F. Loizides and B. Schmidt, 87–90. IOS Press.
- Kopp, Matthias, Marc Wilhelm Küster, and Wilhelm Ott. 2000. "TUSTEP im WWW-Zeitalter : Werkzeug für Anwender und Programmierer". *Historical Social Research* 25 (1): 143–151.
- Kraft, Robert. 1995. "The Use of Computers in New Testament Studies, with Special Reference to Textual Criticism". In *The New Testament in Contemporary Research. Essays on the Status Quaestionis*, First, ed. by Bart Ehrman and Michael Holmes, 268–282. Grand Rapids: Eerdmans.
- Laiacona, Nick. 2007. "The Difference Algorithm". Visited on Feb. 28, 2014. <http://www.juxtaoftware.org/the-difference-algorithm/>.
- Lapin, Hayim. 2013. "Towards a Digital Critical Edition of the Mishnah". In *Envisioning Judaism: Studies in Honor of Peter Schäfer on the Occasion of his Seventieth Birthday*, ed. by Ra'anana S. Boustany et al., 441–464. Tübingen: Mohr Siebeck.
- Lavagnino, John. 2006. "When Not to Use TEI". In *Electronic Textual Editing*, ed. by Lou Burnard, Katherine O'Brien O'Keefe, and John Unsworth, 334–338. New York: Modern Language Association of America.

- Lee, Christopher, Catherine Grasso, and Mark F. Sharlow. 2002. "Multiple sequence alignment using partial order graphs". *Bioinformatics* 18 (3): 452–464. doi:10.1093/bioinformatics/18.3.452.
- Lehnert, Georges. 1903. *Calpurnii Flacci declamationes*. Stutgardiae: Teubner.
- Lemaire, N. E., ed. 1824. "Calpurnii Flacci Excerptae Decem Rhetorum Minorum, cum Pithoei, Gronovii, Schultingii, et aliorum notis, ex Recensione Burmanniana". Visited on Aug. 10, 2017. <http://reader.digitale-sammlungen.de/de/fs1/object/goToPage/bsb10247356.html?pageNo=531>.
- , ed. 1825. *De quorum operibus judicia testimoniaque omnia, item annales Quintilianeos, editiones recensuit in tres indices absolutissimos, emendavit, auxit N.-E. Lemaire... Volumen VII et ultimum*.
- Lendle, Otto. 1968. *Gregorius Nyssenus. Encomium in Sanctum Stephanum Protomartyrem*. Leiden: Brill.
- Levenshtein, Vladimir I. 1966. "Binary codes capable of correcting deletions, insertions, and reversals". Trans. by P. S. Novikov, 10 (8): 707–710.
- Li, Charles. 2017. "Critical Diplomatic Editing. Applying text-critical principles as algorithms". In *Advances in Digital Scholarly Editing*, ed. by Peter Boot et al., 305–310. Sidestone Press.
- Library of Latin Texts – Series A*. 2017. [Turnhout]: Brepols.
- Lindstrand, Gordon. 1971. "Mechanized Textual Collation and Recent Designs". *Studies in Bibliography* 24:204–214.
- Love, Harold. 1984. "The Ranking of Variants in the Analysis of Moderately Contaminated Manuscript Traditions". *Studies in Bibliography* 37:39–57.
- Maas, Paul. 1958. *Textual criticism*. Trans. by Barbara Flower. Oxford: Clarendon Press.
- Macé, Caroline, Ilse De Vos, and Koen Geuten. 2012. "Comparing Stemmatological and Phylogenetic Methods to Understand the transmission History of the Florilegium Coislinianum". In *Ars Edendi Lecture Series*, ed. by Alessandra Bucossi and Erika Kihlman, 2:107–129. Stockholm: Stockholm University.
- Macé, Caroline, et al. 2015. "Textual criticism and text editing". Chap. 3 in *Comparative Oriental Manuscript Studies: An Introduction*, ed. by Alessandro Bausi et al., 321–466. Hamburg: Tredition. doi:10.5281/zenodo.46784.
- Machan, Tim William. 2002. "Texts". In *A Companion to Chaucer*, ed. by Peter Brown. Blackwell Reference Online. doi:10.1002/9780470693469.ch26.

