Leveraging combinations of epigenomic regulators

Quentin Ferré

Aix*Marseille université
Initiative d'excellence

NNT/NL : 2021AIXM0151/031ED62

THÈSE DE DOCTORAT

Soutenue à Aix-Marseille Université
le 23 mars 2021 par

Quentin FERRÉ

Leveraging combinations of epigenomic regulators

Discipline
Biologie santé

Spécialité
Génomique et bioinformatique

École doctorale
ED 62 (Sciences de la Vie et de la Santé)

Laboratoire/Partenaires de recherche
INSERM U1090 - TAGC
CNRS UMR7020 LIS - Qarma

Composition du jury

Stein AERTS University of Leuven	Rapporteur
Carl HERRMANN BioQuant - Medical Faculty Heidelberg	Rapporteur
Delphine POTIER CIML - AMU	Examinatrice
Nelle VAROQUAUX TIMC-IMAG	Examinatrice
Jacques VAN HELDEN TAGC	Directeur de thèse
Cécile CAPPONI Qarma team - LIS	Co-directrice de thèse
Denis PUTHIER TAGC	Invité

Abstract

Genetic cis-regulation in humans is effected through chromatin regulators, such as histone marks and Transcriptional Regulators (TRs), binding on regions called Cis-Regulatory Elements. Those regulators seldom act alone, forming complexes to perform their functions. For example, while Transcription Factors are regulatory proteins that bind directly to DNA, they are themselves bound by co-factors. The goal of these interacting systems is to regulate gene expression by influencing the activity of the RNA Pol II, which transcribes DNA to messenger RNA. The development of Next Generation Sequencing provides experimental methods to study this regulation, which includes ChIP-seq and other assays. Their main goal is to quantify both chromatin accessibility and protein binding. However, these methods present challenges and sources of noise, where noise is defined as any result differing from the biological reality being quantified. They also suffer from reproducibility problems, hence complicating fair comparison among results. Both these biases are difficult to correct. Besides combinations of regulators themselves, the recent explosion of available data volume, as well as variety of sources, collated in databases such as ENCODE or ReMap gives opportunities for integrating different data views.

While combinations of biological regulators are important to genomic cis-regulation, they are seldom operated for biological insight. Existing approaches suffer from either the precision of the data integration, or the clarity of usage. The goal of this thesis is to leverage such combinations through the use of machine learning methods, which are very effective at learning regularities in the data: in other words, learning combinations. We propose to represent the regions where regulators bind as lists of intervals, converted into matrix and tensor representations. As a result, the approaches of this thesis are generalizable to any lists of intervals. Early work presented in this thesis discusses the prediction of cis-regulatory region status and the detection of alternative promoters in T-ALL leukemia. We propose a new method, based on Cramer's V-score, to robustly identify meaningful alternative promoters in based on promoter expression, discarding low-level noise.

Then, we focus on anomaly detection. ChIP-seq and other experimental assays can suffer from errors and false positives, poor quality control, and several other biases. Those are very difficult to correct, as annotated supervised data is rarely available, and even so it would require a tedious error-by-error approach. Furthermore, the indiscriminate use of larger volumes of data increases the probability of erroneous observations. Instead, we perform unsupervised anomaly detection under the assumption that noise peaks will not respect the usual combinations between sources (ie. combinations between regulators and/or usual dataset combinations). We propose

the atyPeak method which exploits not only combinations of TRs, but also combinations of redundant experiments from the ReMap database. We propose to use a specifically designed multi-view convolutional autoencoder to perform a “Goldilocks” compression. Here, the model is tasked to learn sources (TR, datasets) as part of a groups of correlating sources and not alone. As a result, ChIP-Seq peaks are rebuild as part of a correlation group and rare noisy patterns are not even learned. We identify peaks which have fewer known collaborators present in their vicinity than what would be average for their sources. In terms of methodology, we developed approaches to evaluate autoencoders based on their respect of existing correlations. We also propose a new normalization method based on correcting for the average cardinality of the aforementioned correlation groups. It can be applied to any black box model, and is useful to interpret autoencoders when performing anomaly detection. Our cleaned data improves Cis-Regulatory Element detection.

Finally, on a more fundamental level, the enrichment of given combinations of elements (meaninghow much more often they are found compared to expected by chance) needs to be precisely quantified. We propose the OLOGRAM-MODL approach, demonstrating a Monte Carlo based method to fit a novel Negative Binomial model on the number of base pairs on which a given combinations of elements is observed. This allows us to return much more precise p-values compared to existing approaches. We extend this model to combinations of any $k \geq 2$ elements. We also propose a suited itemset mining algorithm to identify interesting combinations of regulators, based on which itemsets best rebuild the original data. This algorithm leverages dictionary learning for its robustness to noise. Additionally, we demonstrate that the problem is submodular and that a greedy algorithm can find itemsets of interest. This tool was implemented as a part of the gtftk toolset for ease of access.

Keywords: epigenomic regulators, combinations, machine learning, Cis-Regulatory Elements, autoencoders, statistical modeling, Monte Carlo

Résumé

La régulation cis-génomique chez l’homme est effectuée par des régulateurs de la chromatine, tels que les marques d’histones et les régulateurs de transcription (TR), qui se lient à des éléments cis-régulateurs (CRE). Ils fonctionnent rarement seuls, mais plutôt en complexes. Par exemple, les facteurs de transcription (TFs) se lient à l’ADN et sont eux-mêmes liés par des cofacteurs. Leur objectif est de réguler l’activité de l’ARN Pol II. Le développement du séquençage de nouvelle génération (NGS) fournit des méthodes pour étudier cette régulation, incluant le ChIP-seq, afin de quantifier l’accessibilité de la chromatine et la liaison des protéines. Mais ces méthodes présentent des sources de bruit (résultats différents de la réalité), et des problèmes de reproductibilité, ce qui complique la comparaison des résultats. De plus, la récente explosion de la variété et du volume de données disponibles, dans des

bases de données telles que ENCODE ou ReMap, permet l'intégration de différentes vues de données.

Les combinaisons de régulateurs biologiques sont importantes mais sont rarement exploitées. Les approches existantes manquent de précision ou de clarté. Le but de cette thèse est de tirer parti de ces combinaisons en utilisant des méthodes d'apprentissage automatique, qui sont efficaces pour apprendre les régularités dans les données : donc, les combinaisons. Nous représentons les régions d'intérêt sous forme de listes d'intervalles, converties en représentations matricielles et tensorielles. De fait, nos approches sont généralisables à toute liste d'intervalles. Les premiers travaux présentés dans cette thèse portent sur la prédiction du statut des CRE et la détection robuste de promoteurs alternatifs dans la leucémie T-ALL en fonction de leur expression, éliminant le bruit de faible niveau.

Ensuite, nous abordons la détection d'anomalies non supervisée. Le ChIP-seq (et autres) peut souffrir d'erreurs et de faux positifs, d'un contrôle de qualité médiocre et de plusieurs autres biais. Ceux-ci sont difficiles à corriger, car les données annotées et supervisées sont rarement disponibles, et cela demanderait malgré tout une approche erreur-par-erreur fastidieuse. En outre, les grands volumes de données augmentent la probabilité d'erreurs. Au lieu de cela, nous supposons que le bruit ne respectera pas les combinaisons usuelles entre les sources (TR et/ou jeux de données). Nous proposons atyPeak, qui exploite les combinaisons de TR et d'expériences redondantes de ReMap. Nous utilisons un auto-encodeur convolutionnel multi-vues pour une compression "de juste milieu", en apprenant et reconstruisant les sources comme parties d'un groupe de sources corrélées et non pas seules, éliminant les motifs rares (bruit). Nous marquons les pics qui ont moins de collaborateurs à proximité que la moyenne de leur source. Nous proposons aussi des approches pour évaluer les auto-encodeurs selon de leur respect des corrélations de données, et une méthode de normalisation basée sur la cardinalité des groupes. Elles peuvent être appliquées à l'interprétation d'autres modèles. Nos données nettoyées améliorent la détection des CRE.

Enfin, l'enrichissement de combinaisons d'éléments (fréquence par rapport à ce qui est attendu au hasard) doit être quantifié avec précision. Nous proposons OLOGRAM-MODL, une méthode Monte Carlo ajustant un modèle binomial négatif sur le nombre de paires de bases où une combinaison est observée. Cela renvoie des p-valeurs plus précises par rapport aux approches existantes. Nous l'étendons aux combinaisons de >2 éléments et proposons un algorithme d'extraction d'itemsets pour identifier les combinaisons intéressantes de régulateurs, qui reconstruisent le mieux les données d'origine. Nous utilisons l'apprentissage par dictionnaire pour sa robustesse au bruit. Nous montrons que le problème est sous-modulaire et qu'un algorithme glouton peut trouver ces ensembles intéressants. Il a été implémenté dans le jeu d'outils gtftk.

Acknowledgements

Well, here we are at last. The dust has settled, and three years have passed. I now find myself older than when I began, that is for certain. And, perhaps, a whit wiser after all? But that is for you to judge, dear reader.

First and foremost, I would like to thank Stein Aerts, Carl Herrmann, Delphine Potier and Nelle Varoquaux for agreeing to review my thesis work. I would also like to thank Salvatore Spicuglia, Laurent Tichit and Badih Ghattas for their participation in my thesis committee. Further thanks to Catherine Nguyen, Pascal Rihet and Hachem Qadri for welcoming me in their laboratories.

Thanks to Jacques van Helden for accepting to be my official thesis advisor. Heartfelt thanks to my thesis advisors, Denis Puthier and Cécile Capponi, for putting up with my antics, but especially for believing in me. The lessons you taught me will stay with me forever. Thank you for allowing me to explore the wonderful world that lies at the interface of bioinformatics and machine learning.

I would also like to thank some friends and proverbial "war buddies". Thanks to Jeanne Chèneby for our scientific discussions and collaboration, and for our more esoteric historical and political ones. I am glad to now have the answer to the question, "Should the Holy Roman Emperor imprison his children, assuming they wish to murder the Pope?" It's existential questions such as these that net someone a spot on a governmental watchlist. Further thanks to Florian Rosier, Laurent Hannouche, and Pauline Brochet for our geek-ish discussions and card games (there was science in there too, I swear!).

Thanks to Benoît Ballester for his insight on this whole "thesis" thing, and for accepting to collaborate with me (God knows what possessed him that day). Further thanks to Salvatore Spicuglia, Guillaume Charbonnier and Nori Sadouni for giving me the opportunity to explore fascinating biological thematics in our collaborations.

My most academic thanks to the entire teaching staff for welcoming me in their strides, and giving me the opportunity to bore to tears three new generations of Master's students by inflicting **THE MACHINE LEARNINGS** upon them.

Thanks to Marina Kreme, Alexis Prod'Homme and Zacharie Ménétrier for their perpetual joviality, and for our sport sessions. Particular thanks to all the Hippo'Thèse members for giving me such a warm welcome into the world of associative life, and for our unforgettable AfterLab outings !

Finally, my warmest thanks to my family for their support in this adventure. In particular to my grandmother, who taught me that hard work always pays off.

Acronyms

Biology

TR	Transcriptional Regulator
TF	Transcription Factor (a TR that binds directly to DNA)
TFBS	Transcription Factor Binding Site
TRBS	Transcriptional Regulator Binding Site
CRE	Cis-Regulatory Element (regulating region)
CRM	Cis Regulatory Module (local cluster or TRs)
RNA Pol II	RNA Polymerase II
TAD	Topologically Associated Domain
PIC	Pre-Initiation Complex

Genomic assays

NGS	Next Generation Sequencing
ChIP-Seq	Chromatin Immuno-Precipitation - Sequencing
IDR	Irreproducible Discovery Rate
WCE	Whole Cell Extract

Bioinformatics

BED	Browser Extensible Data
GTF	Gene Transfer Format
CLI	Command Line Interface

Machine Learning

ML	Machine Learning
PCA	Principal Component Analysis
LR	Learning Rate
NLP	Natural Language Processing
DL	Dictionary Learning

Statistics

RV	Random Variable
FDR	False Discovery Rate

Deep Learning

ANN	Artificial Neural Network
DNN	Deep Neural Networks
CNN	Convolutional Neural Network
LSTM	Long Short Term Memory

New acronyms introduced

Q-score	Quality Score
OLOGRAM	OverLap Of Genomic Regions Analysis using Monte Carlo
MODL	Multiple Overlap combinations with Dictionary Learning

Contents

List of Figures

List of Tables

1. Introduction

Sommaire

Let us begin this manuscript with the background information necessary to the comprehension of the research presented in this thesis. First, the biological context and actors of the genetic cis-regulation in humans are introduced (Section 1.1, p. 14) with a focus on those regulators which act in combinations. Then the experimental approaches used to characterize these regulators (Section 1.2, p. 31) and the challenges posed by the most recent ones are presented. The explosion of data volume and variety is discussed, but also the opportunities for integrating different data views that such an explosion now permits (Section 1.3, p. 39). Finally, the mathematical modelisation that will be used to characterize combinations of regulators throughout this thesis is presented, as well as background on the machine learning problems to which this work is relevant (Section 1.4, p. 52).

1.1. Human genetic cis-regulation

The expression **"human genome"** designates all the genetic material of a given human individual. As with all eukaryotes, it is stocked in the nucleus of all their nucleated cells (Figure 1.1), with the exception of the mitochrondrial chromosome. It is packaged inside the nucleus in a macromolecular structure called **chromatin**, constituted of desoxyribonucleic acid (DNA) combined with ribonucleic acid (RNA) and proteins. Its structure, as well as the impact of various DNA-binding proteins in regulating gene expression are presented in this section.

In humans, most cells contain the same genome, excepting processes such meiosis, somatic mutations and genomic rearrangements (e.g. mature B or T cells). However, cells differentiate to fulfill very different biological functions. Indeed, although all cells share a genome, their genomic expression patterns are very different. This implies

the existence of mechanisms capable of regulating the expression of all the different genes in the genome, resulting in different phenotypes.

The existence of such mechanisms has been postulated by Waddington in 1942, creating the science of epigenetics (Dupont, Armant, and Brenner 2009). While "epigenetics" originally designated heritable genetic modifications on anything but the DNA sequence, the term has now come to encompass many chromatin modification that affect gene expression independently of DNA sequence. Most current research supports the idea that the paramount epigenetic regulation mechanisms are modifications on histones and DNA, which we present here. Another part is due to the action of non-coding RNA on regulation (Frías-Lasserre and Villagra 2017), which is not studied here.

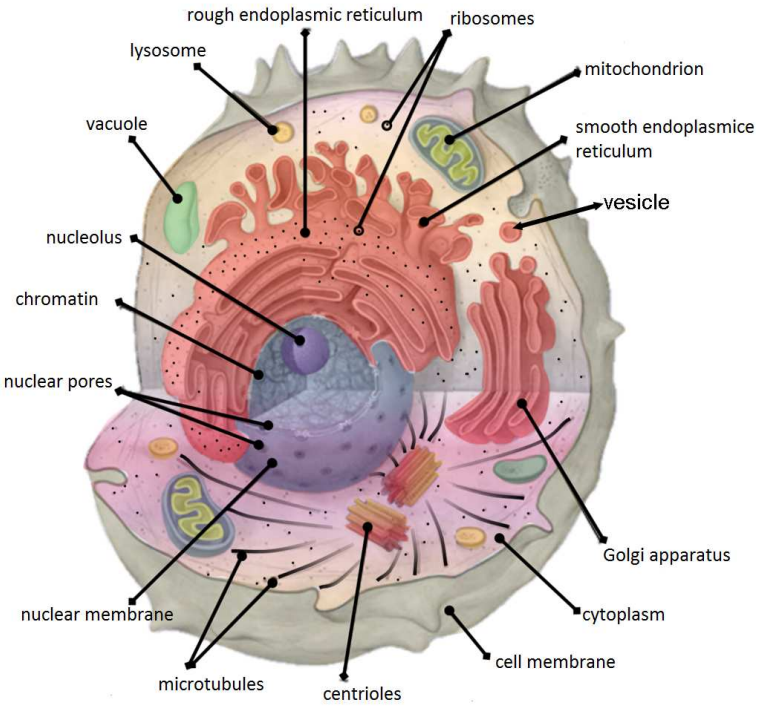


Figure 1.1. – Structure of a classical eukaryotic cell.

1.1.1. Regulatory proteins

1.1.1.1. Chromatin

. Figure 1.1 - Koswac / English Wikipedia / CC BY-SA 3.0
. Figure 1.2 - The Cell Cycle. Principles of Control. David O Morgan, 2007
. Figure 1.3 - Richard Wheeler (Zephyris) / English Wikipedia / CC BY-SA 3.0

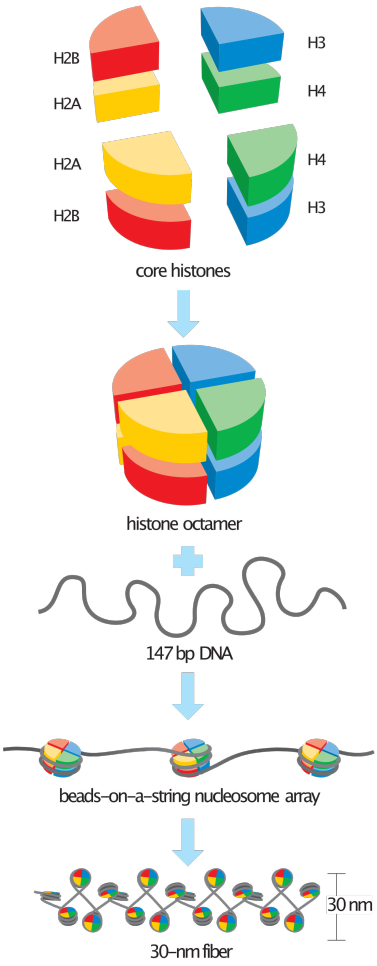


Figure 1.2. – Elementary structure and compaction levels of the nucleosomes.

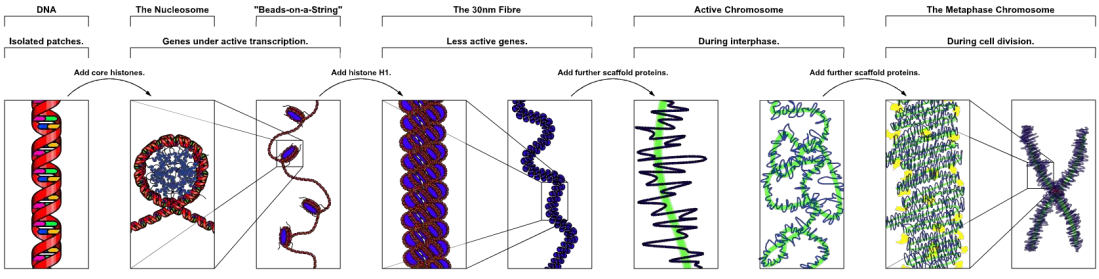


Figure 1.3. – Compaction levels of the chromatin. From left to right, lower to higher order. The top scale gives the structure name and the part of the cell or expression cycle in which the chromatin is found in this particular compaction level.

The main molecular components of chromatin are called histones, which are one of the main classes of DNA binding proteins (D. E. Olins and A. L. Olins 2003). The core canonical histones are H2A, H2B, H3 and H4. They form octamers around which DNA will bind, with each octamer having 147 bp of DNA wrapped around it. All together, this DNA-histone association consitutes a nucleosome. Histones of the H1 class bind the nucleosome together and regulate the chromatin's compaction level. Those nucleosomes are then packaged in a "beads-on-a-string" chromatin fiber (Figure 1.2). While this fiber is the elementary packaging structure of DNA in the cell, it is also a part of higher order structures of chromatin (Figure 1.3). It is combined with scaffold proteins, forming domains that interact with each other (Topologically Associated Domains), and ultimately the chromosomes themselves.

As far as genomic regulation is concerned, one of chromatin's major characteristics is its accessibility level. It varies depending on the cell, or the current stage of the transcription cycle. Open chromatin has a low density of nucleosomes and is called euchromatin, whose low compaction means the DNA can be accessed, recognized by DNA-binding proteins, and its genes transcribed for whatever use is required by the cell. This is contrast to high density chromatin, or heterochromatin.

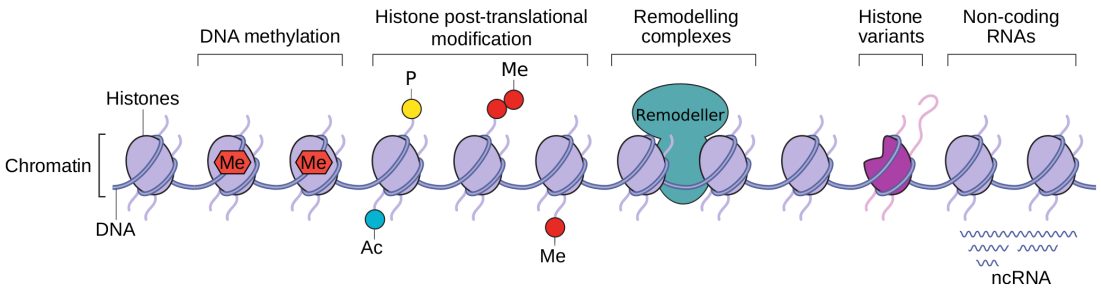


Figure 1.4. – Main chromatin epigenetic modifications.

The regulation of the compaction level is mostly done through various epigenetic, post-traductional mechanisms presented in this section. Figure 1.4 presents the main types of chromatin modifications encountered in humans, which we now discuss.

Histone modifications The main regulator of chromatin accessibility are post-traductional histone modifications of the N-terminal extremity of histones (Figure 1.5), notably on the lysines (K). Acetylation and methylation are generally considered as having opposed roles, although exceptions exist. This constitutes what is colloquially called the "histone code". A selection of representative histone marks with their effectors and their impact on gene transcription is presented in Table 1.1.

A general review of this code, of the impact of histone modifications on the chromatin and the means by which they are effected can be found in Bannister and

. Figure 1.4 - Dulac 2010

. Figure 1.5 - http://cnx.org/contents/GFy_h8cu@10.53:rZudN6XP@2/Introduction

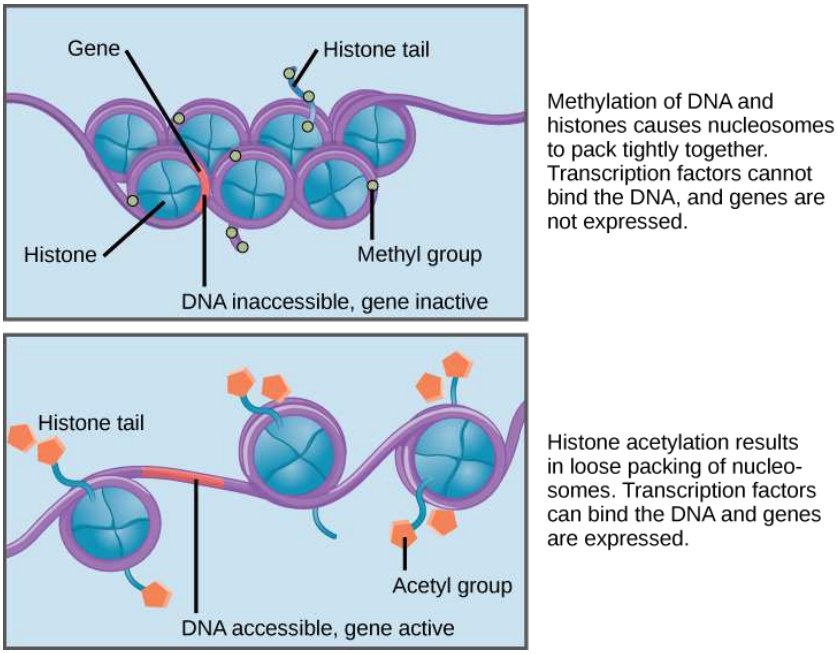


Figure 1.5. – DNA accessibility depending on histone modifications.