- Maretti, E., and G. P. Zarri. 1967. "Collatio Codicum: An Exercise in COMMIT Programming". *La ricerca scientifica* 37:608–611.
- Marín, Francisco Marcos. 1991. "Computers and Text Editing: A Review of Tools, an Introduction to UNITE and Some Observations Concerning its Application to Old Spanish Texts". *Romance Philology* 45 (1): 102–122.
- Marín, Francisco Marcos, and Juan de Dios Godoy García. 1989. "Requisitos para la edición crítica informatizada: UNITE. Conjunto de programas para la Unificación automática de Textos. Versión para microordenadores SUN". *AIH. Actas X*: 1227–1251.
- Marwick, Ben. 2015. "How Computers Broke Science — and What We Can Do to Fix It". Visited on Nov. 10, 2015. gizmodo.com/how-computers-broke-science-and-what-we-can-do-to-fix-i-1741649207.
- McCarty, Willard. 2004. "Modeling: A Study in Words and Meanings". Chap. 19 in *A Companion to Digital Humanities*, ed. by Susan Schreibman, Ray Siemens, and John Unsworth. Oxford: Blackwell.
- McGurk, Patrick. 1961. "Citation marks in early Latin manuscripts. (With a list of citation marks in manuscripts earlier than A. D. 800 in English and Irish libraries)". *Scriptorium* 15 (1): 3–13.
- Monella, Paolo. 2014a. "Many witnesses , many layers : the digital scholarly edition of the Iudicium coci et pistoris (Anth . Lat . 199 Riese)". In *Digital Humanities: progetti italiani ed esperienze di convergenza multidisciplinare. Atti del convegno annuale dell'Associazione per l'Informatica Umanistica e la Cultura Digitale (AIUCD) Firenze, 13-14 dicembre 2012*, ed. by Fabio Ciotti, 173–206. Quaderni DigiLab.
- . 2014b. "Why are there no comprehensively digital scholarly editions of classical texts ?" Visited on Apr. 12, 2016. http://www1.unipa.it/paolo.monella/lincei/files/why/why_paper.pdf.
- Monroy, Carlos, et al. 2002. "Visualization of Variants in Textual Collations to Analyze the Evolution of Literary Works in the Cervantes Project". In *Research and Advanced Technology for Digital Libraries. ECDL 2002. Lecture Notes in Computer Science*, ed. by M. Agosti and C. Thanos, 2458:638–653. Berlin, Heidelberg: Springer. doi:http://dx.doi.org/10.1007/3-540-45747-X_48.
- Montanari, Franco. 2002. "Alexandrian Homeric Philology. The Form of Ekdoxis and the Variarum Lectiones". In *Epea Pteroenta. Beiträge zur Homerforschung, Festschrift für Wolfgang Kullmann zum 75. Geburtstag*, ed. by Michael Reichel and Antonios Rengakos, 119–140. Stuttgart: Franz Steiner Verlag.

- . 2015. “From Book to Edition: Philology in Ancient Greece”. In *World Philology*, ed. by Sheldon Pollock, Benjamin A. Elman, and Ku-Ming Kevin Chang, 25–44. Cambridge, Massachusetts; London, England: Harvard University Press.
- Montefusco, Lucia. 2003. “Ductus and color: the right way to compose a suitable speech”. *Rhetorica: A Journal of the History of Rhetoric* 21 (2): 113–131.
- Mordenti, R. 2001. *Informatica e critica dei testi*. Rome: Bulzoni Editore.
- Myers, Eugene W. 1986. “An O(ND) Difference Algorithm and Its Variations”. *Algorithmica* 1 (1-4): 251–266.
- Nagy, Gregory. 2010. “The Homer Multitext”. In *Online Humanities Scholarship: The Shape of Things to Come. Proceedings of the Mellon Foundation Online Humanities Conference at the University of Virginia March 26-28, 2010*, ed. by Jerome McGann et al., 87–112. Rice University Press.
- Naji, Nada, and Jacques Savoy. 2011. “Information Retrieval Strategies for Digitized Handwritten Medieval Documents”. In *Information Retrieval Technologies. Proceedings of the 7th Asia Information Retrieval Societies Conference, AIRS 2011, Dubai, United Arab Emirates, December 18-20, 2011*, ed. by M.V.M. Salem et al., 103–114. Berlin, Heidelberg: Springer.
- né Gießler Medek, André, et al. 2015. “Differenzanalyse komplexer Textvarianten”. *Datenbank-Spektrum* 15 (1): 25–31. doi:[10.1007/s13222-014-0173-y](https://doi.org/10.1007/s13222-014-0173-y).
- Needleman, Saul B., and Christian D. Wunsch. 1970. “A general method applicable to the search for similarities in the amino acid sequence of two proteins”. *Journal of Molecular Biology* 48 (3): 443–453. doi:[10.1016/0022-2836\(70\)90057-4](https://doi.org/10.1016/0022-2836(70)90057-4).
- Nyhan, Julianne, and Andrew Flinn. 2016. “The University Was Still Taking Account of *universitas scientiarum*: Wilhelm Ott and Julianne Nyhan”. In *Computation and the Humanities. Towards an Oral History of Digital Humanities*, 55–73. Springer Series on Cultural Computing. Cham: Springer. doi:https://doi.org/10.1007/978-3-319-20170-2_4.
- Oakman, Robert. 1972. “The Present State of Computerized Collation”. *Proof* 2:319–345.
- Orlandi, Tito. 2010. *Informatica testuale. Teoria e prassi*. Roma: Editori Laterza.
- Ott, Wilhelm. 1989. “Vom Manuskript zur Edition. Das Programm SATZ als Baustein in TUSTEP”. In *Historische Edition und Computer*, ed. by Anton Schwob, Karin Kranich-Hofbauer, and Diethard Suntinger, 153–176. Graz: Leykam.

- . 1991. “TUSTEP”. In *Computers in the humanities and the social sciences: Achievements of the 1980s, prospects for the 1990s. Proceedings of the Cologne Computer Conference 1988 uses of the computer in the humanities and social sciences held at the University of Cologne, Sept*, ed. by Heinrich Best, 432–437. Köln: De Gruyter.
- Parker, David C. 2000. “The Text of the New Testament and Computers: The International Greek New Testament Project”. *Literary and Linguistic Computing* 15 (1): 27–41.
- . 2006. “Electronic religious texts : the Gospel of John”. In *Electronic Textual Editing*, ed. by Lou Burnard, Katherine O’Brien O’Keeffe, and John Unsworth, 202–205. New York: Modern Language Association of America.
- . 2008. *An Introduction to the New Testament Manuscripts and their Texts*. Cambridge: Cambridge University Press.
- . 2012. *Textual Scholarship and the Making of the New Testament*. Oxford: Oxford University Press.
- Pasquali, Giorgio. 1952. *Storia della Tradizione e Critica del Testo*. 2nd ed. Firenze: Felice Le Monnier.
- Pearl, Judea. 1984. *Heuristics: Intelligent Search Strategies for Computer Problem Solving*. Reading, Mass.: Addison-Wesley.
- Peirce, Charles Santiago Sanders. 1906. “Prolegomena to an apology for pragmatism”. *The Monist* 16 (4): 492–546.
- Petti, Anthony E. 1977. *English Literary Hands from Chaucer to Dryden*. London: Edward Arnold.
- Pierazzo, Elena. 2008. “Editorial Teamwork in a Digital Environment : The Edition of the Correspondence of Giacomo Puccini”. In *Jahrbuch für Computerphilologie*, ed. by Malte Rehbein and S. Ryder, 91–110.
- . 2011. “A rationale of digital documentary editions”. *Literary and Linguistic Computing* 26 (4): 463–477. doi:<https://doi.org/10.1093/llc/fqr033>.
- . 2015. *Digital Scholarly Editing: Theories, Models and Methods*. Farnham, Surrey, UK; Burlington, VT: Ashgate.
- . 2016. “Textual Scholarship and Text Encoding”. In *A New Companion to Digital Humanities*, ed. by Susan Schreibman, Raymond G Siemens, and J M Unsworth, 307–321. Wiley-Blackwell.
- Pirovano, W., and J. Heringa. 2008. “Multiple sequence alignment.” In *Bioinformatics. Methods in Molecular Biology™*, ed. by J. M. Keith, vol. 452. Humana Press. doi:https://doi.org/10.1007/978-1-60327-159-2_7.

- Pithoeus, Petrus, ed. 1594. *M. Fab. Quintiliani Declamationes, quae ex CCCLXXXVIII. supersunt, CXLV. ex vetere exemplari restitutae. Calpurnii Flacci excerptae X. Rhetorum minorum LI. nunc primum editae. Dialogus de oratoribus, sive de caussis corruptae eloquentiae. Ex bibliotheca P.* Heidelberg: Hieronymus Commelinus. doi:<http://dx.doi.org/10.3931/e-rara-3278>.
- Poulos, Alex. 2014. "Editing with Juxta and the CTE | The Poulos Blog". Visited on July 13, 2017. <https://mapoulos.wordpress.com/2014/06/07/editing-with-juxta-and-the-cte/>.
- Prebor, Gila. 2013. "New Technologies for the Collation of Hebrew Texts". *Zutot: Perspectives on Jewish Culture* 10:53–64. doi:10.1163/18750214-12341254.
- Raabe, Wesley. 2008a. "Collation in Scholarly Editing: An Introduction". Visited on Aug. 25, 2015. <http://wraabe.wordpress.com/2008/07/26/collation-in-scholarly-editing-an-introduction-draft/>.
- . 2008b. "How to Be a Human Hinman Collator". Visited on May 4, 2015. <https://wraabe.wordpress.com/2008/05/13/how-to-be-a-human-hinman-collator/>.
- . 2008c. "The Digital Archive and Literary Scholarship : Textual Collation for Dummies". Visited on Aug. 25, 2015. <https://wraabe.wordpress.com/2008/01/17/the-digital-archive-and-literary-scholarship-textual-collation-for-dummies/>.
- . 2014. "Running CollateX on your Macintosh OSX (Mavericks)". Visited on May 4, 2017. <https://wraabe.wordpress.com/2014/10/07/running-collatex-on-your-macintosh-osx-mavericks/>.
- . 2015. "CollateX, Python, Anaconda, Oh My: Or, What Have I Done? (Week 3 Reflections)". Visited on May 4, 2017. <https://wraabe.wordpress.com/2015/09/21/collatex-python-anaconda-oh-my-or-what-have-i-done-week-3-reflections/>.
- Raben, Joseph. 1979. "De Acibus et Faeni Acervis: Text Comparison as a Means of Collation". In *La pratique des ordinateurs dans la critique des textes. Paris, 29-31 Mars 1978*, ed. by Jean Irigoien and G. P. Zarri, 256–261. Paris: Éditions du Centre National de la Recherche Scientifique.
- Ravy, Tawnya. 2015. "Juxta and Frankenstein | Studies In The Novel". Visited on July 13, 2017. <https://studiesinthenovel.org/content/juxta-and-frankenstein>.
- Raynaud, Dominique. 2014. "Building the stemma codicum from geometric diagrams". *Archive for History of Exact Sciences* 68 (2): 207–239. doi:<https://doi.org/10.1007/s00407-013-0134-0>.
- Reeve, Michael. 2000. "'Cuius in Usum?' Recent and Future Editing". *The Journal of Roman Studies* (Roma) 90:196–206.