Kouzarides 2011. In most - but not all - cases, acetylation is associated to an increase in transcriptional activity, and vice-versa for methylation. These modifications are carried out by proteins histone acetyl transferases (HAT) such as EP300 (Ogryzko, Schiltz, Russanova, et al. 1996), or histone methytransferases and deacetylases for the corresponding modifications. They are complemented by pioneer factors whose role is to open the chromatin and then recruit further Transcriptional Regulators, such as what FOXA1 does for ESR1 (Ross-Innes, Stark, Teschendorff, et al. 2012).

As a consequence, the presence of certain histone marks correlates with the function of the genomic region on which it is bound. For example, the presence of H3K4me3 and H3K9ac characterizes active promoters (Liang, J. C. Y. Lin, V. Wei, et al. 2004). H3K27ac is associated to active enhancers (broadly speaking, as opposed to poised enhancers) (Creyghton, A. W. Cheng, Welstead, et al. 2010) and H3K36me3 is present in the gene bodies of actively transcribed genes (Teissandier and Bourc'his 2017). These associations are *not* absolute however, and those marks can be found on regions with other roles.

Histone variants There also exist histone variants differing from the canonical ones presented above by a few amino acids. In eukaryotes, centromers are defined by the presence of H3's centromeric variant (cenH3), although its function does not seem to differ from the canonical histone. In contrast, H2A possesses numerous studied variants (Bönisch and Hake 2012). The most common are H2A.X, implicated in DNA repair and H2A.Z contributing to transcriptional regulation. One may also

Histone residual	Modification	Modeling factor	Effect on transcription
H3K4	me3	SET7	Activation
H3K4	me1	ALL-1	Activation
H3K9	me3	SUV39H1	Repression
H3K14	ac	TAF1, EP300	Activation
H3K27	ac	EP300	Activation
H1K26	me	Ezh2	Repression
H3K27	me3	PRC2	Repression
H3K36	me3	SETD2	Activation

Table 1.1. – Selected list of histone modifications, including all those deemed necessary for a reference epigenome as of "Reference Epigenome Standards". In the code, *ac* and *me* respectively designate acetylation or methylation. The afferent number designates the number of functional groups, for example *me3* stands for trimethylation.

cite MacroH2A and H2A.B which contribute to X chromosome inactivation in female mammals, or the H2L family implicated in spermatogenesis.

DNA modifications The most studied DNA modification is the methylation of some of the cytosins in a CpG pair (*ie.* cytosines followed by a guanine). In humans, 70% of such cytosins are methylated, a modification effected by DNA methyltransferases. Regions of at least 200bp enriched in such CpG dinucleotides are named CpG islands. They are frequent in the upstream of promoter regions. The methylation of the cytosins in those islands is generally associated to an absence of transcription in humans (Baubec and Schübeler 2014), and is also observed in enhancers (Bae, J. Y. Kim, and Choi 2016).

1.1.1.2. Transcriptional complex

Transcription is the process by which a gene is expressed, producing premessenger RNA (pre-mRNA). During transcription, the RNA polymerase II (RNA Pol II) will bind on the DNA, open it, and transcribe the sense [1] strand into the pre-mRNA (Figure 1.6). Transcription begins by recruiting the Pol II in the Pre-Initiation-Complex (PIC), itself composed of Transcriptional Regulators. This recruitment can occur on the TATA-box region for those human promoters which possess it, or on the rare analog BRE recognition element. Otherwise, it happens on a region for which there is currently no known strong consensus properties.

1. Also called "coding" strand, but this can be a misnomer since it can be non-coding RNA or UTR regions.

. Figure 1.6 - Hahn 2004 and ©2012 Pearson Education

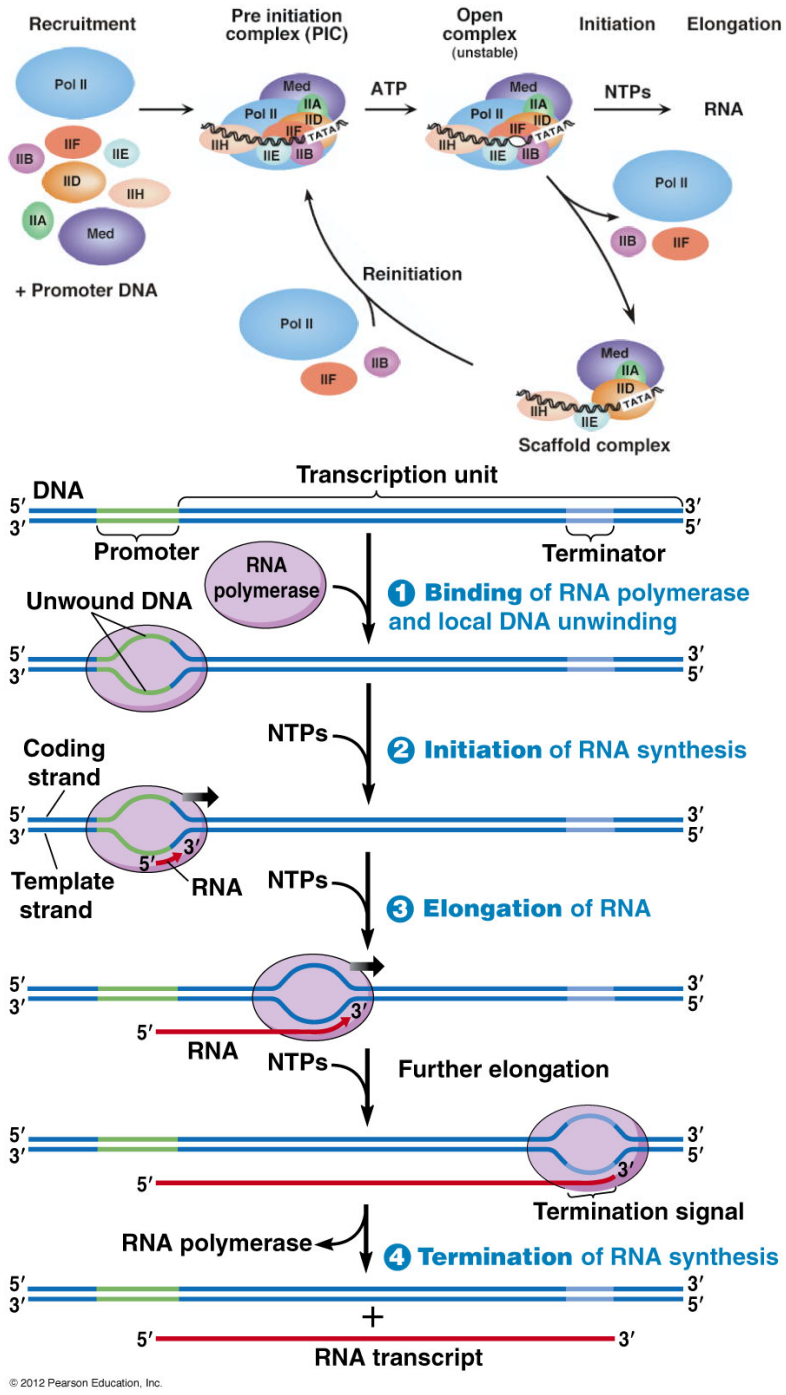


Figure 1.6. – Transcription process. The top figure presents the early recruitment of the PolII and the Pre-Initiation-Complex, which transitions into transcription. Initiation factors can remain in 5' as a scaffold complex. The bottom part of the figure presents the general timeline of transcription itself, where the coding region of the gene is transcribed into RNA until a termination signal is encountered.

RNA Pol II is the polymerase used in the transcription of all premessenger RNA, as well many small nuclear RNA and micro RNA. There also exist Pol I and Pol III, which are mostly responsible for the transcription of ribosomal RNA and transfer RNA respectively. Transcription initiation is characterized by the successive recruitment of the factors, or sub-units, composing the PIC. First, TFIID fixates on the BRE. Then, if it finds its binding sites, TFIIA comes stabilize the fixation of TFIID. This is followed by the recruitment of TFIIB, then TFIIF which recruits the Pol II itself. Finally, TFIIE and TFIIH have a protein kinase activity on the Pol II, which activates it.

The RNA Pol II itself is a complex protein of 550 Kilodaltons. In humans, it contains 12 sub-units numbered RPB1 to RPB12. Most noteworthy among them is RBP1, the largest, which forms along with RBP9 the groove into which DNA is transcribed. Furthermore, RPB2 maintains the contact between the chromatin and the RNA being synthesized.

The result of the transcription process is a molecule called a premessenger RNA. It is then matured, meaning the exons are removed, a polyA tail in its 3' end is added and methylated coif is added in 5'. Alternative splicing, meaning the removing of different exons, can result in different messenger RNA starting from the same premessenger RNA. These modifications will mostly serve to regulate the RNA molecule's post-transcriptional lifetime, and subsequent expression level. It is then exported in the cytoplasm for transcription into a protein.

The existence of the PIC as a complex of several factors is our first clue that the transcription process, and by extension gene expression, is regulated by more than a single actor and that combinations of regulators will be a crucial problem. This intuition is then solidified through the fact that the elements presented above are not the only constituents of the transcription initiation complex: there are also Transcriptional Regulators, whose binding is far less predictable and depends on other factors discussed below.

Transcriptional Regulators Transcriptional Regulators (TRs) are factors, usually proteins, that come to bind the transcription complex and influence its activity. This fuller complex is what will influence (facilitate, but sometimes hinder) the priming of RNA Pol II on the promoter and subsequent transcriptions. In humans, thousands of different regulators are known (Lambert, Jolma, Campitelli, et al. 2018).

TRs that can directly bind to DNA on regulatory regions are called Transcription Factors. They bind to the genome on sites known as Transcription Factor Binding Sites (TFBS) of around 6-12 basepairs. These are degenerate sequences and can vary for the same TF across the genome (Figure 1.7), hence a consensus binding motif is generally given with probabilities based on the sequences observed for experimentally verified bindings. Their mechanisms of action are discussed in Section 1.1.2.3 (p. 24).

As such, TFBS can appear or disappear due to mutations on the genome and general genomic plasticity (transposons, *etc.*), and as a result regulation modalities can

. Figure 1.7 - Ambrosini, Vorontsov, Penzar, et al. 2020

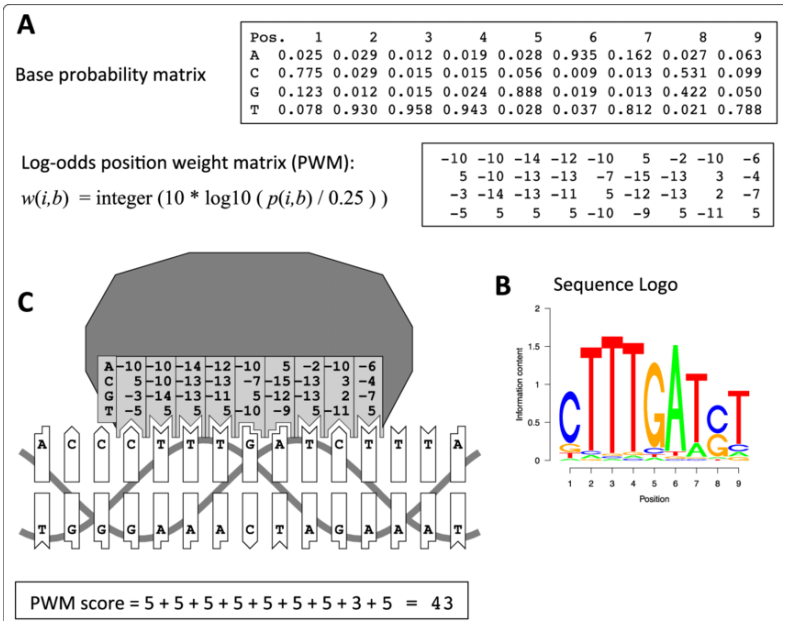


Figure 1.7. – Example of Position-Weight-Matrix calculation for a given Transcription Factor Binding Site (TFBS). The binding sites of TFs can be rather degenerate, so the consensus motif is calculated by averaging across binding sites verified experimentally.

vary between individuals. Furthermore, certain TRs such as YY1 can either act as transcriptional activators or repressors depending on the conditions (Verheul, Hijfte, Perenthaler, et al. 2020).

1.1.2. Genomic functional elements

In this section, we explore and distinguish two classes of genomic functional elements:

— *Genes*, which are transcribed into RNA.

— *Cis-Regulatory Elements (CREs)*, where the aforementioned Transcription Factors fixate.

The "Cis" epithet in CRE means they regulate genes on the same DNA molecule. This differs from trans-regulatory elements, which is another word for genes coding for Transcriptional Regulators that may fixate on another DNA molecule.

1.1.2.1. Genes

In the broadest possible sense, a gene is a nucleotide sequence coding for a molecule which has a function. In most cases, this molecule will be a messenger RNA produced through the transcription mechanisms outlined above. However, a gene may also

code for a RNA that is functional on its own. Furthermore, some genes have suffered mutations perturbing the coding of a functional protein and are called pseudogenes.

During transcription, the gene sequence on the template strand of DNA is read by the RNA polymerase in the 3' to 5' direction (or "reverse" direction). This results in the synthesis of a RNA which is a copy of the complementary transcribed strand from the 5' to 3' direction ("forward" direction). Directions are named after the position of the carbon atom on the ribose sugar which remains free (meaning it is not attached to another nucleotide) at the extremity of the DNA molecule. When discussing the position of genomic elements, "upstream" means towards 5', and "downstream" means towards 3'.

Genes are be present in either the (+) or (-) strand of DNA in relatively equal proportions, where in humans by convention the (+) strand is the one whose 5' extremity is closest to the centromer. 10 % of genes are overlapping with another gene. For more information on genes in general, see Kellis, Wold, Snyder, et al. 2014.

As seen in Figure 1.8, a gene contains several components. At each extremity of the transcribed sequence are UnTranslated Elements (5' UTR and 3' UTR) which mainly serve to regulate the expression of the subsequent RNA. The remaineder forms the Open Reading Frame, which contains both introns and exons. Introns can be removed from the RNA in a process known as splicing. A gene can code for several transcripts through the use of alternative promoters or alternative splicing of its exons.

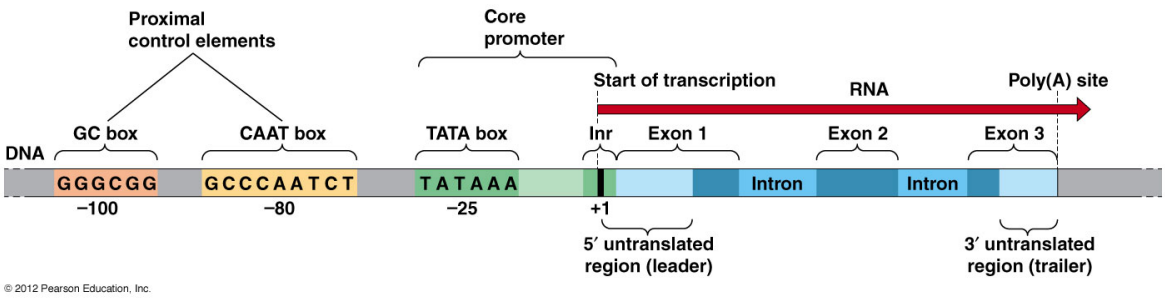


Figure 1.8. – General structure of an eukayorotic gene and associated promoter.

1.1.2.2. Promoters

Promoters are defined as the region on which the transcriptional initiation complex binds and RNA Pol II is recruited (Smale and Kadonaga 2003). Their structure is also presented in Figure 1.8. From 5' to 3', their conventional structure is as follows:

— A GC box followed by a CAAT box. These are proximal promoter elements, in opposition to the following core promoter. They fixate Transcription Factors.

— The TFIIB recognition element, followed by a TATA box. They play a role in transcription initiation as presented above.

. Figure 1.8 - ©Pearsson Education 2012

— The initiator is a degenerate region containing Transcription Start Site (TSS) or "+1" nucleotide, which as the name implies is the first transcribed nucleotide.
— The Motif Ten Element and the Downstream Promoter Element.

It should be noted that those elements are not constants. For example, the TATA box is only present in 24 % of human promoters (C. Yang, Bolotin, Jiang, et al. 2007) and can replaced by an analogue. There is strong degeneracy in the consensus sequences presented, much like there was for the TF Binding Sites in general.

This variability results in a sliding scale of promoter strength, from strong to weak. Strong promoters need little if any additional activation from other regulators to produce a strong active transcription, and are often present in front of constitutive genes. On the other hand, whether weak promoters are activated depends on their regulatory environment. Due to these discrepancies, a promoter is conventionally defined in bioinformatics as the region present for several kilobases upstream of a TSS (Shin, T. Liu, Manrai, et al. 2009).

Alternative promoters Each gene does not code for a single transcript. Alternative promoter usage, along with alternative splicing, is a source of transcript diversity. The two often work in tandem (Pal, Gupta, H. Kim, et al. 2011). In the human genome, between 30 and 50 percent of protein-coding genes possess multiple promoters whose differential activity creates transcriptomic diversity. And even inside a given promoter, the FANTOM-5 project has shown that most mammalian promoters are made of narrowly-separated TSS with cell-specific expression profiles (FANTOM Consortium and the RIKEN PMI and CLST (DGT), Forrest, Kawaji, et al. 2014a). Although the molecular mechanism of promoter selection remains unclear, it has been suggested that this regulation could come from promoter methylation status (Cheong, Yamada, Yamashita, et al. 2006).

There is a considerable difference in promoter usage between many cell types, development stages. However, it can also have deleterious consequences for the cell: this can cause developmental disorders (Pal, Gupta, H. Kim, et al. 2011), and promoters may play a role in the malignant transformation of cells and affect oncogenes. However, the role of alternative promoters remains unexplored in many cancer types as H3K4me3 profiles or CAGE-Tag are not readily available (Demircioğlu, Kindermans, Nandi, et al. 2018).

1.1.2.3. Enhancers and modulators

Enhancers are genomic cis-regulators upon which Transcription Factors can bind for the purpose of increasing the transcription level of a nearby gene. They tend to be short DNA regions with a length varying generally between 50 and 1500 bp (base pairs), but sometimes more (super-enhancers). Their range of action, upstream or downstream, can vary considerably from around 2 kilo bp to 2 million bp.

The currently accepted model to explain the effects of enhancer regions involves forming a DNA loop with the involved Transcriptional Regulators and the promoter

of the influenced gene (Figure 1.9). As we discussed before and as can be seen in the figure, the TRs involved can act by biding directly to the enhancer, to the transcription complex, or can bind to proteins in this larger regulatory complex which includes mediator TRs. See Kolovos, Knoch, Grosveld, et al. 2012 for more details on this DNA loop model.

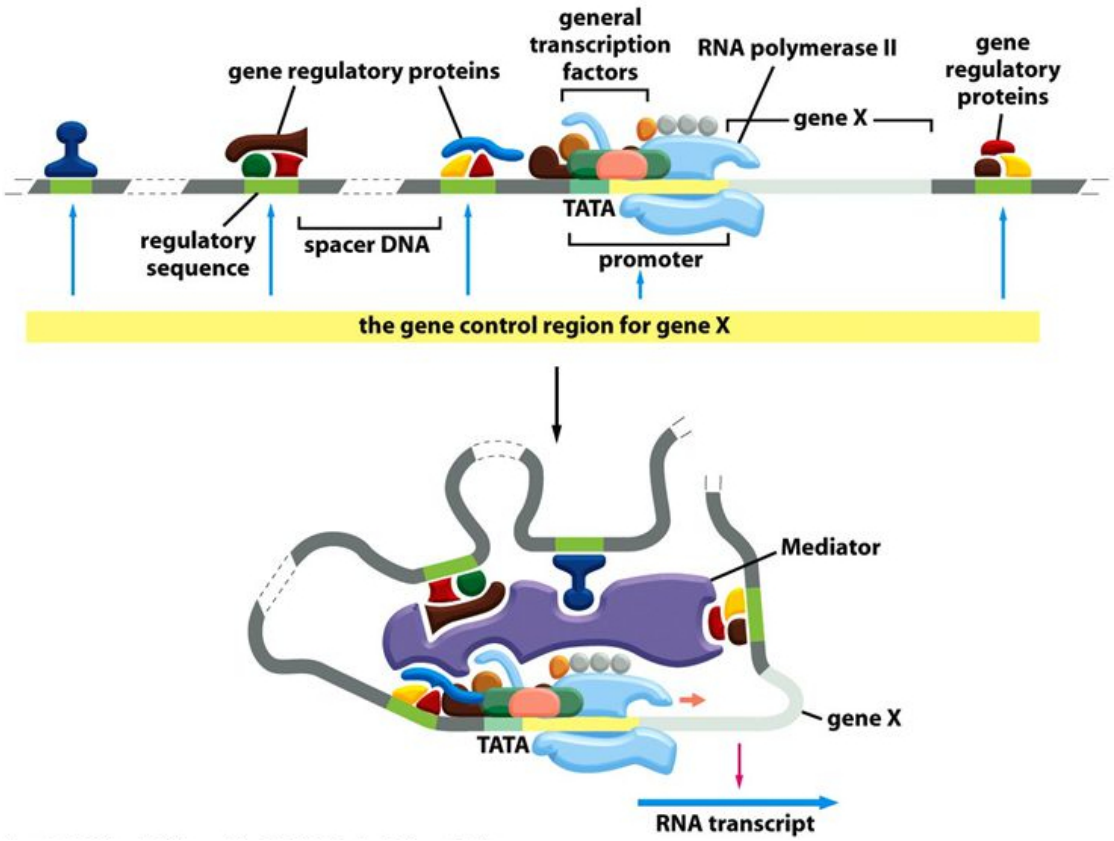


Figure 1.9. – General mechanism of cis-regulation. This involves the fixation of various Transcriptional Regulators on target Cis-Regulatory Elements, contributing to the formation of a DNA loop with the promoter of the gene to be regulated, and the formation of a larger cis-regulatory complex.

However, the activity level of enhancers is not constant between cell lines, nor is it constant in time or development stage (Long, Prescott, and Wysocka 2016). This is also true for the level of TF binding in general. Enhancers (nor silencers) have no consensus sequence or elements, which makes their *de novo* prediction a hard problem (Kleftogiannis, Kalnis, and Bajic 2016).