- . 2011. “Editing classical texts with a computer: Hyginus’s *Astronomica*”. In *Manuscripts and Methods: Essays on Editing and Transmission*, 361–393. Roma: Edizioni Di Storia e Letteratura.
- . (1989) 2011. “Eliminatio codicum descriptorum: A methodological problem”. In *Manuscripts and Methods: Essays on Editing and Transmission*, ed. by Michael Reeve, 145–174. Repr., Roma: Edizioni Di Storia e Letteratura.
- Reitz, Joan M. 2013. “Collation”. Visited on Oct. 23, 2014.
http://www.abc-clio.com/ODLIS/odlis_c.aspx.
- Reynolds, Leighton D., and Nigel G. Wilson. 1991. *Scribes and scholars. A guide to the transmission of Greek and Latin literature*. 3rd ed. Oxford: Clarendon Press.
- Rizzo, Silvia. 1973. *Il lessico filologico degli umanisti*. Roma: Edizioni di storia e letteratura.
- Robinson, Peter. 1989a. “The Collation and Textual Criticism of Icelandic Manuscripts (1): Collation”. *Literary and Linguistic Computing* 4 (2): 99–105.
- . 1989b. “The Collation and Textual Criticism of Icelandic Manuscripts (2): Textual Criticism”. *Literary and Linguistic Computing* 4 (3): 174–181.
- . 1991. “Collation, Textual Criticism, Publication, and the Computer”. *Text* 7:77–94.
- . 1994. “Collate: A Program For Interactive Collation of Large Textual Traditions”. In *Research in Humanities Computing* 3, ed. by Susan Hockey and Nancy Ide, 32–45. Oxford: Oxford University Press.
- . 2004. “Rationale and Implementation of the Collation System Used on this CD-ROM”. In *The Miller’s Tale on CD-ROM*. Leicester, UK: Scholarly Digital Editions.
- . 2005. “Current issues in making digital editions of medieval texts - or, do electronic scholarly editions have a future?” *Digital Medievalist* 1.
doi:[tpp://doi.org/10.16995/dm.8](http://doi.org/10.16995/dm.8).
- . 2007a. “Electronic Editions Which We Have Made and Which We Want to Make”. In *Digital Philology and Medieval Texts*, ed. by Arianna Ciula and Francesco Stella, 1–12. Pisa: Pacini.
- . 2007b. “The History of Collate”. Visited on June 9, 2014.
<http://www.sd-editions.com/blog/?p=15>.