. Figure 1.9 - Figure 7-44 Molecular Biology of the Cell 5/e, ©Garland Science 2008

Silencers and overlap with other functions Silencers are the negative pendant of enhancers. They share most of their properties as presented here, with the difference that the TFs that they bind repress transcription instead of enhancing it (Della Rosa and Spivakov 2020). An enhancer can become a silencer later and vice versa (Kuwahara, Saito, Ogawa, et al. 2001 depending on the conditions.

Furthermore, some enhancers are transcribed, producing short eRNA that will not be matured. Their function is disputed (De Santa, Barozzi, Mietton, et al. 2010). Certain promoters can have an enhancer function for neighboring genes (L. T. M. Dao, Galindo-Albarrán, Castro-Mondragon, et al. 2017). Parenthetically, enhancers and silencers can be found spatially regrouped in Transcription factories to then influence several genes. (Rieder, Trajanoski, and McNally 2012)

Based on those facts, we can see that enhancers and promoters and cis-regulatory regions in general have many common points, and that the boundary between classes of CRE is nebulous.

1.1.2.4. Other elements

The list of CREs also includes insulators, which are barriers between the genomic domains situated in its upstream and its downstream. Other CREs tend not influence genes from which they are separated by an insulator. The mechanism is the formation of a DNA loop, physically blocking the interactions. The mechanism of TAD (Topologically Associated Domain, Figure 1.10) formation is not completely understood, but is is believed that it involves flanking a group of genes with two insulators. Loci (*ie.* genes, CRE, TRs) inside the TAD tend to interact much more than with outside the TAD. (Pombo and Dillon 2015), while genes situated inside a given TAD tend to share regulation and be active in the same context. It should be noted that TADs can be nested inside of other TADs to form regions of even more preferential interactions. In terms of the regulatory proteins involved, at least 2/3 of insulators bind CTCF.

There are also several other types of genomic functional elements that we should mention.

— Replication originals are the genomic locations where the replication complex originally fixates. There are between 30k and 100k of them in humans. They can form structures known as G-quadruplexes (Cayrou, Ballester, Peiffer, et al. 2015).
— Centromers are the region of contacts between the two chromatids of each chromosome. In humans, they are several Mbp long and full of repeated regions.
— Telomers are the extremity of each chromosomes, also forming G-quadruplexes. At each replication, roughly 50 bp are trimmed due to imperfections in the replication process. This trimming can be reversed by the telomerase enzyme, but only partially. The reason why this trimming is only partially reversed is unknown. A clue might be found in the fact that telomerase hyperactivity is known to play a role in cancer cell formation (Jafri, Ansari, Alqahtani, et al. 2016).

. Figure 1.10 - Adapted from Angg!ng - English Wikipedia and from Beagan and Phillips-Cremins 2020

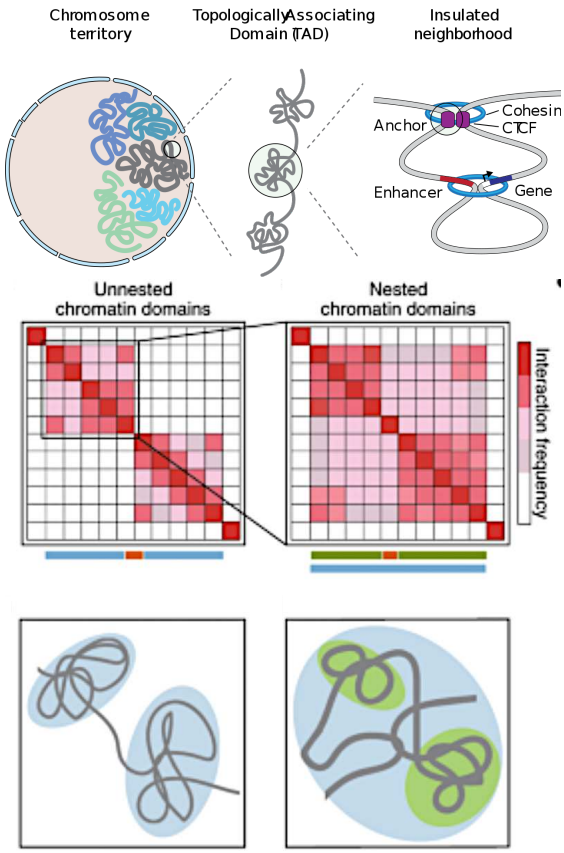


Figure 1.10. – Topologically Associated Domains (TADs) are flanked by insulators (top). Loci tend to interact preferentially with loci situated in the same TAD. This is visible when looking at interaction maps (bottom).

— Transposons are DNA sequences that can duplicate and move across the genome. Their role is currently poorly understood, with some speculating that they are plasticity and mutation factors in the genome (Bourque, Burns, Gehring, et al. 2018).

1.1.3. Combinations of epigenetic regulators

> The whole is greater than the sum of its parts.
>
> *Aristotle*

Having discuss the impact of individual regulators, we now discuss examples where $n \geq 2$ regulators act in cooperation to produce a different impact on the regulation. Of course, on a more abstract level it could be argues that since histone marks regulate

the chromatin state so that Transcriptional Regulator can be bound, this is in and of itself a type of *n*-wise cooperation between histones and TRs. I fully agree with this analysis, and this aspect is explored further later. Here, we focus specifically on molecular level cooperation.

1.1.3.1. Histone marks combinations

We discussed how the individual presence of certain histone marks correlated with the function of the genomic region on which they are bound, but this is also true for *combinations* of histone marks. Indeed, combinations of histone modifications can have antagonistic or synergistic effects (Strahl and Allis 2000) through cross-talk [2] (Barski, Cuddapah, Cui, et al. 2007).

This is illustrated by the fact that the presence or absence of certain combinations at a given locus is a good estimator of the state of the afferent chromatin (active enhancer, active promoter, etc.), better than individual marks. This is at the heart of ChromHMM (Ernst and Kellis 2012, Figure 1.11) which uses a Hidden Markov Model to partition chromatin in any desired number of states dependent on the combinations of histone marks observed. Each state is often found to be associated to a specific cis-regulatory region status (active promoter, inactive enhancer, *etc.*).

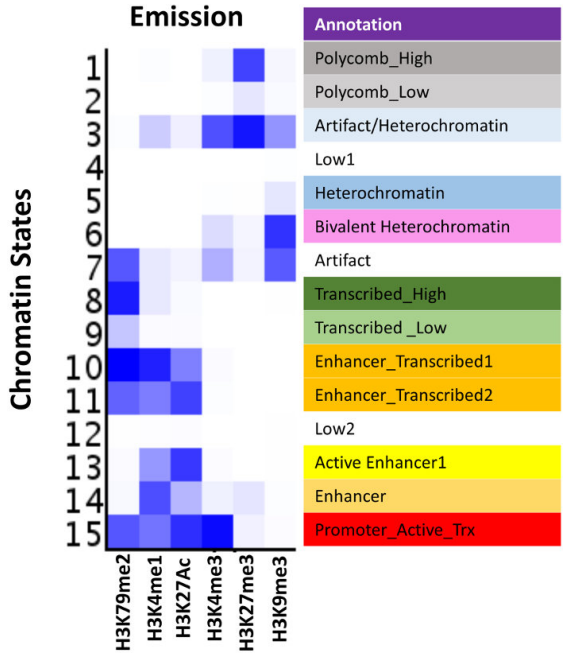


Figure 1.11. – Example model of segmentation of ChromHMM through histone status, based on melanoma tumor samples.

2. In this case, "cross-talk" is synonymous with reciprocal influence.
. Figure 1.11 - Terranova, M. Tang, Orouji, et al. 2018

This is further amplified by the phenomena of bivalent chromatin, defined segments of DNA with the presence of both an activator and a regulator at the same time. For example, a combination of H3K27me3 and H3K4me3 is associated with low-expression promoters.(Vastenhouw and Schier 2012). This illustrates how combinations of histones can have a different roles than the individual histones they are made of.

1.1.3.2. Transcriptional Regulator complexes

Most, if not all, Transcriptional Regulators do not influence genomic transcription all by their lonesome. Some of them possess activation domains upon which other Regulators, known as cofactors, can bind to them once they themselves are bound on the genome. This is exemplified by the Transcription Preinitiation Complex. This is also illustrated by all the regulatory complexes, in a broader sense, that we discussed when presenting the functions of enhancers.

Such complexes can perform a variety of functions, from remodeling the chromatin to stimulating the transcription itself. The modalities by which this is done are presented in Figure 1.12. Their operating range, as discussed, is around several thousand base pairs. In a broader sense, this cooperation can be temporally staggered such as with pionner factors fixating first to open the chromatin and which may or may not disassociate later, but the temporality aspect is beyond our scope. We mostly focus on complexes obtained by co-localization of TFs at any point in time, as provided by the snapshot given by the experimental data. Indeed, in most cases an observed colocalization of TRs is the result of a cooperation between them (Biggar and Crabtree 2001).

The impact of TR cooperation can be linear, with more factors resulting in a higher activation (Giorgetti, Siggers, Tiana, et al. 2010) or it can behave as an on-off switch for transcription (Chopra and Levine 2009). In any case, such combinations are responsible for the proper functioning of the regulatory regions described above.

Several relevant examples can be given. A classical one is the cooperation between CTCF and RAD21 to form the cohesin loop delimiting the TADs presented above (Stedman, Kang, S. Lin, et al. 2008). In an example that does not necessitate direct protein-protein interaction, FOXA1 is a pioneer factor permitting the later fixation of ESR1 (Ross-Innes, Stark, Teschendorff, et al. 2012), while FOXA1 is itself a downstream target of GATA3 (Kouros-Mehr, Slorach, Sternlicht, et al. 2006).

Some Transcriptional Regulators are known as master regulators, as they recruit and/or influence the activity of many other TRs across many different sites of the genome (The ENCODE Consortium 2012, J. Yang, Mani, Donaher, et al. 2004).

. Figure 1.12 - Spitz and Furlong 2012

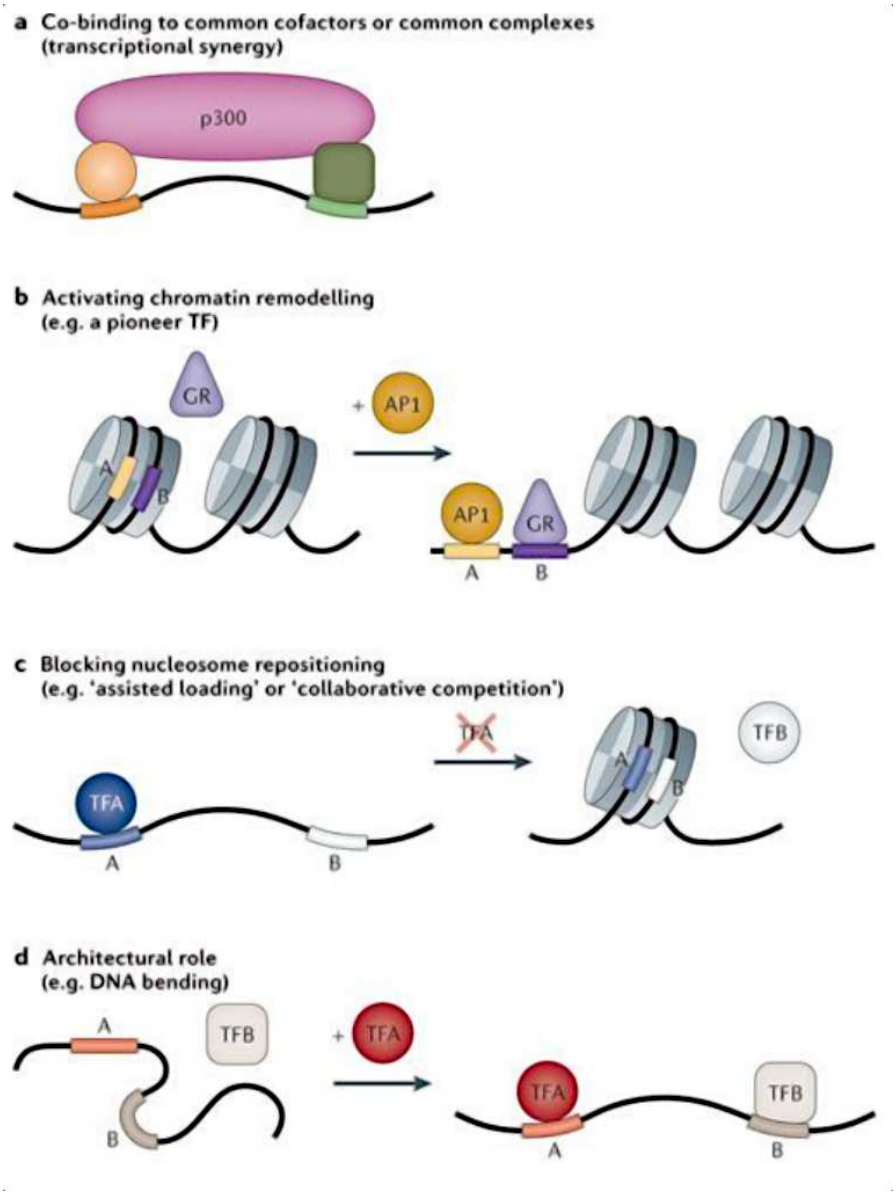


Figure 1.12. – Mechanisms of cooperation between Transcriptional Regulators. Possible modality of cooperation include: (a) cooperation for the recruitment of a cofactor, as in the case of EP300, and is the most common form of collaboration; (b) remodeling of the chromatin to facilitate the fixation of other factors, as pioneer factors do; (c) forcing chromatin decompaction by preventing nucleosome positioning; (d) inducing chromatin conformation changes to facilitiate the binding of other TFs.

1.2. Regulatory assays as data sources

In this section, we present the main genomic assays currently used to study the human cis-regulation mechanisms presented above. These assays allow for the localization of various regions of interest, such as epigenomic features and TF binding regions. Understanding them helps inform our decision about how to best represent this data for our analysis, as discussed in Section 1.4 (p. 52).

1.2.1. Gene sequencing

DNA, RNA, or more generally nucleotide sequencing is the process of determining the complete and precise succession of nucleotides in a given nucleid acid molecule. The first comprehensive method was designed by Frederick Sanger in 1977. However, but modern sequencing methods can sequence entire genomes in a matter of hours, for minimal cost.

From a practical standpoint, sequencing involves purifying DNA, performing library preparation, and processing the DNA with a sequencer. This unprecedented access to DNA sequences has become indispensable in biological research. The availability of comparatively cheap DNA sequencing combined with the rise of bioinformatics methods to treat such volumes of data has led DNA sequencing to become a cornerstone of modern biological research, being used in various applications such as, of course, determining the sequences of genes to identify mutations, but also in transcriptomic studies, phylogenetic studies, *etc.* The general principle of RNA-Seq is presented in Figure 1.13, but the steps starting from "High-througput sequencing" are universal.

The output of this process is called “reads”: the sequencer does not work on the entire DNA molecule at once, it produces short reads from the given sequence, which then have to be assembled to rebuild the complete original molecule. The length of such reads depends on the technology:

— “short-read” technologies (Illumina, ...) produced reads of around 26bp a decade ago, and modern ones produce reads hundreds of bases long. They work on sonicated DNA fragments.

— “long-read” technologies, on the other hand, are capable of producing reads of up to several kilobases (Nanopore, ...), usually by sequencing a single DNA molecule. These however often contain many errors, including insertions and deletions.

Techniques Ever since the Sanger method, many sequencing methods have been successively developed including pyrosequencing (454), semiconductor ionz (Ion Torrent) and ligation (SOLiD) that can today be considered obsolete compared to sequencing by synthesis (Illumina). In broad strokes, sequencing by synthesis consists of the following. After clonal amplification, a primer attaches to the forward string,

. Figure 1.13 - Thomas Shafee / English Wikipedia / CC BY-SA 3.0

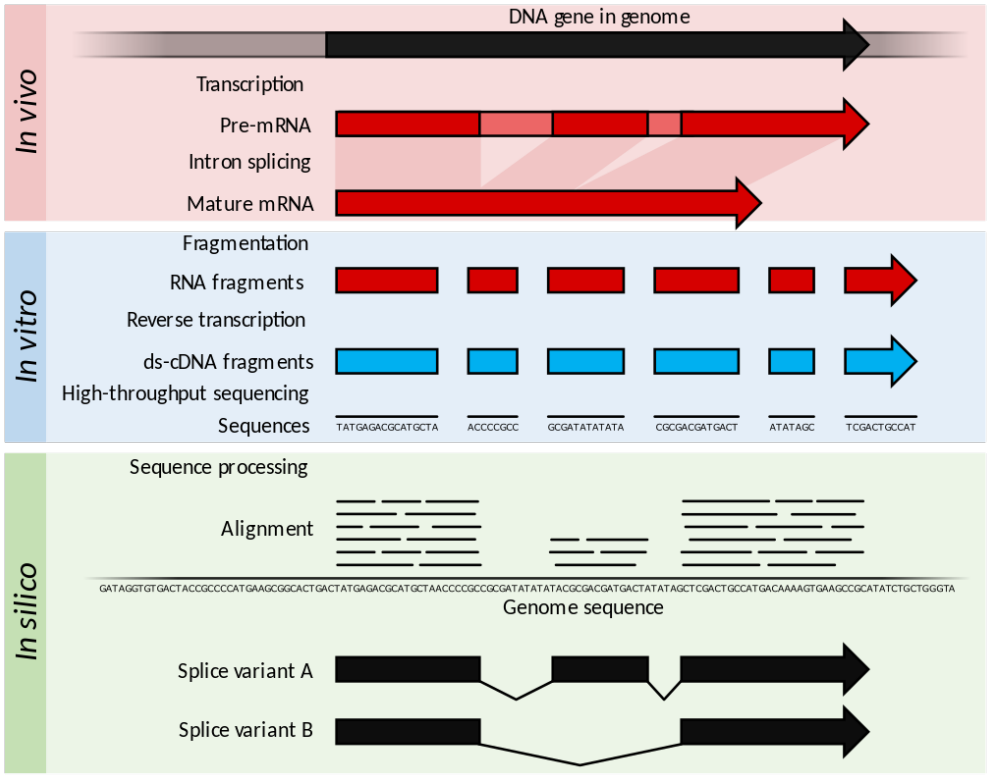


Figure 1.13. – Principle of RNA-Seq. The *in vivo* part corresponds the the transcription processes described previiously. In the *in vitro* part, the RNA or more generally the sequence is fragmented and passed to a sequencer. After the reads are produced comes the *in silico* part where the reads are aligned on a reference genome and studied for whatever purpose.

and a polymerase adds a fluorescently tagged dNTP nucleotide. Only one base is added per round, as the fluorophore is blocking. Each of the four possible nucleotides has a characteristic emission which is recorded by a computer. Then the fluorophore is washed away and the cycle is repeated.

While currently reigning supreme, it is expected that sequencing by synthesis will be brought to coexist with certain new specialized technologies for specific problems, such as the aforementioned single-molecule long reads (SMRT) approaches which have the advantage of not requiring an amplification step. A review of modern NGS (Next Generation Sequencing) methods has been performed by Besser, Carleton, Gerner-Smidt, et al. 2018. To summarize their conclusions, Illumina sequencing-by-synthesis approaches produce short reads of about 50 to 200 bp on average. This includes methods such as MiniSeq or MiSeq. More sophisticated Illumina methods such as NextSeq or NovaSeq have much larger throughput in terms of sequencing speed and are more precise, but have a steeper cost. In contrast, single molecule

seuqencing produces reads of up to 60-100kb (resp. Pacific Biosciences and Oxford Nanopore), but are very slow and have a high error rate.

Subsequent bioinformatic analysis Recall that the "reads" are short DNA sequences corresponding to the experimental fragments. As such, to be exploitable they must be mapped on a reference genome using an aligner, the most popular of which is Bowtie. For a recent review on aligners, see Schbath, Martin, Zytnicki, et al. 2012.

There are different approaches to sequencing that will require different protocols. A step that is usually constant is the need to run the reads through quality control such as FastQC and remove the primers/adapters for the approaches that use them. Sequencing can be single or paired end, meaning the DNA fragments were sequenced from either one extremity or both which can result in two shifted peaks in the signal depending on fragment length.

Whether the sequenced molecules were a cell's messenger RNA or genomic fragments from an experiment such as ChIP-Seq will, in turn, change the interpretation as we show below. For RNA-Seq, there is a quantification step where the sequenced transcripts must be assigned to their gene of origin using tools such as Cufflinks or KALYPSO. This quantification allows for an estimation of the expression level of a gene, but also the detection of new genes and alternate transcripts. If the experimental goal was instead to assemble a genome by sequencing it, overlapping reads need to be assembled in larger structures known as contigs.

For the sake of brevity, we focus on the pipelines for methods which are relevant to the identification of CRE and TR binding sites. This usually involves some flavor of peak detection in the sequencing signal, as explained later.

Single cell and bulk sequencing Traditionally, sequencing is done in "bulk", which means mixing genetic material from many cells of the biological sample. However, recent new approaches allow for the isolation of single cells, and amplification of the resulting small amounts of genetic material when the approach requires it. This allows for more precise analysis by taking, for example, a snapshot of a single cell in a tissue, or comparing transcriptomic profiles between cells at different differentiation stages in embryology.

However, the isolation of single cells remains experimentally challenging. On the *in silico* side, the small amount of genetic material results in a smaller amount of reads, so drop-out becomes a challenge. This is a situation where the data captures only a small fraction of the transcriptome of each cell combined in the stochasticity of gene expression resulting in certains genes or position of interest having zero corresponding reads (Qiu 2020).

Bulk sequencing remains less expensive, and is still relevant when the goal is to perform a global, cell-line wide profiling. These considerations aside, the treatments of single cell sequencing data is similar to bulk data in both the experimental and bioinformatic aspects.

1.2.2. Experimental methods for cis-regulatory annotation

This class of methods mostly concerns the detection of binding sites of chromatin regulatory factors such as Transcriptional Regulators and histones, as well as quantifying chromatin openness which is characteristic of actively transcribed regions.

1.2.2.1. ChIP-Seq

The most widely used approach in that regard is ChIP-Sequencing or ChIP-Seq. This approach combines chromatin immunoprecipitation (ChIP) with DNA sequencing to identify the binding sites of DNA-associated proteins.

The first step is enforcing a cross-linking between the protein to be studied and the genomic DNA. This is followed by chromatin fragmentation into fragments of, usually, about 500 bp. The third step is immunoprecipitation of the crosslinked DNA-protein complexes using an antibody against the protein of interest followed by incubation and precipitation. The penultimate step consists of DNA recovery, purification, and the addition of oligonucleotide adaptator to the stretches of DNA that were bound by the protein. This permits their parallel sequencing using the techniques discussed above. The final *in silico* step is to align the sequenced DNA fragments to a reference genome so as to identify the sites on said genome where the protein of interest is bound, for the specific biological context of the cell used in the experiment.

It should be noted that a ChIP-Seq experiment can and will also be used to identify the binding sites of cofactors, namely those Transcriptional Regulators which do not directly bind DNA, and will not distinguish between them. In the immortal words of the Apple II Reference Manual, "it's not a bug, it's a feature" as it allows us to estimate the binding sites of any other chromatin protein of interest.

Chromatin immunoprecipitation may instead be combined with PCR or another immunoprecipitation such as with ChIP-ChIP, the ancestor of the ChIP-Seq.

Peak calling After sequencing, one obtains a ChIP-Seq signal as a time-series (where "time" is the genomic position) corresponding to the number of reads (after normalization, usually RPKM [3]) present at each position on the genome for this experiment.

The goal is now to identify peaks in the signal. The basic assumption behind a ChIP-Seq experiment is that each such peak will correspond to a genomic region where the protein of interest (for this particular ChIP-Seq experiment) is indeed bound on the genome (or more accurately, as a chromatin complex) for the studied cell or cell line at this point in time.

The algorithmic means of peak calling vary depending on the tool, but they usually have in common that the true binding event is assumed to be located between the

. Figure 1.14 - Adapted from Park 2009

3. Reads Per Kilobase and per Million. A more robust normalization is used by methods such as DESeq, where one normalizes against the geometric mean per sample. This is applicable to all sequencing experiments.

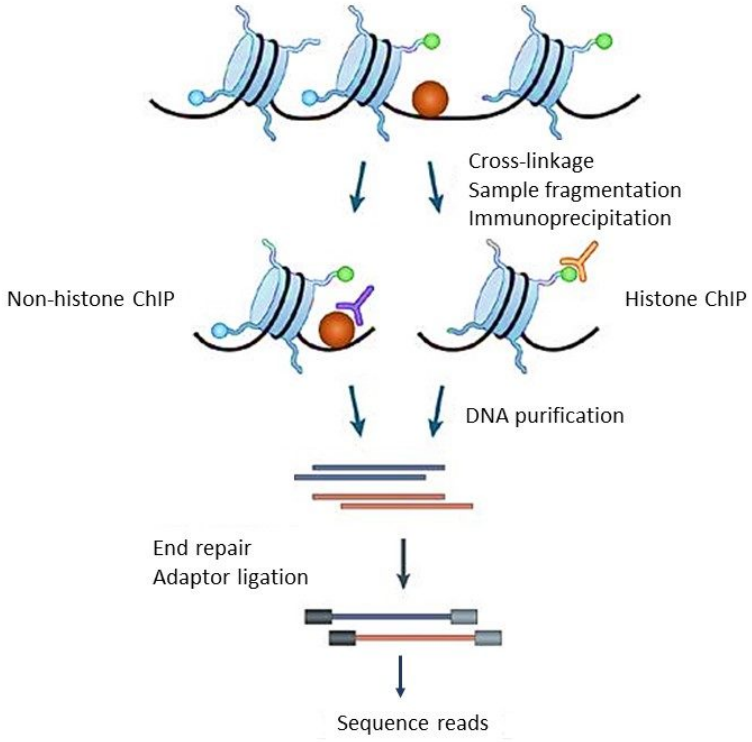


Figure 1.14. – ChIP-Seq process. This method involves cross-linking the protein of interest with the genome and sequencing the DNA fragments on which this complex is bound, to identify the binding sites of the protein of interest in this context.

peaks between the sense and antisense reads (Figure 1.15). Broadly speaking, this binding event is located either by shifting the fragments by half the length of the sonication fragment, finding the midpoint between sense and antisense peaks, or using a probabilistic method of the repartition of the reads depending on the true binding event (Mahony and Pugh 2015).

Once this process is complete, the mathematical object obtained is a list of genomic intervals corresponding to the putative regions on which the protein of interest is bound on the genome.

1.2.2.2. Other chromatin assays

However, ChIP-Seq is not the only method capable of assaying chromatin openness. Recently, other methods have been proposed and are presented in Figure 1.16. They follow the same general pattern of chromatin fragmentation, amplification of the desired regions, and sequencing followed by genome mapping. What differs is the

. Figure 1.15 - Bardet, Steinmann, Bafna, et al. 2013

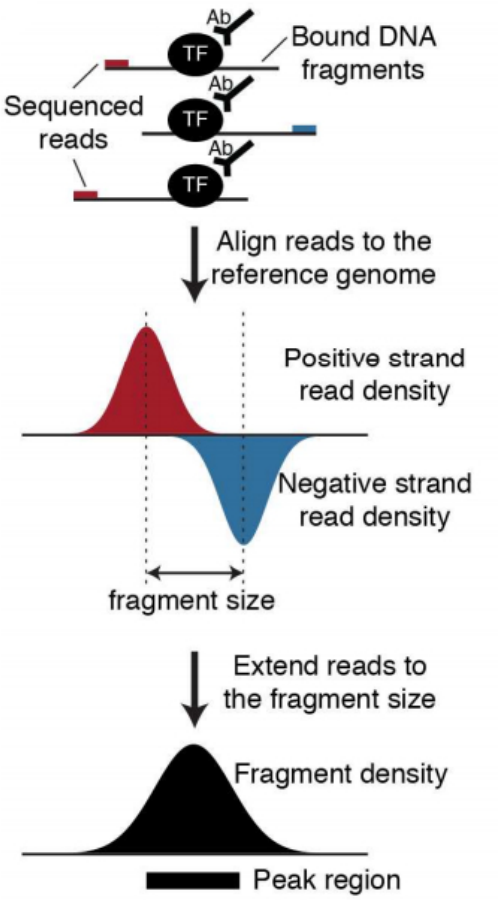


Figure 1.15. – Principle of ChIP-Seq peak calling.

criterion and thus the method used to select the region of interest. See Meyer and X. S. Liu 2014 for a recent review.

- In MNAse-Seq, chromatin is digested by a MNAse. The resulting fragments are all regions that were still bound by a nucleosome. Open chromatin regions area all regions where the signal was low.
- DNAse-Seq shows sites that are hypersensitive to DNAse I, which are open chromatin regions.
- FAIRE-Seq uses formaldehyde cross-linking to permanently bind nucleosomes to DNA. DNA that was not bound is then sequences, revealing open chromatin areas.
- ATAC-Seq works by marking open chromatin with hyperactive mutant Tn5 Transposase, inserting sequencing adapters into open regions of the genome.
- DAP-Seq works by hybridizing native DNA with special TFs that were tagged *in vitro*.

A major difference is that ChIP-Seq (thanks to the antibody) and DAP-seq are specific, while the others are not. But in the end, we get lists of regions.

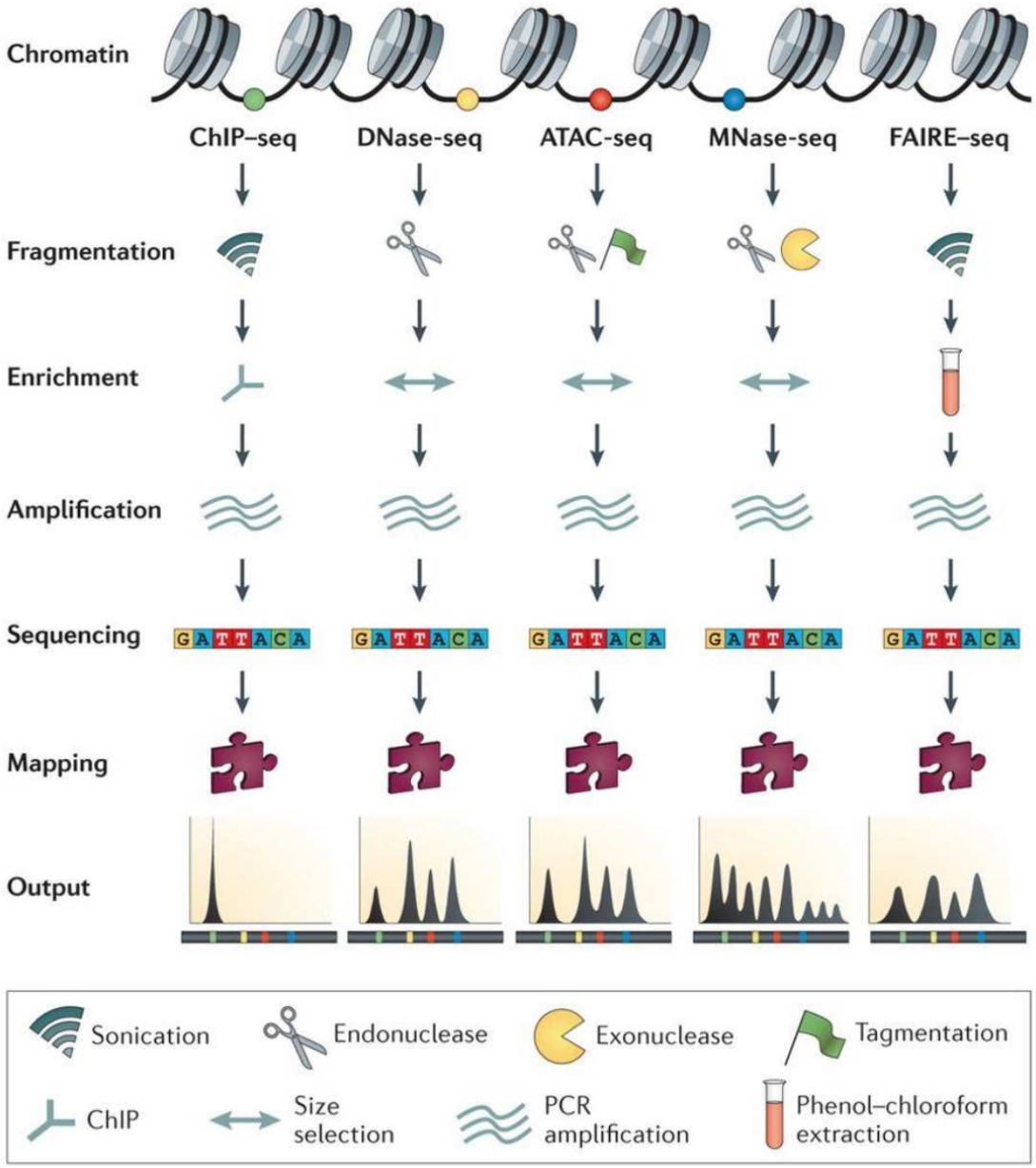


Figure 1.16. – Comparison of open chromatin assays. Most methods presented here are non-specific and instead assay chromatin openness. A description of each can be found in the afferent main text of the manuscript.

. Figure 1.16 - Meyer and X. S. Liu 2014

1.2.3. Other genomic assays

Another possibility to evaluate the function of a candidate Cis-Regulatory region is to use reporter assays. This generally consists of slotting the DNA sequence to be evaluated into a reporter genetic construct and evaluating the impact of the sequence on transcription. An example of this is the STARR-Seq approach for enhancer evaluation (Figure 1.17), which consists of slotting a candidate enhancer downstream of a strong promoter and quantifying how much it will amplify its own transcription (through sequencing). This was extended with Cap-STARR-Seq (Vanhille, Griffon, Maqbool, et al. 2015) with the capture of regions of interest.

Another possibility is adding a reporter gene that codes for a fluorescent protein downsteam of the promoter. The influence of the candidate regulatory region on transcription is proportional to the observed fluorescence. However, it should be noted that reporter assays may not properly quantity a region's role since the candidate regulatory region is isolated from its wider biological context. It could be missing required activators, or not be supposed to be activated as a general rule.

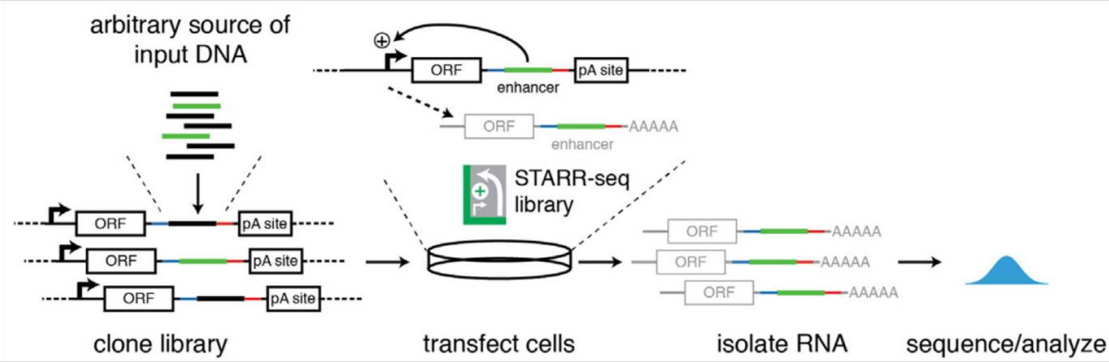


Figure 1.17. – Principle of STARR-Seq. The candidate regulatory region to be evaluated is transfected onto a clone library. The approach consists of evaluating its ability to stimulate (or repress) its own transcription in this construct.

Another assay that we should mention is chromatin contact assay through Hi-C approaches (Oluwadare, Highsmith, and J. Cheng 2019). It involves cross-linking close chromatin strands and sequencing the paired DNA sequences. Those paired DNA regions were regions that, *in vivo*, interact with each other.

For a more general review of enhancer assays, see Santiago-Algarra, L. T. Dao, Pradel, et al. 2017.

1.2.3.1. Purely *in silico* approaches

It is also possible to predict Cis-Regulatory Elements based only on their sequence through various purely *in silico* approaches. This includes analyzing their sequences

. Figure 1.17 - Muerdter, Boryń, and Arnold 2015

for known motifs, seeking homology with other known regulatory sequences in other species, or through chromosomal conformation. This is defined as an *ab initio* approach (literally "from nothing"), referring to the absence of any experimental assay. An example of tool suite to perform such analyzes is RSAT (Nguyen, Contreras-Moreira, Castro-Mondragon, et al. 2018).

This, however, is notoriously unreliable. For example, in practice only a fraction of predicted Transcription Factor Binding Sites translate to *actual* binding sites when experimental confirmation is sought out (Kaplan, X.-Y. Li, Sabo, et al. 2011). Indeed, as we discussed previously, CREs only possess weak consensus elements. Trying to infer them by analyzing which common patterns were a set of pre-selected CRE is very vulnerable to bias, where such an approach will identify CREs sharing characteristics with the training set. This set can be very narrowly focused or more generally of poor quality. As such, experimental approaches are usually preferred.

1.3. *Big data* in bioinformatics

Localizing various regions of interest (epigenomic features, TF binding regions, *etc.*) through the approaches presented in the previous section is now easier than ever, thanks to their decreasing cost. This has resulted in a wealth of experimental data from the broader scientific community, as well as from large consortia. As a result, bioinformatics has entered what is commonly called the *big data* era.

However, before proceeding, let us first define "big data". Recent attempts to ground the term based on a meta-analysis of academic articles (De Mauro, Greco, and Grimaldi 2016) use it to designate "data of such a high volume and variety that they necessitate different technologies and analytical methods to extract value from it". As far as I understand them, the only commonality in all these definitions is that the data was, at any one point in time, not saved as a spreadsheet.

That term is often abused by people who have somehow come under the delusion that useful information is correlated to weight, a notion of which the existence of Henry VIII ought to disabuse us. It is, to my mind, a symptom of the widespread misuse of the term that a meta-analysis of the literature would still amount to such a vague consensus. Hence, in this thesis, I wish to offer a different definition. I would point out that it is trivial to generate terabytes upon terabytes of white noise containing no meaningful information. Hence, volume is clearly not sufficient. As for variety, a similar argument can be made that generating *different* types of white noise will not increase the information quantity.

Indeed, an important but often forgotten assumption is that there should be *meaningful*data in the data: what actually matters is the information content (Shannon entropy). As such I would propose that "big data" be defined as data whose information content is so high that many higher-order underlying rules between its variables are present. This definition is expanded upon in later sections.

Similarly when it comes to variety, to me big data designates a large number of

datasets with links between each other. This is related to the broader field of multi-view learning. In this manuscript in particular, "variety" can mean different datasets are available covering many Transcriptional Regulators, or regions of interest in general. But it may also mean those datasets are corroborations for a given regulator, are assays for two regulators that are usually correlated, *etc.* or any combination of the previous propositions.

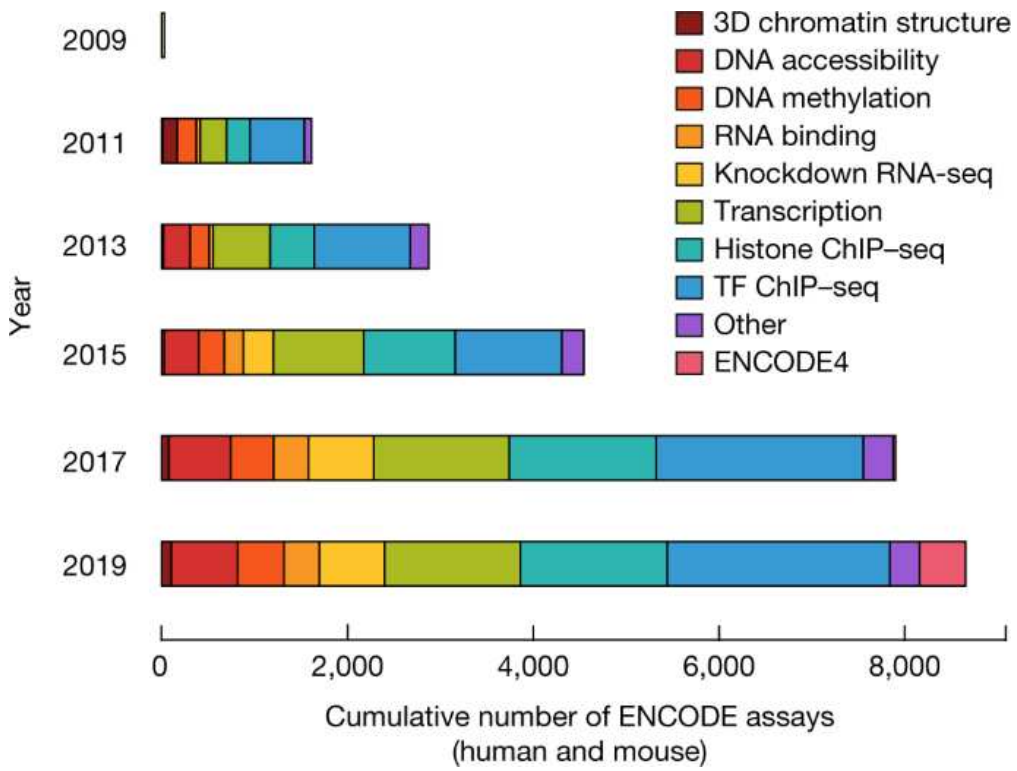


Figure 1.18. – In the ENCODE database, the cumulative number of assays has massively increased over the last decade, with a large variety as well.

In this section, we briefly discuss the material challenges posed by the sheer volume of biological data, however the potential use of big data for corroboration is more of interest. It is undeniable that more bioinformatical data is available than ever before, and is being created and stored faster and faster (Figure 1.18).

This is taxing in terms of computing power required to store and process this data, but also exerts an intellectual pressure on the flesh-and-blood humans performing the analysis (Muir, S. Li, Lou, et al. 2016). This also results in more complicated pipelines, containing a large number of varied methods requiring different processing workflows, if only because technological advancements results in the continuous introduction of new methods. Indeed, as of writing there are at least 534 known protocols for ChIP-Seq alone (Clément, Emeric, J, et al. 2018). Furthermore, leveraging the potential

. Figure 1.18 - Snyder, Gingeras, Moore, et al. 2020

cross-correlation between different datasets also has a cost since one is now expected to compare and re-analyse their data with other available data from different sources, but also of different types.

1.3.1. Genomic databases

In the last decades, there have been initiatives dedicated to making the aforementioned bioinformatical data accessible to the wider public. Since the cost of performing large scale assays for a variety of regulators can remain high, consortia have formed to absorb these costs ever since the Human Genome Project. The end goal is to centralize data and help the scientific community study genomic regulation. In this section, we present some of these efforts.

1.3.1.1. Sequence and genome archiving

The International Nucleotide Sequence Database Collaboration is an initiative between the *European Bioinformatics Institute* (EBI, European Union), the *National Center for Biotechnology Information* (NCBI, USA) and the *DNA Data Band of Japan* (DDBJ, Japan) aiming to offer an archive of the raw data and metadata from high-throughput genomic sequencing experiments, accessible through archives such as SRA.

The genome assemblies themselves are handled by the Genome Reference Consortium. The current assembly of the human genome, *GRCh38*, was released in December 2013 and has since seen patches for the correction of assembly errors (short read problems) and the addition of alternative haplotypes.

1.3.1.2. Cis-regulatory element annotation

ENCODE Consortium As of writing, the most prominent consortium regrouping data for Cis-Regulatory Element annotation is ENCODE *(Encyclopedia of DNA elements)*. It was created in 2003, with the stated aim of regrouping and reprocessing data by subjecting them to a normalized quality control protocol to help study genomic regulation. Their goal is to form a comprehensive encyclopedia of TF Binding Sites, histone marks, and more generally study the chromatin markers we presented in section 1.1.

The project was made of 4 phases, the first of which (pilot phase) ran up to 2007 to identify the most promising methods. Indeed, ENCODE was responsible for the development of many bioinformatical tools and methods (among others: Shen, Myers, Hughes, et al. 2016; M. Teng, Love, Davis, et al. 2016; Q. Li, Brown, H. Huang, et al. 2011).

In terms of output, the ENCODE catalogue ENCODE gives a list of 1.3M putative CRE for 600 cell types (as presented in ther main paper The ENCODE Consortium 2012) based on their centralized data. One could also mention GENCODE, which

is a sub-project to identify and classify all genes. Its annotations mostly come from Ensembl, another sub-project.

The key point is that the data, which mostly consists of the sources (ChIP-Seq, etc.) described above, is available online [4].

FANTOM The FANTOM (Functional Annotation of the Mammalian Genome) project focuses on the study of the transcriptome. In its third phase, FANTOM developed the CAGE method (Takahashi, Kato, Murata, et al. 2012) to study transcription initiation and promoters, by focusing on the 5' extremity of the mature mRNA.

The most recent phases of the project focus on the study of alternative promoters (5) and long non-coding RNA (6). It recently produced the largest collection to date of annotated promoters and TSS in human and mice (FANTOM Consortium and the RIKEN PMI and CLST (DGT), Forrest, Kawaji, et al. 2014b).

International Epigenome Consortium ENCODE is not the only group focusing on epigenomic annotation. In 2010, the *International Epigenome Consortium* was created to regroup those, including for example the BLUEPRINT project (European Union). Their goal is to produce reference epigenome for a variety of species, by regrouping RNA-Seq and chromatin assays (ChIP-Seq, DNAse-seq, etc.) for a variety of key chromatin markers, such as histone marks (Kundaje, Meuleman, Ernst, et al. 2015).

ReMap Faced with this abundance of data, other efforts have sprung to standardize it. One may cite ReMap (Chèneby, Ménétrier, Mestdagh, et al. 2020), which endeavors to identify and characterize regulatory regions from a large-scale integrative analysis of DNA-binding protein experiments. The 2018 human update was used in certain projects in this manuscript as part of a collaboration with Jeanne Chèneby. It is made of uniformly annotated and processed 3,180 ChIP-seq experiments, including some biological replicas, in a variety of cell types and tissues. This data also contributed to the JASPAR project, which aims to identify consensus binding sites for many Transcription Factors and gives across the genome the sites matching those consensus motifs, along with the score of the match Fornes, Castro-Mondragon, Khan, et al. 2020.