- . 2009. “Towards a Scholarly Editing System for the Next Decades Introduction : A Short History”. In *Sanskrit Computational Linguistics: First and Second International Symposia Rocquencourt, France, October 29-31, 2007 Providence, RI, USA, May 15-17, 2008, Revised Selected Papers*, ed. by Gérard Huet, Amba Kulkarni, and Peter Scharf, 346–357. Berlin, Heidelberg: Springer.
 - . 2014. “Scholarly Digital Editions: Collate 2, and the design for its successor: CollateXML (now, CollateX)”. Visited on Aug. 5, 2018. <http://scholarlydigitaleditions.blogspot.com/2014/09/collate-2-and-design-for-its-successor.html>.
 - . 2017. “Some principles for making collaborative scholarly editions in digital form”. *Digital Humanities Quarterly* 11 (2).
- Robinson, Peter, and Elizabeth Solopova. 1993. “Guidelines for Transcription of the Manuscripts of the Wife of Bath’s Prologue”. In *The Canterbury Tales Project: Occasional Papers*, ed. by Norman F Blake and Peter Robinson, 19–52. Oxford: Oxford University Computing Service.
- Roelli, Philip. 2014. “Petrus Alfonsi or On the mutual benefit of traditional and computerised stemmatology”. In *Analysis of Ancient and Medieval Texts and Manuscripts: Digital Approaches*. Ed. by Tara Andrews and Caroline Macé, 43–68. Turnhout: Brepols. doi:<https://doi.org/10.1484/M.LECTIO-EB.5.102564>.
- Roelli, Philip, and Dieter Bachmann. 2010. “Towards generating a stemma of complicated manuscript traditions: Petrus Alfonsi’s Dialogus”. *Revue d’Histoire des Textes* 5:307–321.
- Rolfe, John C. 1927. *The Attic Nights of Aulus Gellius. With An English Translation*. Cambridge Mass. ; London: Harvard University Press.
- Roos, Teemu, and Yuan Zou. 2011. “Analysis of textual variation by latent tree structures”. *Proceedings - IEEE International Conference on Data Mining, December 11-14, Vancouver, 2011*: 567–576. doi:[10.1109/ICDM.2011.24](https://doi.org/10.1109/ICDM.2011.24).
- Ruffell, Ian. 2015. “Supporting MS collation”. Digital Classicist Archives. Visited on July 31, 2017. <https://www.jiscmail.ac.uk/cgi-bin/webadmin?A2=digitalclassicist;f8cb739e.1509>.
- Russell, D. A. 1983. *Greek Declamation*. Cambridge: Cambridge University Press.
- Sabbadini, Remigio. 1905. *Le scoperte dei codici latini e greci ne’secoli 14 e 15*. Firenze G.C. Sansoni.
- Safaryan, Anahit, Sascha Kaufmann, and Tara L. Andrews. 2016. “Critical Edition as Graph: The Chronicle of Matthew of Edessa Online”. Digital Humanities Conference DH2016, Kraków, 11-16 July 2016. Visited on June 5, 2017. <http://dh2016.adho.org/abstracts/209>.

- Sahle, Patrick. 2013. *Digitale Editionsformen. Zum Umgang mit der Überlieferung unter den Bedingungen des Medienwandels. Teil 3: Textbegriffe und Recodierung*. Norderstedt: Books on Demand.
- . 2017. “A catalog of Digital Scholarly Editions v3.0”. Visited on Jan. 10, 2018. <http://www.digitale-edition.de/index.html>.
- Sahraeian, Sayed Mohammad Ebrahim, and Byung Jun Yoon. 2011. “PicXAA-Web: a web-based platform for non-progressive maximum expected accuracy alignment of multiple biological sequences”. *Nucleic Acids Res* 39 (April): W8–12. doi:10.1093/nar/gkr244.
- Saito, Ken. 2006. “A preliminary study in the critical assessment of diagrams in Greek mathematical works”. *Sciamvs* 7:81–144.
- Salemans, Benedictus Johannes Paulus. 2000. *Building Stemmas with the Computer in a Cladistic, Neo-Lachmannian, way. The Case of Fourteen Text Versions of Lanseloet van Denemerken*. Nijmegen: Katholieke Universiteit Nijmegen.
- Schironi, Francesca. 2012. “The Ambiguity of Signs: Critical *σημεῖα* from Zenodotus to Origen”. In *Homer and the Bible in the Eyes of Ancient Interpreters*, ed. by Maren R. Niehoff, 87–112. Leiden; Boston: Brill.
- Schmid, Ulrich. 2004. “Genealogy by chance! On the significance of accidental variation (parallelisms)”. In *Studies in Stemmatology II*, ed. by Pieter van Reenen, August den Hollander, and Margot van Mulken, 127–144. Amsterdam: John Benjamins Publishing Company.
- Schmidt, Desmond. 2009. “Merging Multi-Version Texts: a Generic Solution to the Overlap Problem”. In *Proceedings of Balisage: The Markup Conference 2009. August 11 - 14, 2009*. Vol. 3. doi:10.4242/BalisageVol3.Schmidt01.
- . 2010. “The inadequacy of embedded markup for cultural heritage texts”. *Literary and Linguistic Computing* 25 (3): 337–356. doi:10.1093/lc/fqq007.
- . 2013. “Collation on the Web”. Digital Humanities Conference DH2013, University of Nebraska–Lincoln, 16-19 July 2013. Visited on May 14, 2017. <http://dh2013.unl.edu/abstracts/ab-108.html>.
- Schmidt, Desmond, and Robert Colomb. 2009. “A data structure for representing multi-version texts online”. *International Journal of Human-Computer Studies* 67 (6): 497–514. doi:10.1016/j.ijhcs.2009.02.001.
- Schmidt, Desmond, and Paul Eggert. 2015. “Technical Design”. Visited on May 30, 2017. <http://charles-harpur.org/harpur/tabs?docid=english/harpur/about/technical%7B%5C&%7Dmodule=para%7B%5C%%7D3Fdocid%7B%5C%7D3Denglish/harpur/about/technical%7B%5C&%7Dtabset=about>.