ReMap's initial curation and uniformized reprocessing workflow provide sufficient quantity and quality to use the approaches proposed in this manuscript.

1.3.1.3. Others

Other initiatives can be mentioned here. For the publication of a scientific paper, data should be made accessible. Archivage of experimental data for microarray and NGS is proposed by GEO Omnibus (NCBI) and ArrayExpress (EBI). The raw sequencing

4. It is available as UCSC Genome Browser tracks - `https://genome.ucsc.edu/`

data can be stored at SRA and ENA. Unfortunately, unlike projects such as ENCODE, annotation and processing is *not* uniform.

Speaking of uniformity, Gene Ontology (GO) developed a nomenclature applicable to all eukaryotes to describe gene functions, as an acyclic graph, an approach also used by KEGG.

1.3.2. Integrative analysis of multiple data views

> Quis custodiet ipsos custodes?
> *Who watches the watchers?*
>
> Juvenal

The availability of such volume and variety of data lends itself to integrative analysis, by combining informations provided by different datasets. This can means two different things. Firstly, leveraging the existences of replicates and experimental confirmations, which is the impetus behind ReMap for example. Such a meta-analysis of datasets is emerging, notably in ChIP-Seq (D. Chikina and G. Troyanskaya 2012). Secondly, it can also mean exploiting the relations between the regulators assayed by different assays.

Multi-omics The second philosophy is at the root of the field of multi-omics. A general review can be found at Subramanian, Verma, S. Kumar, et al. 2020. In a general sense, this designates leveraging different kings of assays (transcriptome, epigenome, methybolome, etc.) to get a more complete picture of a biological situation. This is related to the field of multi-view learning, if we consider each assay to be a view. This entails combining information of different natures, such as the facts "presence of EP300 at this position" with "histone acetylation at this position" or "this or that gene is expressed". More specifically, a general review of multi-view learning in biology can be found at Y. Li, Wu, and Ngom 2016. This is also an active area of fundamental research (for example, see Cao, H. Zhou, G. Li, et al. 2016, modeling interactions between views as a tensor that will be factorized). In this thesis, "combinations" can also be extended to this particular kind, between datasets of different natures. We present such examples.

Many multi-omics approach use Machine Learning methods as we present in section 1.4.2 (p. 57). In particular, matrix factorization methods as presented in section 4.4.2.1 (p. 165) are very popular. This often entails custom factorisations of the matrix representing each dataset, with an joint optimisation objective seeking to both explain the original dataset as well as another one, to find commonalities. Other fusion approaches are detailed in section 1.4.1.2 (p. 54).

One can also cite network-based methods, studying the networks formed by different data views. Bayesian methods are also used. Finally, multiple kernel learning is very popular, which is very reminiscent of middle fusion approaches, in that the

objective is to learn a combination of kernel functions, based on the different views (S. Huang, Chaudhary, and Garmire 2017).

1.3.3. Noise

> Nullius in verba.
> *Take nobody's word for it.*
>
> Horace

"Noise" is not just a derogatory term ~~older~~ more respectable generations can use to deride whatever the musical genre *du jour* is. It has precise implications. The Oxford dictionary defines noise in experimental sciences as "any random fluctuations of data that hinders perception of a signal". Noise can be white noise, but the term is sometimes used to refer to any biais in the data, and this second prong is also important to us. Here, we define noise as this:

Definition 1. *Consider a logical proposition X about a biological reality, and consider our experimental observation $\hat{X}$ of that reality. Noise is any factor that causes $\hat{X} \neq X$.*

This definition is rather sweeping, rejoining the conventional definition of *anomaly* (more on that later, in Section 1.4.3, p. 60). I chose to include under this umbrella term of noise all concerns such as experimental bias and general inaccuracies of the experimental methods.

The biological assays described in Section 1.2 (p. 31), like many experimental methods, have sources of noise. In this section we describe several sources of noise and methods to deal with them, but we should keep in mind that they all tend to the same end result: incorrect observation of the biological reality. Some of these problems can be uniformly random (white noise), other such as a bias due to the experimenters themselves can be more systematic. This is why, in this thesis, we focus on whether the usual combinations are respected. More details in Section 1.4.3.

In this section, we list some sources of noise in no particular order. What is important is that there is variety and not necessarily order to this noise, and correcting each specifically would require a different kind of supervision and/or model for each. We explore unsupervised ways to correct it in this manuscript. This prefigures the use of autoencoders and matrix factorizations in the approaches presented in this thesis, as they are known to be resistant to noise.

Mathematical modeling of noise Noise can also be modeled mathematically. This is usually expressed through a relation such as $D = X + N$, where D is the observed data, X is the noiseless data, and N is a random variable with certain characteristics representing the noise. For example, a popular choice for N is the use of a Gaussian noise, meaning the values that the noise can take on are Gaussian-distributed.

However, there exist a multitude of other models. In fact, pretty much any kind of random variable can be used in a noise modeling such as this. It is relevant when

certain characteristics of the noise are known, so the model can be as close as possible to the real noise: indeed, the noise then becomes an element of the system's modelisation. In statistics and regressions, the noise is sometimes equated to the fraction of the variance in the target Y that cannot be explained by the known variables X. In effect, it becomes a catch-all for *effects of unknown causes*.

Indeed, when generating artificial CRE representations (see section 3.4.1) we have modeled false positive peaks as an addition to the noiseless data, using random variables to detemine their characteristics such as their position. The RV used do not respect the usual correlation between sources, by design, as the real noise would.

In this thesis, since our goal was often to remove noise in an unsupervised manner by more broadly removing elements that do not respect the usual combinations, no specific modeling of the noise was used. That being said, those modelings are nevertheless related to the weak matrix factorizations we used (see section 4.4.2.1), where an error term is permitted, resulting in an approximate reconstruction of the original matrix. This error term is often equated to this N random variable.

1.3.3.1. Noise in ChIP-Seq

Let us begin with ChIP-Seq. There are many potential sources of noise (as defined above) in such experiments, presented here in no particular order.

The main source of noise in ChIP-Seq is immunoprecipitation quality. Antibodies can be insufficiently specific (Kidder, G. Hu, and Zhao 2011) and non-specifically bind to other proteins. Furthermore, antibodies may have different affinities for target proteins, creating a bias in intensity between assay for different regulators. The second main source of noise comes from the library preparation and other sequencing biases, mostly in the form of uneven genomic sonication. These result in an abundance of spurious sites J. Xu, Kudron, Victorsen, et al. 2019.

Furthermore, the human genome possesses *blacklisted* regions, known to cause problems in many genomic assays by being often the site of an artifactual unwanted signal, which can be due to their high frequency of repeated short sequences or anomalous antibody fixation (Amemiya, Kundaje, and Boyle 2019). Note that there are very few of them in the latest human genome (*hg38*) assembly.

Inadequate experimental controls (such as the absence of *background*, see next paragraph) can complicate peak calling, along with other factors (Wilbanks and Facciotti 2010). False positives can be introduced for biological reasons, such as on active promoters (Jain, Baldi, Zabel, et al. 2015) and highly expressed loci (Teytelman, Thurtle, Rine, et al. 2013). This is compounded by all errors that can arise in the sequencing itself (read quality).

Besides errors, anomalous peaks can be caused by other biological and technical specificities (eg. different protein fixation kinetics), systematic experimentator biases, mutations creating new TFBS, TRs having rare secondary roles, etc. To top it all off, the peak caller itself is a source of error peak callers (False Discovery Rate of 1-5 % or more, Chitpin, Awdeh, and Perkins 2018).

Finally, ChIP-seq peaks tend to be much larger than the actual binding site of the Transcriptional Regulator, which is only a few base pairs long. This can be partially alleviated by trying to find the peak summit, which is assumed to correspond to the binding event itself.

Fighting noise So, how can this propensity for noise be countered? We specifically discuss anomaly detection later, but let us present here the methods used by ENCODE to counter this bias, as presented in Landt, Marinov, Kundaje, et al. 2012.

Firstly, ChIP-Seq peak calling is usually done against a *background*. This can take two forms. The first is to compare the experimental signal to another run done with a non-specific antibody like IgG, which is called a *mock IP*[5]. This should correct both library and sequencing biases as well as biases due to antibody fixations. Another possibility is to simply sequence against an *input DNA* made by sequencing Whole Cell Extracts (see below), which theoretically should correct only the library and sequencing biases. However in practice, IgG and input DNA perform similarly[6], and today input DNA is used almost exclusively (J. Xu, Kudron, Victorsen, et al. 2019).

Tangentially, the background can be done with other antibodies than IgG to get the difference in fixation relative to them; and one may even use another ChIP-Seq experiment's signal as background, but this is considered a very unreliable expedient at best. Of course, even such a control is not foolproof and errors may still remain. Regardless, the majority of ChIP-Seq experiments publicly available have not used any kind of intput, which means those errors were not corrected[7].

At the experimental level, the proportion of mapped reads (how many of the reads were sucessfully mapped to the genome) is important, as a low proportion could indicate sequencing errors. Conversely, the number of regions mapped by unique reads should remain low[8]. Keep in mind that short reads can map on several genomic regions due to random chance. Alongside FASTQC sequencing quality metrics, these metrics give information about the quality of the sequencing and of the experiment in general. This is further assayed by cross-correlation between strands through the Normalized Strand Cross-correlation coefficient (NSC) and Relative Strand Cross-correlation coefficient (RSC) ensuring the data on both DNA strands matches the other.

Then, for the peak themselves, the total Fraction of Reads in Peaks (FRiP) is computed. For a sufficiently specific antibody, one would expect that most read be found inside of enriched regions designated as peaks (Landt, Marinov, Kundaje, et al. 2012). Furthermore, it is expected that peaks called in two experimental replica will be very

5. This stands for "mock ImmunoPrecipitation".

6. With an ever-so-slight advantage for IgG controls, which means that in an ideal world both would be performed.

7. Is this realization as horrifying for you as it was for me?

8. If a region is mapped by only one read, it is possible it was not part of the DNA sample that was sequenced but instead that there was a read mapped here by mistake (likely because the read contains an error). This is much less likely if many reads indeed do map there.

similar: the Irreproducible Discovery Rate measures the consistency between peaks called for two replica of the bio condition.

Having presented the best-practice methods used by ENCODE, a pattern emerges. I would emphasize that ENCODE indeed recommends *cross-validation in a combination of datasets* to weed out this noise, instead of a tedious, error-by-error specific correction that might introduce new bias. This philosophy is at the heart of this thesis manuscript.

1.3.3.2. Noise in other approaches

There are also noise sources in other approaches, and some that are common to many NGS approaches (ChIP-seq included), such as the ones related to sequencing.

Genomic assays may not always represent biological reality. A detected binding site for a Transcription Factor on a given region does not necessarily mean it is a Cis-Regulatory Element: this may be due to biases inherent in the approach, or could simply be a binding site that is not used due to other regulatory mechanisms superseding it (Cawley, Bekiranov, Ng, et al. 2004) [9]. A solution to this problem is to use more precise/specific experimental assays to determine if the region is truly active. This is part of the missions of ENCODE and FANTOM.

The sequencing itself can be a source of noise. When aligning reads to the genome, low complexity regions complicate mapping and can have many reads falsely assigned to them (H. Li 2014), Relatedly, the PCR itself introduces biases for certain regions (Aird, Ross, W.-S. Chen, et al. 2011); broadly speaking, loci with extreme base compositions (CG-rich mostly but also AT-rich to a degree) can are often under-represented. This can be alleviated through adaptations of the PCR protocol, although such factors (temperature, ...) can introduce their own biases in turn. Finally, deletions, insertions and other mutations on the molecule being sequenced compared to the reference genome, as well as simply errors in the sequenced reads, can result in spurious mapping.

The biases of the previous paragraph can be identified through a control experiment, by sequencing a cell's full raw DNA [10]. This is especially valued in ChIP-Seq experiments (Whole Cell Extract control, aka. input DNA, as discussed above), and is also applicable to all sequencing experiments but is not considered very necessary.

When it comes to other assays, FAIRE-Seq is known for high background noise and is less sensitive as a result (Tsompana and Buck 2014). DNAse-Seq requires more genetic material and needs rigorous calibrating of conditions and fragments (H. H. He, Meyer, S. S. Hu, et al. 2014. DAP-seq has a very high failure rate (Bartlett, O'Malley, S.-s. C. Huang, et al. 2017). Finally, as discussed, the in silico methods of predicting region activity based on sequences have high false positives.

9. This is less likely if known collaborators of that regulator are present.
10. Or the full whatever-it-is that you are sequencing.

1.3.4. Data analysis pipelines and workflows

The use of such volumes of data has also brought more prosaic concerns about the resources needed to process them and reach reproducible results.

1.3.4.1. Computing resources management

The most obvious consequence of processing large amounts of data is the corresponding need for more computing resources. As such, today a large scale bioinformatic analysis will usually not be run on a single computer, but on supercalculators [11]. This is known as High Performance Computing. Although they, as would be expected, have larger memory and computing power, there is more to a supercomputer than this. They usually run on a different, parallelizable architecture which requires a different approach to be fully exploited.

In broad strokes, a supercalculator is generally implemented as a *cluster* of computers, each called a *node*. Each node is itself composed of several CPU cores and a RAM pool. A key notion is that intra-node communication, between cores, is relatively easy thanks to a shared RAM pool, and most algorithms nowadays leverage this.

Communication between nodes however, is another can of worms entirely, as they do *not* share a RAM pool. There are protocols allowing synchronization between nodes (or cores, or threads), such as the Message Passing Interface [12]. The general principle is to send messages that the threads can then interpret autonomously in their code, without accessing the other threads or their memory spaces. In the majority of cases however, the nodes will run independent tasks. For example, Keras permits GPU parallelization in deep learning [13], but this is done by averaging the gradients computed by each node afterwards. This example shows how node parallelization is possible only when the task can be *split* in independent chunks.

This has two consequences:

— Tasks are usually broken into atomic units that can be run on nodes independently, *without knowledge about the others* beyond their completion status.

— A master node is required to process and distribute those tasks.

I would note that communication between cores of a same node poses similar computing challenges on the programming level, despite their shared RAM pool. Indeed, simultaneous access of the same memory address by different cores is a common source of errors, necessitating thread-safety and synchronization protocols not unlike those used to synchronize operations across different nodes.

As a supercomputer is generally a shared resource, they are equipped with software managing the task distribution called a job scheduler. Common ones includes

11. Here, I use "supercalculator" in the broadest possible sense of "a computer more powerful than a general-use computer or laptop".

12. It can be hybridized with other protocols, for example using OpenMP for parallelism within a (multi-core) node while MPI is used for parallelism between nodes.

13. Each GPU is analogous to a node, since it has its own core and especially its own RAM pool that it does not share with other GPUs.

TORQUE (qsub) and SLURM. When submitted, a computing task (ie. a job) is added to a queue, and allocated computing resources when they are available. How to allocate the resources to the users is an administrator decision, but is commonly handled through a karma system tracking resource usage and good behavior for each user.

Other large scale solutions include Apache Spark. Based on the HADOOP file system, it is an API designed to apply operations to datasets scattered across different computing nodes as if it was on a single computer, with the Core API providing dispatching functionality for operations such as map, filter and reduce, as well as scheduling for task distribution. Other parts of the API provide algorithmitic and Machine Learning functionalities.

Algorithmic scaling Any given algorithm, such as one might conceivably run on a supercomputer, has a time complexity. The commonly considered one is the worst-case complexity, giving the maximum number of elementary operations performed by the algorithm as a function of the size of the input data (in bits). One commonly focuses on the asymptotic behavior of the complexity, expressed using a big O notation. For example, a complexity of $O(n)$ means the time cost of the algorithm scales linearly with the size of the input. As the behavior is asymptotic, an algorithm that always adds 2 elementary steps for each bit on input data has the same complexity that one that adds 5: in both cases, the number of steps is linear with the input data size. However, an algorithm that takes 9 steps for an input of size 3 but 36 steps for an input of size 6 has a quadratic $O(n^2)$ scaling. For example, a merge sort (common sorting algorithm based on merging sorted subgroups) has a time complexity of $O(n \log n)$ and a memory complexity of $O(n)$ while the naive inversion sort has a time complexity of $O(n^2)$.

The point of this complex[14] exercise is that, once you know the scaling behavior of your algorithm and have verified its results using a small testing dataset, the only problems that can arise with larger data are implementation-bound (file systems, etc.). This is key in several approaches presented in this manuscript.

1.3.4.2. Modularity of the pipelines

To parallelize these independent tasks and handle their synchronization, meaning each task should only be run once its input is available, worfklow managers have been developed such as Snakemake or Nextflow. The former was used in the demonstration workflows of approaches presented in this thesis.

Snakemake allows easy parallelization of independent tasks, and is easy to modularize. It is based on the creation of a graph of elementary rules, each having input files and output files. The manager generates a task for each rule, starting from the desired final output files and rewinding back up the graph up to the required input files. When a file is missing, Snakemake will try to see if it has a rule to create it, and

14. Pun intended.

will then go to this rule's requirements. The process stops once Snakemake finds an already existing file, otherwise an error is returned.

As a result, elemental operations (ie. alignment of reads on a genome) can be repeated easily for different tasks, and can be made adaptable as elementary building blocks through the use of wildcards.

Snakemake supports parallelization of jobs by dispatching them to different nodes. Since it checks whether a desired file already exists, it is useful for start-and-stop where if a step ends with an error the workflow can pick up where it left off, and when one file is deleted only the steps required to recreate it will be run.

I would note that, as algorithms, the individual approaches developed in my thesis are not parallelizable in and of themselves. However, we have implemented demonstrations as Snakemake pipelines. As such, we show how data parallelization (running the approaches for different datasets in parallel) can be performed by the workflow manager through the Command Line Interface of the approaches. They are, furthermore, parallelized by threads and can be run on all cores of a node.

1.3.4.3. Reproducibility

In recent decades the scientific community has begun to realize, to its great dismay, that there is something rotten in our kingdom. Namely, that we are facing a reproducibility crisis. This phenomenon, present in most scientific domains, designates the increasing number of publications presenting experimental results that cannot be reproduced (Baker 2016). To combat this, new research paradigms have been developed and proposed. The end goal is to ensure that methods are reproducible, meaning that *given the exact same input data, anyone can obtain the exact same results* (Goodman, Fanelli, and Ioannidis 2016). This is done by providing sufficient information about all the procedures used. This is not the same thing as independent verification, where a completely new experiment is used to verify a proposed phenomena.

Practically speaking, such reproducibility of methods is much easier to implement with procedures that are strictly determinisitic. This is the case in computer science, but biological experiments have many more uncertainties due to both their inherent noise and the impossibility to perfectly control all input variables in the real world. While experimental science must strive to reduce such uncertainties so results can be reproduced, bioinformatics does not have this excuse. By making the same input data available, anyone must be able to reach the exact same results as the ones presented by the researcher. Anything else can only be described as a gross miscarriage of the scientific method.

With the philosophical question settled, the practical one becomes, "How can we facilitate reproducibility?". Modular pipelines as presented above are part of the answer. Other recommendations include sharing the full code of one's pipeline, using versioning tools such as Git. The latter is also good practice in collaborative development. It is also important that the environment of execution be reproducible. It should be possible to easily use the exact same versions of the tools used in an

analysis. This can be achieved through the use of Conda environments, which is a package manager that ostensibly help solve versioning and dependency problems, but as a consequence facilitates reproducibility. Another possibility is the use of virtualization, such as the self-contained Docker images.

On a related note, I would add that the practice of using unit tests when developing a tool is also relevant to this goal. Unit tests as small, proof-of-concept tests for the elementary procedures of an approach where the expected result is known and can be verified. They are supplemented by functional tests, which test entire slices of functionality at a higher level. While it does not directly impact reproductibility, it impact the robustness of a tool and ensures future development do not accidentally reach incorrect results.

More to the point of this thesis, this lack of reproducibility is also a considerable source of *noise* as discussed earlier, and of interpretability problems in general.

1.3.4.4. Interoperability

Related to the issue of reproducibility is the issue of interoperability. For both the input data and the output resulting data, it is important that anyone may be able to access and interpret it without difficulty. This has been standardized through the FAIR principles. A given data element is compliant with FAIR principles if it is Findable, Accessible, Interoperable and Reusable.

In practice, this means several things. It should be easily accessible from a shared resource. The metadata describing it unambigouous with standardized identifiers (eg. the gene CTCF has the unique Ensembl ID of ENSG00000102974,). It should also be complete with all necessary details about how data was created and by whom. The formatting of the metadata should be standardized (RDF [15] is recommended by the W3C).

Data formats Storing all this data required the introduction, early on, of standardized data formats. This was a crucial step towards interoperatbility, and allowed data sharing. Our goal here is not to give an exhaustive inventory of data formats used in bioinformatics, nor to discuss the data compression methods that allow to store more information in less space. but instead to discuss the aspects of the data they focus on.

Raw genomic sequence is usually stored as a FASTA file, which is simply a text file with identifiers. Sequencing reads will be provided as a FASTQC file incomporating quality information from the sequencing about its confidence in the inputted base pairs. The alignment of reads on a reference genome is given as a SAM file, which contains information about the confidence and qualiy of the alignment (gaps).

Treated genomic signal, such as given by the number of reads mapping to each position on the genome, is usually provided as a Wiggle (WIG) file, representing the value of the signal on contiguous intervals. One notes the use of language reserved

15. Resource Description Framework. Usually implemented as XML, it consists in a collection of statements framed with an object, a subject and a predicate.

for time-series in this description. Parenthetically, VCF files represent variations on a genomic positions such as polymorphisms (including but not limited to Single Nucleotide Polymorphisms).

Speaking of genomic positions, the position of genomic features is mainly given as a BED file. It is a tabulated file with one line per feature and with the columns representing feature attributes. The three mandatory columns represent respectively a feature's chromosome, start position, and end position. *Which is sufficient to define an interval on the genome.* The other columns give respectively a name, a custom score, and which strand the feature is found on, if applicable. An extension of the BED format is the GFF/GTF (General Feature File) format, which incorporates additional columns for storing various attributes. BEDtools (Quinlan and Hall 2010) is commonly used tool-suite for the manipulation of those files.

Such genomic features can be **anything**. Positions of ChIP-Seq peaks, promoters of certain genes of interest determined by some arcane method, etc. And here lies the crucial fact: many different types of relevant genetic information can be stored as lists of intervals and therefore are suitable for study using the methods presented in this thesis.

1.4. Formal modelisation

In the previous sections, we have presented human genetic cis-regulation and the methods used to study it. A partial conclusion to this is that recent assays have provided the community with a veritable wealth of exploitable data. The fact that this data has been collated in databases means the combinations between different datasets, but also the biologically meaningful combinations between different regulators, can now be exploited. However, such high quantities of data are for now mostly unsupervised.

In this section, we present the mathematical modelisation used to represent them, with some generalities about Machine Learning. Then we discuss the problems, in a mathematical sense, that the approaches in this thesis are designed to tackle.