- Schubert, Charlotte, et al. 2016. *eComparatio - Editionsvergleich*. Digital Humanities Conference DH2016, Kraków, 11-16 July 2016.
- Schulz, Daniela. 2017. "Of general and homemade encoding problems". In *Advances in Digital Scholarly Editing*, ed. by Peter Boot et al., 341–344. Sidestone Press.
- Schütz, Susanne, and Marcus Pöckelmann. 2016. "LERA - Explorative Analyse komplexer Textvarianten in Editionsphilologie und Diskursanalyse". In *Vortrag auf der 3. Jahrestagung der Digital Humanities im deutschsprachigen Raum, DHd 2016, Leipzig 07.-12.03.2016*.
- Sculley, David, and B. M. Pasanek. 2008. "Meaning and mining: the impact of implicit assumptions in data mining for the humanities". *Literary and Linguistic Computing* 23 (4): 409–424. doi:10.1093/lc/fqn019.
- Shaw, Prue, ed. 2010. *Dante Alighieri: Commedia. A Digital Edition*. Saskatoon: Scholarly Digital Editions.
- Shillingsburg, Miriam J. 1978. "Computer Assistance to Scholarly Editing". *Bulletin of Research in the Humanities* 81:448–473.
- Shillingsburg, Peter. 1980. "The Computer as Research Assistant in Scholarly Editing". *Literary Research Newsletter* 5 (1): 31–45.
- . 1996. *Scholarly editing in the computer age: theory and practice*. 3rd ed. Ann Arbor: University of Michigan Press.
- . 2014. "Development Principles for Virtual Archives and Editions". *Variants* 11:11–28.
- Siemens, Raymond. 1994. "Textual collation software for the PC. PC-CASE, UNITE, and the Donne Variorum Collation Program". *Text Technology* 4 (3): 209–222.
- . 2009. "Editing the Early Modern Miscellany: Modelling and Knowledge [Re]Presentation as a Context for the Contemporary Editor". In *New Ways of Looking at Old Texts IV: Papers of the Renaissance English Text Society, 2002–2006*, ed. by Michael Denbog, 115–130.
- Smith, Catherine. 2015. "Developing a Scholarly Collation Editor for New Testament Manuscripts". DiXiT Convention 1: Technology, Software, Standards for the Digital Scholarly Edition September 14-18, 2015. Visited on Nov. 20, 2017. http://dixit.huygens.knaw.nl/?page%7B%5C_%7Ddid=138%7B%5C#%7Dsmith.
- Smith, David Neel, and Sephanie Lindeborg. 2016. "Comparing Digital Scholarly Editions". Digital Humanities Conference DH2016, Kraków, 11-16 July 2016. Visited on Apr. 21, 2017. <http://dh2016.adho.org/abstracts/141>.

- Smith, Steven Escar. 2000. "'The Eternal Verities Verified': Charlton Hinman and the Roots of Mechanical Collation". *Bibliographical Society of the University of Virginia* 53:129–161.
- . 2002. "'Armadillos of Invention': A Census of Mechanical Collators". *Studies in Bibliography* 55:133–170.
- Spadini, Elena. 2015. "Annotating Document Changes". In *Proceedings of the 3rd International Workshop on (Document) Changes: modeling, detection, storage and visualization Lausanne, Switzerland — September 08 - 08, 2015*, 23–26.
- . 2016. "Studi sul Lancelot en prose". PhD thesis, Sapienza Università di Roma.
- Spencer, Matthew, and Christopher J. Howe. 2004. "Collating Texts Using Progressive Multiple Alignment". *Computers* 38 (3): 253–270.
- Spencer, Matthew, et al. 2004. "The effects of weighting kinds of variants". In *Studies in Stemmatology II*, ed. by Pieter Van Reenen, August Den Hollander, and Margot Van Mulken, 227–240. Studies in Stemmatology. Amsterdam: John Benjamins Publishing Company.
- Sperberg-McQueen, C. M. 1989. "A directed-graph data structure for text manipulation". <http://cmsmcq.com/1989/rhine-delta-abstract.html>.
- . 2009. "How to teach your edition how to swim". *Literary and Linguistic Computing* 24 (1): 27–39. doi:10.1093/lc/fqn034.
- Stählin, Otto. 1914. *Editionstechnik*. London & Berlin: Teubner.
- Stokes, Peter. 2012. "Modeling Medieval Handwriting: A New Approach to Digital Palaeography | Digital Humanities 2012". In *DH2012 Book of Abstracts*, ed. by J.C. et al. Meister, 382–385. Hamburg: University of Hamburg.
- Stringer, Gary A., and William R. Vilberg. 1987. "The Donne Variorum Textual Collation Program". *Computers and the Humanities* 21 (2): 83–89.
- Stussi, Alfredo. 1994. *Introduzione agli studi di filologia italiana*. Manuali : Filologia e critica letteraria. Bologna: Il Mulino.
- Sussman, Lewis A. 1994. *The Declamations of Calpurnius Flaccus: Text, Translation, and Commentary*. Ed. by Lewis A. Sussman. Mnemosyne Bibliotheca Classica Batava. Leiden: New York: E.J. Brill.
- . 2013. "Controversiae". In *The Encyclopedia of Ancient History*, First Edition. Online version, ed. by Roger S. Bagnall et al. Blackwell Publishing Ltd. doi:10.1002/9781444338386.wbeah13057.
- TEI Consortium eds. 2013. "JuxtaCommons". Visited on Apr. 10, 2017. <https://wiki.tei-c.org/index.php/JuxtaCommons>.