1.4.1. Mathematical representation of biological regions

As has been discussed in the Introduction, the regulatory elements of interest fixate on given position on the genome, resulting in the definition of Cis-Regulatory Elements. The end product of many experimental genomic assays, such as ChIP-Seq, will give a list of positions on the genomes corresponding to the putative binding sites of transcriptional regulators, or more generally epigenomic regulators and chromatin elements.

The position of such elements, as well as the regions they define, can then be represented as a *list on intervals* along the entire genome. Conversely, this means the methods presented in this manuscript can be applied to any data that can be

represented as a list of intervals. For example, "promoters of overexpressed genes in the condition ABC" also fit that definition.

Interval sets In mathematics, the most common meaning of an *interval* is as a set of real numbers containing all real numbers lying between its boundaries, for example $[0;1] = \{x \in \mathbb{R} | 0 \leq x \leq 1\}$. However, intervals can be defined on any partial ordered set (*poset*). A poset is a set associated with a binary relation allowing to position one element before another in the set. This relation need not be applicable to any two elements (hence the *partially* ordered moniker), which is indeed the case for a family tree. It must, however, be reflexive, antisymmetric, and transitive.

In the context of this manuscript, we use the common generalization of ordered sets to discrete sets of time positions. As a result, intervals are defined as sets of contiguous time positions. This brings us to the definition of time series, which is a series of data point indexed in time order. The value of a time series y at the time t is denoted $y(t)$. The analogy is that here, *time* is the the position along the genome, with each step being one nucleotide. We introduce the following notation:

Definition 2. *Let A_i be a genomic region, that is a position interval on the genome (*eg. *$A_i = [100;200]_{chr1}$ = "chromosome 1, base pairs 100 to 200"). Then, the set $A = \{A_i\}_{i \in [1..n]}$ is defined as a finite set of individual genomic regions.*

1.4.1.1. Matrix representations

As we are interested in studying the combinations of regions encountered along the genome, having discussed the notion of intervals, we now introduce the combinations thereof. If the intervals represent the binding of a given regulator (or any other element, see above) at a given position, then if two regulators A and B are present at the same position, the sets containing all their binding sites should overlap at this point:

Definition 3. *A combination $\gamma = \{A + B + C\}$ is defined whenever genomic regions from the interval sets A, B and C embed a common genomic position. Combinations can be defined on any $n \geq 2$ sets.*

Definition 4. *The number of sets in a combination γ is its cardinality, noted* $\text{card}(\gamma) = n$.

For example, consider the presence, at a given position on the genome, of regions in the sets A and B, but not C. This means this position can be represented as the following vector:

$$x = \begin{pmatrix} 1 & 1 & 0 \end{pmatrix}$$

By concatenating vectors of this type, we can produce a matrix such as:

$$X = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Where each line represents a genomic position, giving the sets which have an open region at said position with one set per column.

Representations of this kind are essential in itemset mining: indeed, this representation is simply equivalent to a matrix representation of a list of transactions, with one transaction per line and one element per column. We can easily see how factoring this matrix would allow us to extract relevant itemsets. This is explained in section 4.4.2.1 (p. 165).

Furthermore, each line does not necessarily represent only one basepair or genomic position. It can instead represent a set of contiguous base pairs with the same configuration of regions (binning) or intervals between critical events (opening or closing of a region in a set). See the OLOGRAM-MODL paper for more, especially the Figure 1.

1.4.1.2. Tensor representation

The matrix representation we described shows how adding a new dimension to the classical time dimension, resulting is a two-dimensional object, allows one to consider combinations. This could be extended to compare combinations between datasets of a different nature. For example, combinations between different Transcriptional Regulators, but also between biological replicates for the same regulator. However, simply concatenating the column vectors for these datasets would give them equal billing and belie the fact that biological replicates are more closely associated with each other than with the other regulators.

In broad strokes, multi-view integration in machine learning and elsewhere is divided into three possible approaches. The first is early fusion, where the different views are simply concatenated into a larger object to be processed. There is also late fusion, where the views are processed independently by models, and the final result is calculated based on the individual verdicts. Finally, the most challenging but most relevant is middle, fusion where latent space representation of the views are considered, not the views themselves. These representations can be internal to the ML models and be fed to another model, used in a vote, etc. The bottom line is that, in multi-view learning, one has to either adapt the data structure, or produce a different representation through a model and feed the results to a different model. More specifically, for multi-omics integration in bioinformatics, see section 1.3.2 (p. 43).

As a consequence, we also consider tensor-based representations, with three or more axes [16]. Conceptually, this is obtained by stacking several matrices (or tensors)

16. "Axes" is the plural of "axis", not of "axe". To all my lumberjack and Frankish readers, I apologize.

of the same dimensions along a new axis to form an n-dimensional object.

The particular representation used in this thesis (for the atyPeak project, see figure 1.19) has three axes: genomic position, Transcriptional Regulator, and dataset ID of the experiment. However, any relevant axis can be used, for example by combining different types of assays for the same position on the third axis, or comparing Transcriptional Regulator positions on cell lines instead of on different datasets. Of course, as in multi-view learning, there should be some sort of relation between the axes to justify their use. The matrix representations are the 2D analogue of this, considering only a projection of this on two axes.

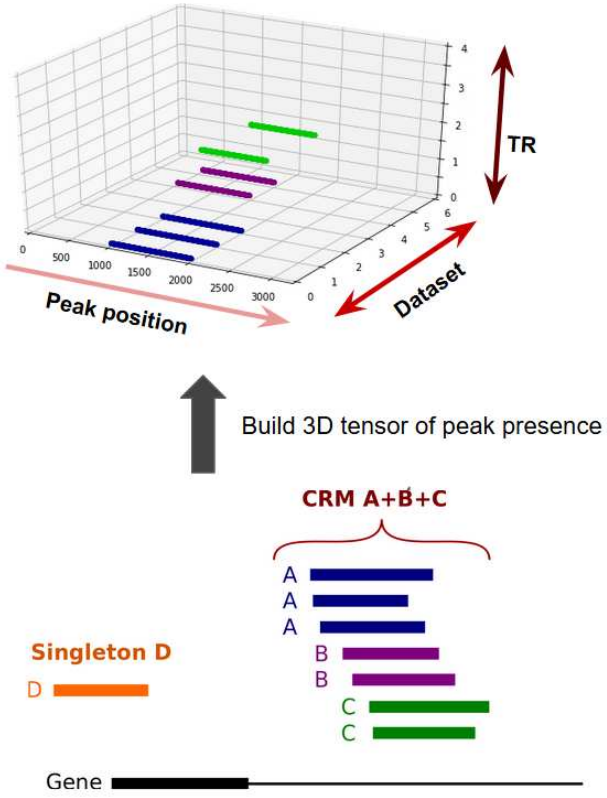


Figure 1.19. – Tensor representation of CRMs, which are candidate regulatory regions. We consider three axes: genomic position, Transcriptional Regulator, and dataset ID of the experiment. A value of 1 means there is a peak with these characteristics at this position, 0 otherwise.

Background and definition *Stricto sensu*, a tensor is defined as a multilinear application on a vector space V. It is an object which, given k vectors and h linear forms of

V, returns a single scalar. An example of tensor is a constraint tensor of all the forces applied to a physical object. In this thesis however, we use a more classical definition of a tensor as used in statistics and machine learning (Bi, X. Tang, Yuan, et al. 2021). They are seen as a generalization of scalars (of dimension 0), vectors (dimension 1) and matrices (dimension 2) to n-dimensional arrays.

Usage Stacking different, but comparable, kinds of information as tensors is precedented in bioinformatics. For example, comparing the distribution of fish communities across time and space (Frelat, Lindegren, Denker, et al. 2017). However, as of writing, this representation is still much less common than matrices or graphs. it is still rarely used compare to matrices or graphs. Our approach inscribes itself in this continuity.

Of course, a representation by itself is meaningless. A representation is to be chosen because it highlights characteristics that we want to exploit. In return, it is necessary to use approaches and algorithms that leverage the representation's strength while acknowledging its peculiarities. More specifically, one must not believe all axes to be of an equivalent nature (unlike, say, longitude and latitude for geographical coordinates). Furthermore, for some axes (like an axis listing datasets ID) ordering along the axis is unimportant. This is discussed further in section (3.3, p. 95). Parenthetically, another possible meaning of a third axis is to represent different timestamps (or, as is more commonly done, stages of cellular development) since TF fixation is not constant in time.

Such approaches can be tensor decompositions. For example, the Tucker decomposition consists of decomposing a 3rd-order (3 dimensional) tensor T as $T = \mathcal{G} \times_1 X \times_2 Y \times_3 Z$ where $\mathcal{G}$ is a core tensor and the remaining terms are matrices. It is a generalization of Singular Value Decomposition (and can be generalized to higher orders) and is used to generalize Principal Component Analysis. Another example of decomposition is the CP decomposition, written as $T = \sum_i^k \boldsymbol{a}_i \otimes \boldsymbol{b}_i \otimes \boldsymbol{c}_i$ where $\boldsymbol{a}_i$, $\boldsymbol{b}_i$, and $\boldsymbol{c}_i$ are k trios of vectors. From these examples, we can see that tensor decomposition, unlike matrix decomposition, usually results in a decomposition into components of smaller order (dimension).

There is further precedent for the use of the aforementioned decomposition methods on tensor representations, such as in studying single cell RNA-Seq data (Taguchi and Turki 2019) and more specifically for multi-view applications by integrating different genomic and epigenomic datasets (Fang 2019). However, since the tensor as a mathematical object is at the heart of all Deep Learning approaches, and the latter are much more common, in practice tensor manipulation is more the purview of neural networks methods [17].

17. Of course, the tensors involved can themselves be decomposed and studied using the aforementioned methods.

1.4.2. Generalities on Machine Learning

In this thesis, we are interested in leveraging the relations between combinations of elements. This means we want to *learn* underlying properties in the data for a variety of biologically relevant tasks. Having defined a mathematical representations of the combinations, we now introduce the methods that can leverage these representations and that required the data to be formatted as matrices and tensors: **Machine Learning (ML) methods**.

Machine Learning is defined as the study of computer algorithms that improve through experience, as opposed to manual modifications or explicit instructions by a scientist. As a broad generalization, machine learning consists of building a mathematical model with predetermined structure based on sample *training* data. It is used to make predictions about other data that was heretofore unseen by the model, with the goal of limiting the errors made on those predictions. Machine learning has a wide range of applications, from predictive business analytics to computer vision and email filtering.

The term machine learning was coined in 1959 by Arthur Samuel. Machine learning is sometimes seen as a subset of the field of artificial intelligence, and grew out of the quest for universal A.I. It also grew out of the field of statistics, and the case is made to regroup statistical and ML as a broader "data science" scientific field. Similarly, data mining employs much of the same methods as machine learning, but is characterized as discovering unknown properties in the data as opposed to prediction and is related to unsupervised learning. Perhaps a more interesting fundamental link is with optimization: many learning problems are formulated as minimization of some loss function on a training set of examples, where the loss function expresses the difference between the real properties of the data and the properties predicted by the ML model.

Parenthetically, people tend to conflate *big data* and *machine learning* due to the fact that, in ML, the more meaningful data there is to learn upon, the better the learning. In this part, we present generalities about the classification and purpose of machine learning approaches. In the interest of clarity, functional details about the methods used are only introduced when germane to the scientific problematic being treated.

1.4.2.1. Mathematical foundations

Let us begin by presenting some common notations. For each example in the data, let x be the input variables containing the information about this example (x is usually a vector). Let y be the output variable(s). (x_i, y_i) constitutes the i-th example. The goal of most ML models is to learn a hypothesis function h_θ to estimate the output variables when given the input variables, where θ is the vector of parameters of the model, so that $h_\theta(x) = \hat{y}$. In practice, this means we seek to minimize a loss [18] function

18. The loss function is sometimes improperly referred to as the "cost function".

$J(\theta) = f(h_\theta(x), y)$ based on the difference between $h_\theta(x)$ and y, which is often the Mean Squared Error. A common addition is regularization, adding a smoothing term λ to the loss that depends on the model parameters, so that $J(\theta) = f(h_\theta(x), y) + \lambda * g(\theta)$. This is usually used to keep $\sum \theta$ small.

In many cases, finding $\theta^* = \text{argmin}_\theta J(\theta)$ is done by gradient descent. This is not always the case however, and we show at least one example of a different minimization algorithm in this thesis. Gradient descent is a method where, at each learning step, each element of the parameter vector θ is updated as $\theta_j := \theta_j + \alpha \frac{\partial}{\partial \theta_j} J(\theta)$. The choice of the learning rate α is important, as a too high α will let the algorithm jump out of local minima, while a too low α will get stuck in them. Some approaches dynamically adjust the learning rate (Kingma and Ba 2014). There are variants, like batch gradient descent where $J(\Theta)$ is not calculated on all x but only a subset each time (this particular variant is used in deep learning). Gradient descent is effective for convex or partially convex cost functions.

1.4.2.2. Classification of Machine Learning approaches

Machine learning approaches can be divided into four main classes. First and most common is *supervised learning*, where the training data contains inputs x and desired outputs y. This includes mainstays like classifications and regression. Tangentially, *semi-supervised learning* is a case where some of the aformentioned examples are missing labels. In *unsupervised learning* however, the examples have no labels. This family of approaches mostly concerns itself with trying to find a structure in the input data, such as with clustering algorithm. Finally *reinforcement learning* uses dynamic programming methods and does not require explicitly labeled input data, instead seeking to explore the space of solutions and reinforces the weights of solutions producing a desired outcome during exploration.

Here is a short list of the most common Machine Learning models. Some of them are relevant for several of the aforementioned approches. The models used in this thesis will be introduced in more detail when required.

— **Regression** methods, such as linear or polynomial regressions where the hypothesis function is a linear or polynomial combination of the x_i. Logistic regression also exists.
— **Decision trees** are tree structures using successive thresholds on the input features x_i to sort the examples into boxes of high purity (composed as much as possible of a single class).
— **Support Vector Machines** (SVM) look for a separating hyperplan between two predefined classes of examples, by mapping the input into a high dimensional feature space. This relies on calculating distance analogs between the examples using a kernel function to produce a Gram matrix of distances. As a result, the representation space has been transformed in a larger dimensional space where a linear classifier can be used: this is known as the *kernel trick*.
— **Neural networks** are an assembly of logistical regressions capable of learning

complex non-linear hypothesis functions. See section 3.1 for a much more exhaustive presentation.

— **Matrix factorization** separate the data matrix into a product of other matrices whose components are significant in some way. See section 4.4.2.1 for more details.
— **Bayesian methods**, where the probability of observed events is modulated using a prior probability distribution that can be iteratively updated.
— **Genetic algorithms** are a metaheuristic inspired by the process of natural selection where parameter vectors θ giving high-quality solutions are hybridized in an attempt to create even better solutions.
— **Ensemblist methods** consist of regrouping several predictors among those presented here. For instance, *Random Forest* methods consists of using several decision trees trained on different random subsets of the data, followed by a majority vote. Another example of ensemblist method is *boosting*: during the training, one trains subsequent models where hard examples for the previous model are given more weighting. The final result is a weighted vote.

Which approach to use depends on the problematic to be solved. Do we want information about the features? Do we want to predict the status of an unknown example (for example, a cancer patient)? For a general review of machine learning algorithms and more details on the approaches presented here, see Dhall, Kaur, and Juneja 2020 [19].

1.4.2.3. Evaluation of ML models

In practice, the data is often divided into two sets: a larger training set on which the model is trained, and a testing set never seen before by the model. This ensures the model has no opportunity to learn the latter "by heart" and its performance can be accurately assessed on it, a process known as cross-validation. A model is said to be overfitting (ie. high variance) when J(train) is low and J(test) is high, meaning it learned the test data too well and does not generalize. Conversely an underfitting model (ie. high biais) has both high J(train) and J(test), meaning the model is inadapted to the data. This is generally due to the model not having enough entropic capcity.

Binary classification models can be further evaluated by computing their precision $P = \frac{TP}{TP+FP}$ and their recall $R = \frac{TP}{TP+FN}$, where TP designates for each category in the output the number of True Positives, FP of False Positives and FN of False Negatives.

Toolsets The most widespread toolsets implementing Machine Learning approaches, and the ones that were used in this thesis, are the SciKit-Learn Python library (Pedregosa, Varoquaux, Gramfort, et al. 2011) and the Keras Python library for deep

19. More specifically, an astute reader will be mostly interested in their References section. If that was also your first reflex, do let me know. I am always proud to have such insightful and attractive readers.

learning (*keras-team/keras* 2015) using the Tensorflow backend (Abadi, Barham, Jianmin Chen, et al. 2016).

1.4.3. Anomaly detection

Anomaly detection (also known as outlier detection) is the problem of the identification of rare items, events or observations which raise suspicions by differing significantly from the majority of the data (Chandola, Banerjee, and V. Kumar 2009). In other words, anomaly detection seeks to identify elements in the data that do not conform to the usual patterns between examples. An anomaly can be a point anomaly, where a single point deviates from the rest of the data. It can also be a sequential or contextual anomaly, meaning a data point is not anomalous by itself, but based on its neighborhood. This is more relevant in time series and ordered data. For recent reviews of anomaly detection, see X. Xu, H. Liu, and Yao 2019 for a general review (with a focus on high dimensional data) and Akoglu, Tong, and Koutra 2015 for graph data.

This is relevant for our purposes, as the noise we presented in section 1.3.3 (p. 44) will likely not conform to the usual patterns observed in non-noisy (correct) observations. For example, as is relevant in the work being presented, finding a sub-unit of a regulatory complex without the other associated elements would be suspicious. This means this noise can be characterized as an anomaly with the definition given here, and detected with such approaches.

1.4.3.1. Usual methods

Broadly speaking, anomaly detection can be done using a supervised or an unsupervised approach. Supervision tends to make the training easier and improve detection of the sought type of anomaly. However, it also tends to bias the model towards the particular kind of anomaly found in the training set and will conversely not generalize to other anomalies. Unsupervised methods do not introduce such bias as they rely on the intrinsic characteristic of the data under the assumption that anomalous data points will have differing characteristics, regardless of the source of the anomaly. However, they rely on the assumption that normal instances are far more frequent than abnormal ones. This is true in most cases, but not all. Regardless, when considering genomic assays data labeled training sets of anomalous data are seldom available, as discussed previously. This means we must usually resort to unsupervised detection.

Classification-based methods often feature known Machine Learning classification methods (DNN, SVM, etc., see above). On the other hand, clustering-based methods will instead seek to form data clusters, with the assumption that anomalous points (1) do not below to a cluster, or (2) will be far from their cluster's centroid, or (3) anomalies will belong to smaller or sparser clusters. Finally, statistical approaches seek to fit a model to the underlying data, with the assumption that anomalies will have a low value on the probability density function. It should be noted that most

anomaly detection algorithms return a real value known as an anomaly score. This permits custom thresholding later, to split the input space into *normal* and *abnormal* elements.

1.4.3.2. Compression

Compression methods can also be used to perform anomaly detection. Compressing data means creating a smaller (in bit size) version of it that can be used to later rebuild the original data. In lossless compression, the rebuilding is exact. In lossy compression however, an error tolerance is added so that the rebuilt data may only be an approximate match for the original, up to the error term.

When performing a lossy compression, noise and other non-information are the first elements to be lost. However, fine-grained details will be lost, under the same assumption that they are too rare to constitute a meanigful pattern. A more intereesting consequence of this is that anomalies, by definition, tend to be lost on compression. This can be done with an autoencoder Ponomarenko, Lukin, Zriakhov, et al. 2005, and we show an example of this in this manuscript. More details are presented in the Methods section of the atyPeak paper. But any compression method is suitable.

1.4.4. Frequent itemset mining

Frequent itemset mining (also known as association rule mining) was popularized by Agrawal, Imieliński, and Swami 1993. It consists of identifying combinations [20] of elements that are often found together. See Luna, Fournier-Viger, and Ventura 2019 for a recent review. The iconic problem of the field is to find regularities in the shopping behavior of customers in a shop, but it can be generalized to any co-occurences between occurences drawn from sets of elements.

More formally, let I be a set of items. An itemset is a subset of I. D is a transactional database, whose individual elements are transactions. Each transaction is a subset of I. In each transaction, some of the possible elements of I are purchased. Our goal is to find frequent patterns of items (ie. itemsets) purchased/retrieved together. For the problems presented in this thesis, a transaction corresponds to a genomic position, and the items are the various genomic regulators and chromatin elements that are bound in this position. Biologically speaking, since those regulators work in combinations, there is a need of a framework to work on and study those combinations and identify combinations of interest.

Let P be one such frequent itemset, or association rule. A rule is defined as an implication of the form $X \Rightarrow Y$ where $X, Y \subseteq I$. The corresponding itemset to this rule is $X \cup Y$, with X the antecedent and Y the consequent. Transaction databases are usually represented using the same matrix representation we introduced in section

20. Drinking game: take a shot every time you read the word "combination" in my thesis. If you make it to the end of my manuscript alive, feel free to email me so I may send you my heartfelt congratulations. Then proceed to sign yourself up in your friendly neighborhood Alcoholics Anonymous meeting.

1.4.1 (p. 52), where D is also be represented as a matrix $X \in \mathbb{R}^{m \times k}$ where m is the number of transactions and k is the number of sets.

For instance, consider the database D containing the following transactions[21]:

Customer	**Shopping cart**
Charles	cheese, wine
Louis	wine, bread
Lothaire	cheese, wine, olives

It can be represented as the following matrix, where each line is a transaction and the columns represent respectively *cheese*, *bread*, *wine* and *olives*:

$$D = \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \end{pmatrix}$$

Some additional metrics can be defined. The support of an itemset P in the database D is an indication of how frequent it is:

$$\text{support}(P, D) = |\{t | t \in D \wedge P \subseteq t\}|$$

In the above example, support for $P = \{\text{cheese}, \text{wine}\}$ is $\frac{2}{3}$ since it appears in two out of three transactions. Conversely, the confidence of a given association rule $X \Rightarrow Y$ is simply $\frac{\text{support}(X \cup Y)}{\text{support}(X)}$. Related to association rules, we introduce for the purposes of this thesis the following definition of parent combinations:

Definition 5. *A combination γ_1 may include all the sets of a combination γ_2, plus some others: γ_2 is the parent and γ_1 is the child of the relationship, denoted by $\gamma_2 \preceq \gamma_1$.*

Indeed, combinations are analogous to itemsets, and a child combination is analogous to a subset of an itemset. Note that this subset is is an itemset itself.

It is obvious at a glance that for $\#I = k$ (meaning the itemset I contains k items) the set of all its possible subsets is simply the power set over I, which has $2^k - 1$ members. To limit this exponential complexity, itemset miners limit the returned rules by a minimum support threshold. Efficient search is possible thanks to the downward-closure property, which simply asserts that for a frequent itemset, all its subsets are also frequent. As a result a frequent itemset must have at least one frequent parent, and it is not necessary to work on the children of non-frequent itemsets.