- , ed. 2017a. “10 Manuscript Description”. Visited on Aug. 25, 2017.
<http://www.tei-c.org/release/doc/tei-p5-doc/en/html/MS.html>.
 - . 2017b. “11.1 Digital Facsimiles”. Visited on Mar. 5, 2017.
<http://www.tei-c.org/release/doc/tei-p5-doc/en/html/PH.html#PHFAX>.
 - . 2017c. “12.2.3 The Parallel Segmentation Method”. Visited on Jan. 31, 2017.
<http://www.tei-c.org/release/doc/tei-p5-doc/en/html/TC.html#TCAPPS>.
 - . 2017d. “3.12.1 Core Tags for Verse”. Visited on Dec. 5, 2017.
<http://www.tei-c.org/release/doc/tei-p5-doc/en/html/CO.html#COVE>.
- Ter Braake, S., et al. 2016. “Digital history: Towards new methodologies”. *IFIP Advances in Information and Communication Technology* 482.
doi:10.1007/978-3-319-46224-0_3.
- Timpanaro, S. 1985. “Recentiores e deteriores, codices descripti e codices inutilis”.
Filologia e critica 10:164–192.
- . 2005. *The Genesis of Lachmann's Method*. Ed. and trans. by Glenn W Most. Chicago and London: University of Chicago Press.
- Tissoni, Francesco. 2004. “Testi latini on line ad accesso libero: una prima valutazione”. *ACME* LV:43–79.
- . 2008. “Pubblicare testi latini on-line. Obiettivi, metodi e strategie”. In *Prassi Ecdotiche. Esperienze editoriali su tesi manoscritti et testi a stampa. Milano, 7 giugno e 31 ottobre 2007*, ed. by Alberto Cadioli and Paolo Chiesa, 137–154. Milano: Cisalpino, Istituto Editoriale Universitario.
- Trovato, Paolo. 2014. *Everything You Always Wanted to Know about Lachmann's Method. A Non-Standard Handbook of Genealogical Textual Criticism in the Age of Post-Structuralism, Cladistics, and Copy-Text*. Padova: libreriauniversitaria.it edizioni.
- Underwood, Ted, and Jordan Sellers. 2015. “How Quickly Do Literary Standards Change?”: 1–37. doi:<https://doi.org/10.6084/m9.figshare.1418394.v1>.
- Urbina, Eduardo, et al. 2002. “Critical Editing in the Digital Age: Informatics and Humanities Research”. In *The new Information Order and the Future of Archive. Proceedings of a Conference held at Old College, the University of Edinburgh (March 20-23, 2002)*, ed. by J. Frow. Edinburgh: University of Edinburgh.
- Valla, Lorenzo. 1540. *Laurentii Vallae Opera*. Apud Henricum Petrum.
- Van Hulle, Dirk, Vincent Neyt, and Mark Nixon. 2014. “Update: collation feature now containing deletions and additions”. Visited on Feb. 1, 2018.
<http://www.beckettarchive.org/news.jsp>.