1.4.4.1. Algorithms

The classical algorithms used in frequent itemset mining include the following:

21. Based on their purchases, can you guess which part of the Holy Roman Empire each client inherited?

— *Apriori* itself, considered the progenitor for the entire field (Agrawal, Mannila, Srikant, et al. 1996). Its general principle is to perform a breadth-first search. This consists of counting the occurences of all 1-wise itemsets, then moving on to evaluate the 2-wise candidates, etc. and spotting once the itemset considered is no longer frequent. Thanks to the downward closure property, we know the children of a non-frequent itemset cannot be frequent. Although it is now of a venerable age[22], it is still being worked on today. Its complexity scaling is relatively poor, but the answers it gives are both correct and exhaustive. Subsequent algorithms are based on its principle.
— *FP-Growth* is an improved version of apriori. While the principle remains broadly the same, the database is instead sorted into a tree of frequent patterns, for efficient query and browsing. To do so, the transactions in the database are arranged into a *trie*[23]. Each node of the trie is an item, the children of each nodes are the associated items in each transaction (Han 2004).
— *ECLAT* on the other hand used a depth-first seach algorithm (Zaki 2000). This makes it good for parallel execution.
— Closed itemset miners seek to directly extract closed itemsets. An itemset X is closed if it is frequent and has no children $Y \preceq X$ with the same support. For example, LCM (Linear time Closed itemset Miner) is based on prefix-preserving closure extension, meaning itemsets are ordered in a search tree and they jump from closed itemset to closed itemset only (Uno, Kiyomi, and Arimura 2004).
— Approximate itemset mining, where the itemsets I' can be written as $I + \boldsymbol{e}$, where I is an itemset and $\boldsymbol{e}$ a vector of slack variables, to help fight noise. In such a scenario, "ABD" might still be counted when looking for "ABC", depending on the permissivity. These methods are used in noisy datasets, to introduce a tolerance to noise. For example, the KRIMP method (Vreeken, Leeuwen, and Siebes 2011) is based on compression (to be more specific, a given itemset is selected depending on how it helps rebuild all itemsets).
— There exist others methods, such as ASSOC or OPUS, and with itemset mining being an important topic in e-commerce, research is ongoing.

It should be noted that the time gains for FP-Growth and ECLAT are, for usual cases, only of about an order of magnitude compared to Apriori (Garg and D. Kumar 2013).

Furthermore, there exists a link between closed itemset mining and approximate itemset mining: since closed itemset mining is vulnerable to noise, approximate itemset mining is often used as part of a solution to find closed itemsets (Junbo Chen, B. Zhou, X. Wang, et al. 2009).

22. Granted, it does not hold a candle to Euclid's algorithm, but still. It is, in fact, merely five months older than I am. Positively ancient and decrepit. *Get off my lawn, you damn youngsters.*
23. This is not a misspelling of "tree". It actually means "digital tree", or "prefix tree".

1.4.5. State of the art

Up until now, we have presented the context and background necessary to understand the results presented in this thesis. As to the question of direct precedents, the state of the art is accessible to the reader in the Introductions (and Methods) sections of the papers attached to this thesis, as well as in the commentary in the following chapters. For the reader's convenience, we summarize those here.

The significance of the overlap between genomic overlaps has garnered some interest before with several approaches (Simovski, Kanduri, Gundersen, et al. 2018), with Bedtools Fisher being the most popular. For overlaps between $n > 2$ sets, some work has been done (Aszódi 2012) but the models used are often inadapted, or only empirical, requiring large amounts of shuffles to get any sort of precision.

Combinations of epigenetic regulators have been studies in CRE before (L. Teng, B. He, Gao, et al. 2014) and even used to predict their status (Vandel, Cassan, Lebre, et al. 2017). Some of these approaches use matrix factorization methods (Giannopoulou and Elemento 2013). However, they tend to be focused on pairwise interactions and/or on combinations of individual discriminant regulators (some regulators that are correlated to the relevant ones can be lost in the analysis). Relatedly, some work on itemset selection based on certain criterions of interest has been performed, such as selecting the itemsets that best explain the query region sets, but their interpretation is rather opaque (Bryner, Criscione, Leith, et al. 2017). As for the statistical model, the modeling is often inadapted, using abstractions such as reducing regions to single points. Finally, the tools proposed are often either difficult to interpret or difficult to understand and use, sometimes both.

There is a precedent for the denoising of ChIP-Seq data based on combinations between different views, but it is supervised (Koh, Pierson, and Kundaje 2017), making it harder to apply in the general case. On the other hand, snsupervised anomaly detection has some precedents, but they usually refer to data that has a different structure. Indeed, a recent approach for denoising without access to clean data uses the L2 loss Lehtinen, Munkberg, Hasselgren, et al. 2018 but is about images. As for the more widespread approaches like the ENCODE IDR, they are little better than a simple pairwise correlation between datasets.

1.5. Partial conclusion

At this point, we have discussed how genetic regulation in humans and other eukaryotes relies on a complex apparatus of regulatory elements. These elements do not exist in a vacuum but have complex interactions with each other. For example, bivalent histones result in an intermediary chromatin state, and transcriptional activators may be composed of several sub-units such as FOS and JUN. It follows that the study of such combinations is of paramount importance. Such interactions mostly consist of forming complexes. As a result, they are found co-localized on the genome. As a result, approaches studying local correlations between sets of elements are particularly

adapted to work on this problem. Many different types of relevant genetic information can be stored as lists of intervals and therefore are suitable for study using the methods presented in this thesis.

> Quis, quid, ubi, quibus auxiliis, cur, quomodo, quando ?
> *Who, what, where, with what means, why, how, when?*
>
> *Quintilien*

In the following sections, the details of my work on those aspects is presented. I sought to present the impetus and purpose behind each work, and provide a commentary on why the methods developed were necessary. These explanations are given, for lack of a better word, mostly in layman's terms. My goal is to provide a exegesis of the work presented in the articles, present my line of thinking, and explain why I did what I did. The technical details and mathematical aspects are presented in the articles for the benefit of the scientific community. When that is necessary, I also include background information on the methods used.

To summarize, the *quid* and the *quomodo* are mostly in the attached articles, but the *cur*, *quando* and *quibus auxilis* are in this thesis.

2. Early work on specific Cis-Regulatory Elements

Sommaire

2.1. Combinations of regulators for CRE status prediction

In the context of this thesis, my early work focused on leveraging the combinations of epigenomic regulators to predict the status of Cis-Regulatory Elements, meaning whether they are active enhancers, inactive enhancers, active promoters, etc. This led to exploratory work which allowed me to develop the approaches presented in the following chapters. Here however, I would like to make a short aparté to discuss some other insights gained in the process.

2.1.1. Background

The prediction of Cis Regulatory Element status is a hard problem. Features that can be informative include the binding of the epigenomic regulators presented in the Introduction (Y. Li, C.-Y. Chen, Kaye, et al. 2015). A general review focused on

the challenges in this domain has been performed by Kleftogiannis, Kalnis, and Bajic 2016.

Machine Learning methods, including deep learning, have been used to try to solve this problem (S. G. Kim, Harwani, Grama, et al. 2016). The reported accuracy of these methods vary, from 0.7 (Siwo, Rider, Tan, et al. 2016) to > 0.95 (Quang and Xie 2015). However, it has been found that there is little overlap between the predictions made by many different methods (at least for enhancers), which casts them all into doubt (Benton, Talipineni, Kostka, et al. 2018). Other potential sources of error include completely separating enhancers and promoters when predicting for certain models (Y. Li, Shi, and Wasserman 2016), while we now know there can be some overlap between the two (e-promoters).

Decision trees I privileged decision trees for this problem, as they provide immediate visual interpretability. Decision trees are predictive models that go from observations about a variable's value (x_i, in the branches) to placing the samples in groups of maximum purity in terms of class or target value (y).

Their classical construction algorithm is called CART: at each step, the algorithm looks to the data and seeks a criterion. A criterion is defined as a condition on the input features, for example height > $170cm$. At each step, the criterion that will be selected is the criterion which, when applied, will result in a division of the input space in two groups of maximal purity. Purity is usually defined as a low Gini impurity. An alternative is to instead use the criterion that would result in maximal information gain. The cycle then repeats until the end conditions are met (number of nodes reached, or fully pure groups). Tangentially, such decision trees also come in *boosted* and *Random Forest* variants, but this costs them their ease of interpretability.

It is important to note that decision trees can be non-informative, but their classification of the input data is never wrong for the training data: the criteria selected are indeed those that best divide the data, and the composition of the nodes is always precisely given.

As for their limitations, they tend to use simple criteria. As such, non-linear or spatially complex decision boundaries (ie. spheres) can be difficult to learn. Furthermore, as is relevant below, a visual representation focusing on the criteria only does not show the features that could also be important, but were not selected as they are correlated to another feature that itself was selected. This is problematic for us if we look for combinations of regulators as it would show only one of the two.

Furthermore, decision trees are quite unstable, because they will still consider very small differences in the variables to be discriminant. This means that the decision rules can change heavily when new examples are introduced in the training set.

2.1.2. Results

Decision trees were applied to data from Salvatore Spicuglia's team for two different problems. We sought to find a link between the combinations of Transcription Factors

(and histone marks) present on certain CRE, and the enhancer activity of that CRE. Training was supersived, with the enhancer activity value used in the supervision assayed through Cap-STARR-Seq in both cases (see section 1.2.3, p. 38).

2.1.2.1. Lymphoid enhancers activity

The first problematic concerns the prediction of enhancer activity in lymphoid cells, using TF and histone marks ChIP-Seq data. This study is performed on the p5424 cell line [1]: their genotype is $RAG(-/-) \times P53(-/-)$, and as a result their phenotype resembles that of Double Positive developing lymphocytes [2].

The enhancer activity is discretized with the following thresholds applied to the $\log(\text{Fold Change})$ of Cap-STARR-Seq: 1,5 and 3. This results in highly imbalanced classes, with roughly 4000 Inactive, 2000 Weak and 400 Strong enhancers. This is solved by weighting by abundance in the loss function.

When processing the full data to find statistical associations (we do not concern ourselves with prediction yet), we find that the ETS1 and HEB transcription factors play determinant roles in the activity of enhancers (Figure 2.1).

As for prediction, we get an AUC [3] of 0.7 roughly for decision trees (depending on random seed), with similar values with Random Forest methods. Other tries using AdaBoost regressions confirm the data is noisy with only poor predictive power. This is similar to the AUC of 0.68 found when using TFcoop (this analysis was run by the authors of TFcoop themselves as a courtesy).

2.1.2.2. E-promoters

We also considered E-promoter prediction (promoters that exhibit significant enhancer activity, see section 1.1.2.3, p. 26, and L. T. M. Dao, Galindo-Albarrán, Castro-Mondragon, et al. 2017) in K562 cells. Here, we work on ChiP-Seq coverage data for all TFs. The two classes we try to distinguish between are (1) a control set of promoters and (2) a set of E-promoters with equivalent promoter activity (promoter activity quantified by RNA-Seq). Enhancer activity is also quantified by Cap-STARR-Seq.

In stark contrast, when processing the full data (not separating into training and testing set), E-promoters show an onion-layered decision tree structure. The enrichment for the two classes in the node enrichment is often good, unlike lymphoid enhancers.

To find Transcription Factors that are correlated to discriminant ones, but may not have been found to be discriminant themselves, I developed the following procedure: for each node in the decision tree (not just leaves), compute its enrichment in each class, then compute the average values of the features for each sample that was attributed to this node. The most interesting finding from this analysis was a mutual

1. The histone ChIP-Seq data was done on Double Positive cells, however.
2. The classical T-cell development path is: Double Negative → Double Negative pre-TCR → Double Positive with TCR α/β → Single Positive, either CD4+ or CD8+.
3. Area under the curve of Sensitivity (False Positive Rate) as a function of specificity (True Positive Rate). Since there is no output score, here we have simply $AUC = \frac{1-FPR}{TPR}$

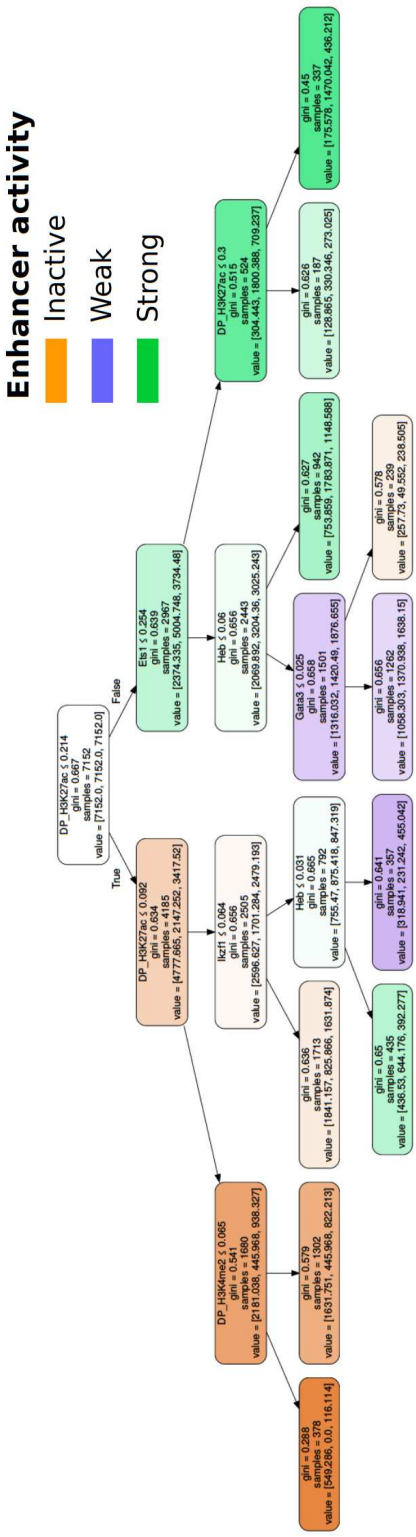


Figure 2.1. – Decision tree for lymphoid enhancers.

exclusivity between YY1 and FOS/JUN (heterodimer AP1). This was invisible in the previous analysis by S. Spicuglia team, which did not focus on combinations. Accents exist to amend those profiles, notably with the presence or absence of MYC.

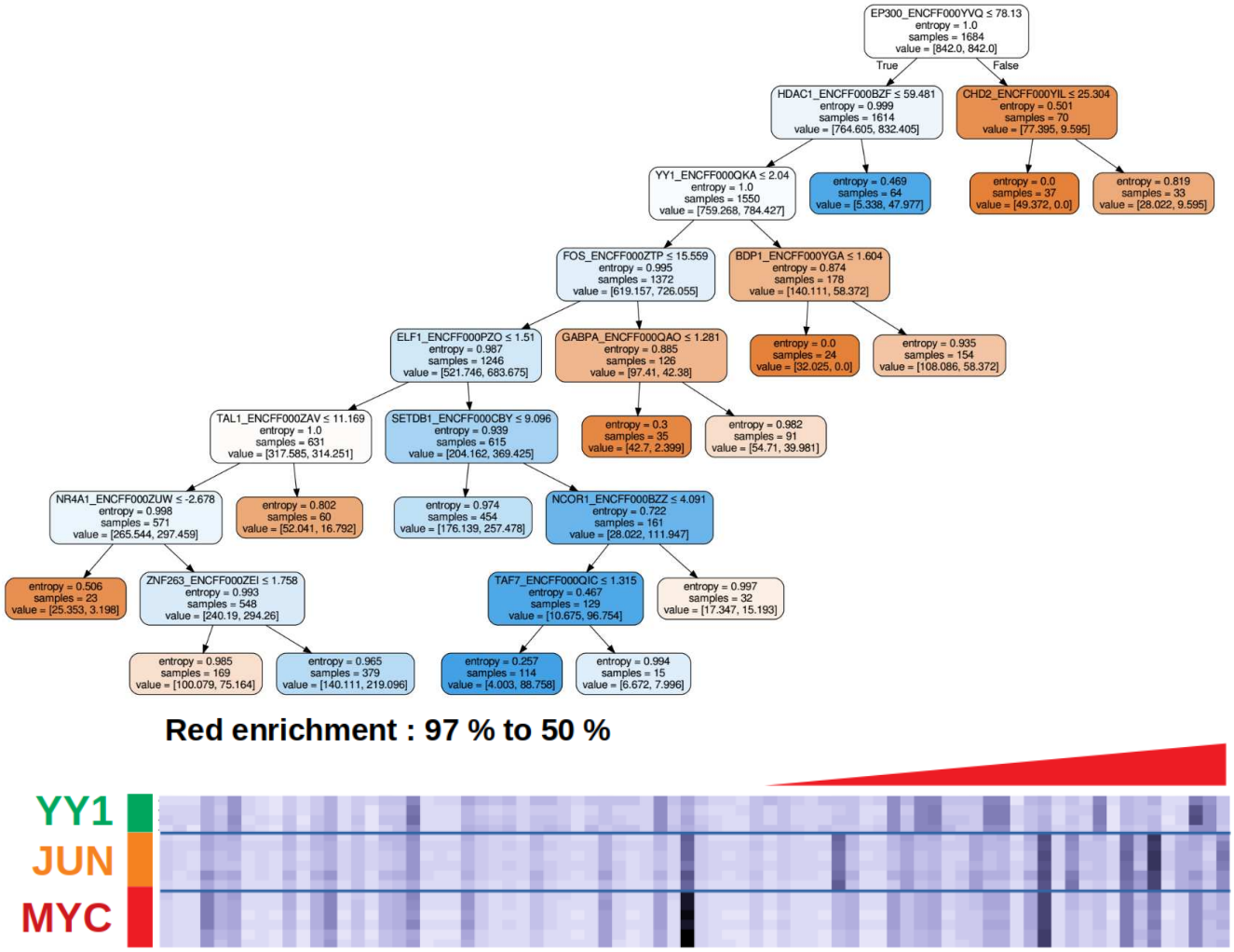


Figure 2.2. – Decision tree for E-promoters. The top figure gives a simplified decision tree on the entire data. In the bottom figure, each line gives a ChIP-Seq signal for either YY1, JUN, or MYC, while each column is a node of the decision tree, sorted by enrichment for E-promoters. Enrichment starts at 50% at the left extremity of the red triangle and goes up to 97%.

2.2. Alternative promoters in T-ALL leukemias

Another project I tackled during my thesis was the study of the usage of alternative promoters (see section 1.1.2.2, p. 24, for more background) in T-ALL leukemias. While the link to my thesis problematic of studying combinations of regulators may seem remote, it can be seen as a precursor to such studies. Indeed, the project involved the identification of Cis-Regulatory Elements, the study of ChIP-seq data, and comparison between different conditions (combinations of datasets). For me, it sensibilized me to the problematics I studied later.

Here, we focus on the study of Acute Lymphoid Leukemias (T-ALL). Those leukemias are characterized by a proliferation of immature thymocytes and high white blood cell counts, and represent roughly 15% and 25% of all acute lymphoblastic leukemias diagnoses in children and adults respectively. Outside of cancer, alternative promoters have indeed been shown to play a role in T-cell development (Chiang, Ku, Cui, et al. 2018). This is relevant since T-ALL leukemias can manifest in any of these cellular development stages.

In this study, we use H3K4me3 ChIP-Seq data from T-ALL patients to perform a genome-wide study of promoter activity in all human genes. We show that alternative promoter usage is widespread in such leukemias. We also identify a set of previously unknown candidate oncogenes with alternative transcript isoforms due to differential TSS usage.

H3K4me3 ChIP-seq data was compiled from samples of three different conditions : 11 T-ALL patients, 5 immortalized cell lines (Loucy, CCRF, SilALL, RPMI, Jurkat) and 5 healthy thymocytes cell lines (CD34, EC, SP4, SP8, LC). This data has been generated in the TAGC and in the Necker hospital in Paris. The bioinformatic processing was performed by Denis Puthier.

2.2.1. Methods developed

We retrieve TSS coordinates from the *hg38* genomic annotation, and intersect them with our H3K4me3 peaks. Intersecting peaks are considered to be marks of potential promoters, and conversely peaks that do not overlap with at least one known RefSeq TSS in at least one of the studied samples (all conditions) are discarded. Their activity is quantified using their H3K4me3 ChIP-seq coverage as a proxy for transcription activity, computed by summing the number of mapped reads for each base pair of the peak.

For each gene, a contingency table of the H3K4me3 peak coverage is generated from the data; each line represents a different promoter for the gene, and each column the sample. From this contingency table, we can compute Cramer's V-score (see Figure 2.4). This score is not interpreted using a p-value, but using a threshold (Cohen 1988). For degrees of freedom equal to and higher than 2, $V > 0.35$ is a strong association and $V > 0.21$ a medium one. We use the V score instead of Fisher's hypergeometric-based exact test or instead of the Chi-Squared tests due to the very large values (in the

millions) of the peak coverage used in the contingency tables (McDonald 2009). We make a distinction between the *global* and *local* approaches. In the global approach, peak coverage is averaged across all samples of a given condition (cell lines, healthy thymic cells, leukemic cells). In the local approach, the V score is computed for every possible pairs of samples, without grouping by condition.

Upon examination of the unfiltered best candidates, we have found that the V score is vulnerable to noise and will find spurious correlations if the coverage values are too low (see below in section 2.2.2.2). To reduce the number of false positives, we filter the genes based on the density of each of their H3K4me3 peaks (peak coverage divided by peak length): in the global approach, we require each gene to have at least two TSS where the mean peak density across all samples is higher than the median (roughly equal to σ) of densities. In the local approach, we require at least one sample to be higher than the median, instead of the mean of all samples, so as not to discard singular promoter apparition events. As a result, this V-score approach that is more robust to noise (due to thresholds) and large values than a classical Chi-Squared test.

2.2.2. Genome-wide results

2.2.2.1. Transcriptomic diversity

Out of 73704 registered H3K4me3 peaks, only 20% are associated with at least one TSS. Conversely, only 44% of all known TSS are associated with at least one peak. We remove the transcripts and peaks without such associations from the analysis. TSSs sharing the same H3K4me3 peak are merged, as we estimate those to be due to a defect [4] in RNA Pol II fixation; such very close TSSs are assumed to be regulated by the same promoter (Frith, Valen, Krogh, et al. 2008). This measure is further supported by CAGE analysis results showing that alternative promoters containing closely-packed tend to use a single, “major” TSS, unlike the more distant “true” alternative promoters (Carninci, Sandelin, Lenhard, et al. 2006). We define a “TSS cluster” as one TSS merged with all the others sharing the same H3K4me3 peak, with the 5’-foremost TSS being kept.

We find that, in our samples, only 1920 genes have more than one TSS that do not belong to the same cluster (ie. promoter, meaning that they do not share a H3K4me3 peak), or roughly 15%. This is significantly lower than the 40% figure commonly reported in the literature, but is likely due to the close proximity between the studied samples: as they are all thymoid lymphocytes, or derived thereof, it stands to reason that their (alternative) promoter usage patterns would be globally similar.