- Van Mal-Maeder, Danielle. 2007. *La Fiction Des Déclamations*. Leiden: Boston: Brill.
- Van Ossenbruggen, Jacco. 2015. "The Nature of Digitally-Produced Data: Towards Social-Scientific Tool Criticism". Visited on July 10, 2017. <https://www.slideshare.net/jrvosse/the-nature-of-digitallyproduced-data-towards-socialscientific-tool-crititism>.
- Van Zundert, Joris. 2016. *Close Reading and Slow Programming — Computer Code as Digital Scholarly Editions*. Presentation given at the ESTS-DiXiT conference, Antwerp, 5-7 October 2016.
- Van Zundert, Joris, and Ronald Haentjens Dekker. 2017. "Code, scholarship, and criticism: When is code scholarship and when is it not?" *Digital Scholarship in the Humanities* 32 (suppl_1): i121–i133. doi:10.1093/lc/fqx006.
- VanderPlas, Jake. 2016. *Python Data Science Handbook*. O'Reilly Media.
- Vanhoutte, Edward. 2010. "Defining Electronic editions: A Historical and Functional Perspective". In *Text and Genre in Reconstruction. Effects of Digitalization on Ideas, Behaviours, Products and Institutions*, ed. by Willard McCarty, 119–144. Cambridge: Open Book Publisher.
- . 2011. "So You Think You Can Edit? The Masterchef Edition". Visited on Aug. 14, 2014. <http://edwardvanhoutte.blogspot.co.uk/2011/10/so-you-think-you-can-edit-masterchef.html>.
- Vierros, Marja, and Erik Henriksson. 2016. "Preprocessing Greek Papyri for Linguistic Annotation". Episciences.org, 2017, Special Issue on Computer-Aided Processing of Intertextuality in Ancient Languages. Visited on Dec. 5, 2016. <https://hal.archives-ouvertes.fr/hal-01279493v1>.
- Vinaver, Eugène. 1939. "Principles of Textual Emendation". In *Studies in French Language and Mediaeval Literature. Presented to Professor Mildred K. Pope, by Pupils, Colleagues, and Friends*. 351–369. Manchester: Publications of the University of Manchester.
- Viré, Ghislaine. 1986. *Informatique et classement des Manuscrits. Essai méthodologique sur le de astronomia d'Hygin*. Bruxelles: Editions de l'Université de Bruxelles.
- Wachtel, Klaus. 2000. "Editing the Greek New Testament on the Threshold of the Twenty-First Century". *Literary and Linguistic Computing* 15 (1): 43–50.
- Watt, W. S. 1996. "Ten notes on Calpurnius Flaccus, *Declamationum Excerpta*". *Eranos* 94:123–127.
- West, Martin L. 1973. *Textual Criticism and Editorial Technique*. Stuttgart: Teubner.

- Wheeles, Dana. 2013. "Using Juxta in the Classroom: Scholar's Lab Presentation". Visited on Aug. 29, 2017. <http://www.juxtaoftware.org/using-juxta-in-the-classroom-scholars-lab-presentation/>.
- Wheeles, Dana, and Kristen Jensen. 2014. "Juxta Commons". *Journal of Digital Humanities* 3 (1).
- Whittaker, John. 1991. "The Practice of Manuscript Collation". *Text* 5:121–130.
- Widmann, R. L. 1971a. "Computers and Literary Scholarship". *Computers and the Humanities* 6 (1): 3–14.
- . 1971b. "The computer in historical collation: use of the IBM 360/75 in collating multiple editions of *A Midsummer Night's Dream*". In *The Computer in Literary and Linguistic Research*, ed. by R. A. Wisbey, 57–63. Cambridge: Cambridge University Press.
- Wiering, Frans. 2010. "Digital Critical Editions of Music: A Multidimensional Model". Chap. 3 in *Modern Methods for Musicology*, ed. by Gibson and Crawford, 23–45. Farnham, Surrey, UK; Burlington, VT: Ashgate.
- Willis, James. 1972. *Latin Textual Criticism*. Illinois University Language and Culture Series. Urbana, Chicago, London: University of Illinois Press.
- Winterbottom, Michael. 1984. *The minor declamations ascribed to Quintilian*. Berlin: De Gruyter.
- . 1995. "Review of L. A. Sussman 'The Declamations of Calpurnius Flaccus. Text, Translation, and Commentary'". *The Classical Review* 45 (01): 40–42. doi:10.1017/S0009840X00292032.
- . 1999. "An Emendation in Calpurnius Flaccus". *The Classical Quarterly* 49, no. 01 (0): 338–339. doi:10.1093/cq/49.1.338.
- . 2017. "The Editors of Calpurnius Flaccus". In *Reading Roman Declamation - Calpurnius Flaccus*, ed. by Martin T. Dinter, Charles Guérin, and Marcos Martinho, 143–162. Berlin, Germany ; Boston, Massachusetts: De Gruyter.
- Winters, Margaret E. 1991. "Manuscript Variation and Syntactic Change". *Text* 5:131–144.
- Yousef, Tariq, and Chiara Palladino. 2016. *iAligner : A tool for syntax-based intra-language text alignment*.
- Yousef, Tariq, Chiara Palladino, and Gregory Crane. 2017. *Intra-language Text Alignment Using iAligner*.

Bibliography

Zeevaert, Ludger. 2015. "Easy tools to get to grips with linguistic variation in the manuscripts of Njáls saga." *Digital Medievalist* 10.

doi:<http://doi.org/10.16995/dm.60>.

Zetzel, James E. G. 1981. *Latin Textual Criticism in Antiquity*. New York: Arno Press.