We observe than most genes have a single digit number of TSS, but there is considerable variance as some genes have dozens of TSS. Having more transcripts than TSS is very common, underlying the role of alternative splicing further down the line. We also studied the distances between the TSS of each gene (Figure 2.3): when considering

4. "Defect" in this context is not necessarily to be understood as an *error* in the negative sense, it also covers the intrinsinc incertitude in the beginning of the transcription by PolII.

the distance between transcript start sites (ie. TSS but without merging alternative isoforms due to splicing) we see a trimodal distribution. However, much of the third mode is due to transcripts with far-away TSS that do not intersect with any H3K4me3 peaks. There appears to be no particular bias in distance regarding which transcripts' start site share a peak with other genes' transcripts. Finally, the first two modes are composed of TSS that share a H3K4me3 peak between themselves, confirming our suggestion that they are RNA Pol II fixation defaults and/or variations and not different promoters unto themselves. This data contributed to our decision to equate one promoter to one peak of H3K4me3 and merge all TSS that share a H3K4me3 peak.

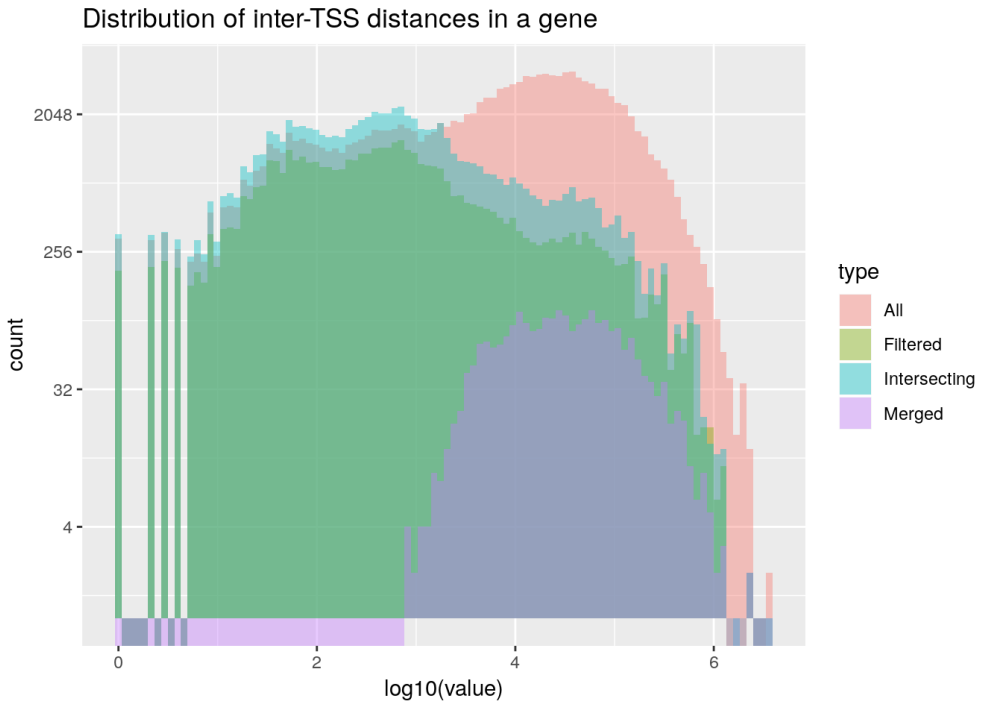


Figure 2.3. – Distribution of inter-TSS (splicing isoforms not merged) distances in a gene. "All" means all TSS. "Intersecting" means we removed TSS that intersect no H3K4me3 peak. "Filtered" means we removed TSS sharing a peak with another gene's TSS, plus all the previous steps. "Merged" means we merged TSS with the same peak, plus all the previous steps (this corresponds to the inter-peak distance). The *y* axis is in log scale, which must be kept in mind when comparing the areas under the curves.

2.2.2.2. Alternative promoter usage

We used our V-score as described, to determine whether the condition impact the H3K4me3 signal.

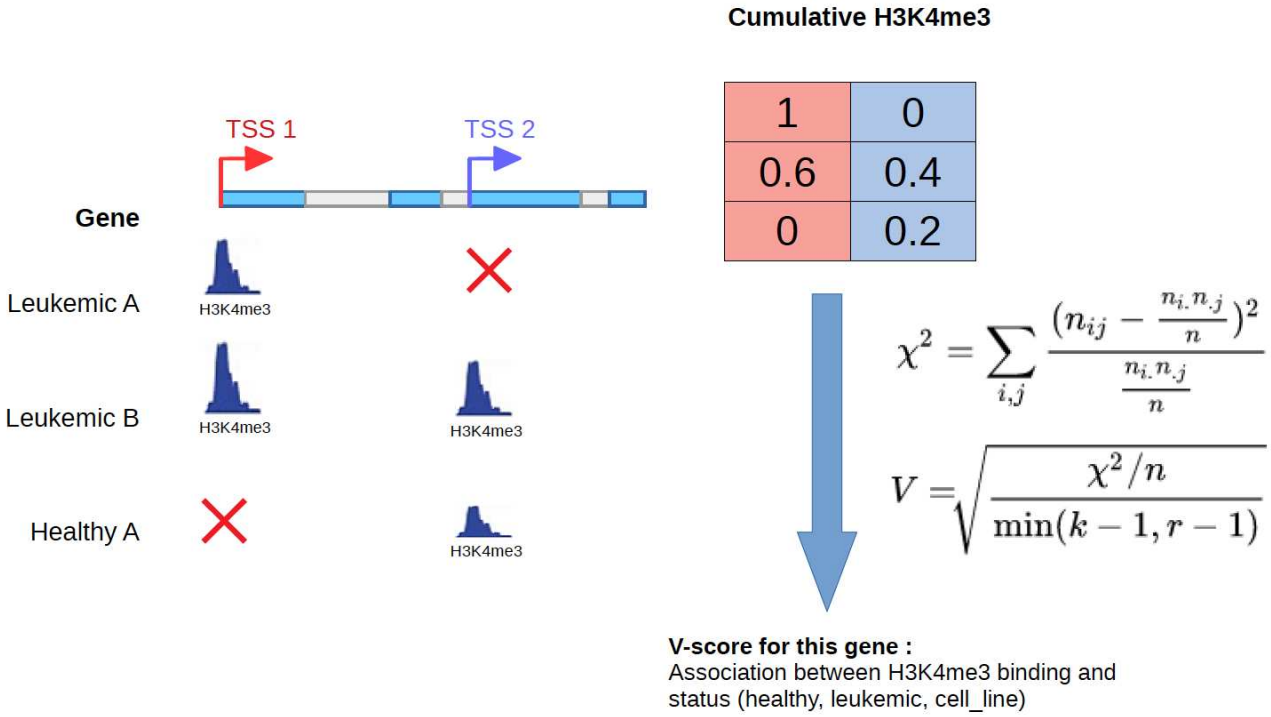


Figure 2.4. – V-score principle. The presence of an H3K4me3 peak is used as a proxy for a promoter's activity. A contigency table (top right) is generated from the number of reads found in each peak. It can be relative to the maximum values (as depicted in the figure) or contain the actual values (the values n_{ij}). The actual values are used to compute the V-score (see formula) for this contingency table. Each line of the contigency table can represent one cell sample (local approach) or all samples of a condition merged together (global approach).

The V-score is computed, and appears to correctly follow a Chi2 distribution. But, before filtering, we have reason to believe this contains many false positives (see below). There is a correlation between the two comparisons involving immortalized cell lines, which again suggests the samples of this condition have a unique alternative promoter profiles for most genes consistently different to the other cell lines and as such appears different in both comparisons.

In Figure 2.5, we can see that peaks with low total coverage (in terms of total base-pairs covered by ChIP-Seq reads) tend to have a higher global V-score. As a result, we discarded all peaks with a too low density (see Methods in section 2.2.1). Coincidentally, σ corresponded roughly to the median value of peak coverage. By reducing the number of false positives, this filter also allow us to be somewhat more lenient and put the V threshold at 0.2 - which is still a statistically significant value, as otherwise we would discard many too many genes.

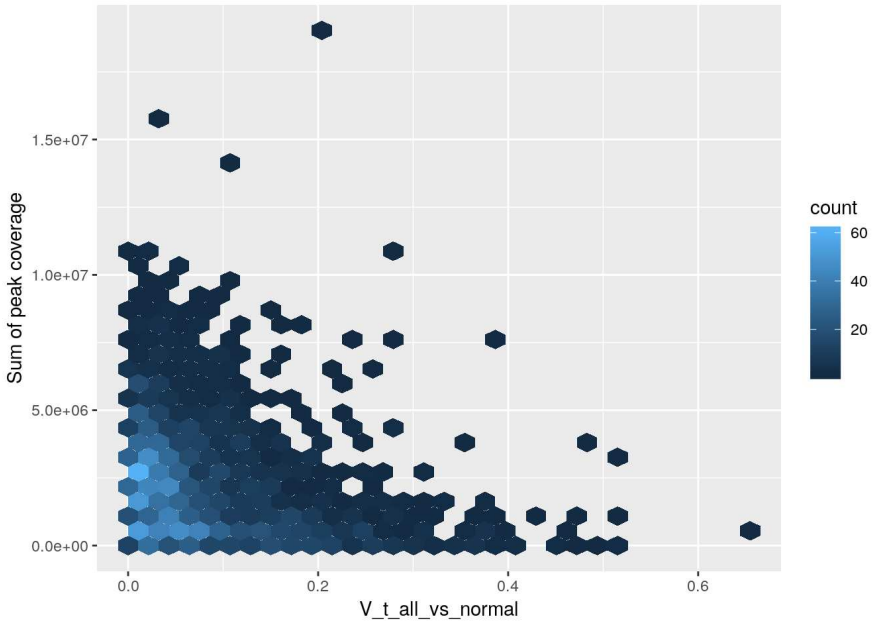


Figure 2.5. – Global V-score (T-ALL vs normal cells) as a function of total peak coverage. We suspect many false positive for high V-scores might be due to low peak coverage.

Our final filter has those two conditions (V score above 0.2, mean peak density in at least two TSS above σ), we add a third condition that there is at least one significant ANOVA ratio when calculated on the coverage table between the two conditions. For each pair of conditions ("Cell Line vs Normal", "T-ALL vs Normal", "Cell Line vs T-ALL"), we retrieve the genes that fulfill all three conditions of our filter. This results in a low number of candidate genes: in "T-ALL vs Normal", only 27 genes match.

However, there are more candidates in the local approach, where we perform a finer-grained analysis by comparing the samples themselves. For each gene, we call "maximum local V" the maximum V-score observed between any two samples. Of the 232 genes with sufficient peak coverage, 206 have their maximum local V score above 0.2 which means that most (89 %) of the genes that satisfy the peak coverage density condition also have at least one significant local alternative promoter usage. Unlike in the global approach, however, those can be isolated examples from certain samples and not condition-wide changes, as we see in the case study. Finally, we perform a manual selection of 8 candidates deemed most promising: ANKRD28, AT2C1, BCL9, MACF1, PEX5L, SSBP4, MAST1, and NRXN2.

We also perform a hierarchical clustering based on the local approach (Figure 2.6). We use the local V score as a metric to quantify distances between samples, by summing all the V scores across all genes into a single matrix and using it as a distance matrix. We do not use the peak coverage filtering condition, since the V score is used

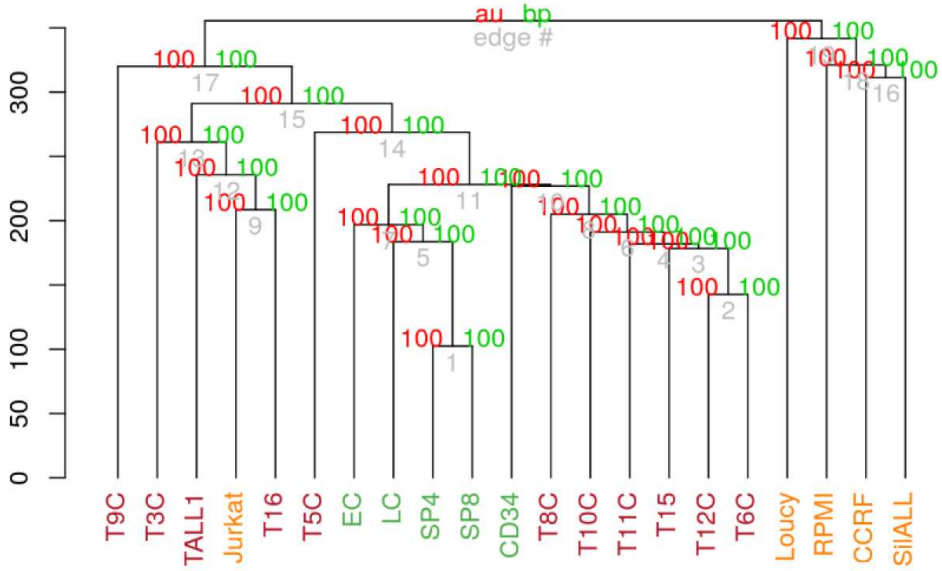


Figure 2.6. – Patient clustering based on local V-score (alternative promoter usage for all genes). Healthy samples are in green, "cell lines" in yellow, leukemic samples in red. On the graphs, AU p-values (red) are “Approximately Unbiased” p-values computer by multiple bootstrap resampling, while BP (green) is the raw “Bootstrap Probability”. Hierarchical clustering is used and 10 bootstraps were performed.

as a distance here, not as a filter. As a result, we find that the immortalized cell lines are mostly grouped together, but they can be very different between themselves as can be seen via the scale of the cladogram on the left. T3C, T15, T16 and TALL1 are grouped together. Indeed, observing the mapped ChIP-Seq peaks in a genome browser often reveals that they have their own unique profile. The healthy samples are grouped together, but the leukemic samples from two distinct clusters, one of which is closer to the healthy samples than to the other leukemic samples.

2.2.3. Case study of ATP2C1

The most promising candidate gene was found to be ATP2C1. ATP2C1 (SPCA2) is an ATPase localized in cellular membranes. Its role is to transport the Ca^{2+} and Mn^{2+} ions. It is known to play a role in the regulation of the cell cycle in cancer cell. Changes in ROS (Reactive Oxygen Species) production, caused by increased cell density and hypoxia (auto)regulate the expression of SPCA2, which in turn is involved in potential removal of ROS. Another direct effect of SPCA2 on the physiology of HCT116 cells is the increase in their proliferation, possibly through the minimization of exposure to high cytosolic Mn2+ . In conclusion, the function of ATP2C1, regulated by hypoxia

and cell density, has been previously linked to generation of ROS and regulation of cancer cell survival. (Cialfi, Le Pera, De Blasio, et al. 2016, Jenkins, Papkovsky, and Dmitriev 2016). This existing oncogenic link piqued our interest, and we show T-ALL leukemias may also be influenced by ATP2C1.

As seen in Figure 2.7, we indeed find a 5' promoter that seems to be more used in leukemic cells than healthy cells when looking manually. We now perform a closer analysis. The usage of this alternative 5' promoter has been validated in a newer "Solid" RNA-seq dataset, as presented in Figure 2.8. This analysis was not performed by me. We found that the transcripts upregulated in leukemic cells are the ones corresponding to the alternative promoter (namely NM-001188180, NM-001188182, and NM-001188183). However, these counts correspond only to a Cufflinks estimate [5].

Finally, we sought to determine the cellular mechanisms regulating both transcripts' promoters. Both transcripts share the same exons, and the resulting protein is assumed to be the same, differing only in the regulation. JASPAR TF binding motifs for each are presented in Figure 2.9. We find that the promoter for the long transcript is bound by several stress response factors (ATF, AP1, STAT1/2, IRF, Nfkb, Stat5) as well as several hematopoietic factors with potential oncogenic roles (Tal1, Bcl11a, CEBPB, GATA, Runx3). As such we hypothesize that the long is involved in stress response and potentially regulated by oncogenic factors. It might be involved in cell survival and escape from cell cycle control for oncogenic cells. Conversely, the short transcript is regulated by factors such as PHF8 (histone demethylase of H3K9me2/3 and H3K27me2/3, involved in cell cycle progression by being required to control G1-S transition) and the E2F family of TFs (plays a crucial role in the control of the cell cycle, E2F6 may silence expression via the recruitment of a chromatin remodeling complex containing histone H3-K9 methyltransferase activity) and ELF1 (a member of the ETS family, is lymphoid specific). So, we hypothesize that the short transcript is regulated through cell cycle and potentially involved in cell cycle control.

2.2.3.1. Ongoing research

This latest stage of research of ATP2C1 is not being performed by me, but by José David Abad Flores, a PhD student in Salvatore Spicuglia's team, as well as other collaborators.

A CRISP-Cas9 deactivation of ATP2C1 was done by inserting a NeoR construct inside the second exon. It was observed that this impeded their growth, compared to the wild type. It was also observed that the 5' most promoter (long transcript) is is the only one than can be induced by PMA-ionomcine. Further research is ongoing.

5. D. Puthier proposed grouping the transcripts by first exon to facilitate transcript inference.

Figure 2.7. – ATP2C1 gene - (A) IGV view of mapped H3K4me3 peak coverage and RNA-seq coverage. (B) Matrix of local V scores. (C) Relative promoter usage per sample.

2.3. Articles

The attached paper entitled "Characterizing transcription factor combinatorics in cis-regulatory regions with supervised classification and sparse encoding" was presented at JOBIM 2018 in Marseille.

The paper pertaining to the alternative promoters is currently being written.

Characterizing transcription factor combinatorics in cis-regulatory regions with supervised classification and sparse encoding

Quentin Ferré[1,2]*, Salvatore Spicuglia[1], Jacques van Helden[1], Cécile Capponi[2], Denis Puthier[1]

1. Theories and Approaches of Genomic Complexity (INSERM U1090) -- Parc Scientifique de Luminy case 928, 163 avenue de Luminy, 13288 MARSEILLE cedex 09
2. Université Aix-Marseille, Laboratoire d'Informatique des Systèmes (UMR CNRS 7020) -- Equipe Qarma ; Technopôle de Chateau-Gombert - CMI, 39 ave. Joliot-Curie Fr-13453 Marseille

* Intervenant and corresponding author : quentin.ferre@inserm.fr

Keywords : cis-regulatory regions; transcription factors; combinatorics; machine learning; decision trees; dictionary learning

Introduction

Transcription factors (TFs) are a class of regulatory proteins that bind to DNA on regions called cis-regulatory elements (CRE), so as to influence the transcription of a target gene. It is now understood that TFs work in combination, by competing and/or collaborating and forming complexes (Chaudhari et al., 2018). TF binding can be studied *in silico* through the prediction of Transcription Factor Binding Sites (TFBS) using Position-Weight Matrices (PWM, Mysickova et al, 2012). However only a fraction of predicted TFBS translate to actual binding sites. Another possibility is to use ChIP-Seq experiments (Chikina et al., 2012).

The combinatorics of TFs (their combined interactions) are often studied through statistics. Most works attempt to find co-occurring TFs pairs, *i.e.* pairs of TFs whose binding sites are often found in a closer proximity than would be expected by chance (Zhu et al., 2005). Other methods include unsupervised mining of association rules (Teng et al., 2014), finding TFs with correlated nucleosome occupancy (Lai et al., 2014), pointwise mutual information (Meckbach et al., 2015), and hypergeometric probability of occurrences (Terada et al., 2013). But as a whole, existing approaches actually seek TFs associations regardless of the type of CRE), and tend to study pairwise associations instead of n-wise combinations.

TF combinatorics are also of interest to CRE detection (Kleftogiannis et al, 2016), sometimes when combined with histone marks; for example, a software tool called *TFcoop* predicts a region's cis-regulatory activity using a suite of PWM matrices' scores (*ie.* nucleotide composition) for the region as variables in a Lasso regression (Vandel et al. , 2015). While these methods focus on CRE detection and annotation, they often consider each TF (and/or chromatin mark) as an individual variable, rank them by importance, without considering the combinatorics of TFs.

Our objective is then twofold. First, we focus on to detecting TFs that are found associated to one each other. Second, we wish to uncover combinations of TFs that are characteristic of a class of CRE as opposed to other classes. We showcase our approach for different meanings of what a “class” is : whether the different classes are different natures of CRE (enhancers vs promoters), or are of same nature but with different activities (active vs inactive enhancers). We propose a machine learning approach where an example is one cis-regulatory element, each characterized by a vector of features with each feature being the fixation level of one known TF as determined by ChIP-Seq.

Methods

We use three datasets focusing on three different kinds of biological problems in the K562 cell line, respectively : TFs combinations characterizing active enhancers, TFs combinations characterizing promoters that also exhibit enhancer activity, and a general application on all types of cis-regulatory regions using public data from ENCODE.

1. The first dataset was generated in our laboratory as part of the study of TF-based regulatory networks in developing primary thymocytes, using the p5424 cell line model. In this work, CRE were selected by computing the overlap between DHS (DNAse-I Hypersensitivity Sites) and ChIP-Seq peaks for 6 specific TFs. These regions were then assessed for enhancer activity using CapSTARR-Seq (Vanhille et al., 2015). Regions were then classified in three categories proportional to tagged activity ; unsupervised clustering using *k*-means was performed according to TFs fixation, proportional to mean ChIP-Seq signals around the region’s center (± 1kb for TFs, ± 5kb for histones).
2. The second dataset is based on a systematic CapSTARR-Seq analysis of E-promoters (Dao et al., 2017). E-promoters are promoters that also exhibit distal enhancer activity. For every human promoter, enhancer activity was assayed and a vector of TF fixation was quantified by the same method as above.
3. The third dataset is created using publicly available data (ENCODE Consortium, 2012), with ChromHMM prediction of genomic regions combined with ENCODE/HAIB ChIP-Seq TF peaks in the K562 cell line. We considered a bin for each region of 4kb around its center. For each region, we built a vector where each component corresponds to a score for a given TF ; that score is equal to the proportion of the bin covered by a peak for the given TF multiplied by the score of the peak. Scores are then L2-normalized.

To highlight class-specific profiles, we use decision trees as a clustering tool. A decision tree (Chen et al., 2007) is a model that aims at grouping samples in various nodes based on several input variables. Each leaf represents a “cluster” which is as pure as possible (only composed of a single class whenever possible given the sample) given the values of the input variables represented by the path from the root to the leaf. The decision tree is used to perform a complex, combinated partitioning of the dataset. Unlike regular *k*-means clustering, this approach is supervised, allowing us to find class-specific profiles. Furthermore, different paths (with vastly different average profiles) can lead to nodes that are pure in the same class, highlighting diversity. Node splitting is performed by entropy and classes are rebalanced through oversampling. Since the decision paths only show variables

that best discriminate between the classes, TFs correlated to a discriminative one will not be visible on the decision path, so we compute the average TF profiles across all the samples in each given node/cluster, allowing us to use a “discriminative” decision tree as a clustering tool. For each node, class enrichment is computed using the hypergeometric law.

This first approach is compared to a sparse encoding of all the regions’ vectors computed via dictionary encoding : this approach rests upon the assumption that a matrix (here, our concatenated vectors) can be approximated by a sparse linear combination of special vectors called “atoms” or “words”, and seeks to find TFs combinations that are common across the entire dataset of studied CRE. Each line (or column) of the query matrix will be expressed as a combination of a single word in the dictionary, and a multiplicative coefficient (Li et al., 2012) Then, for each word in the dictionary, we analyse its usage and associated coefficients by class.

Results

On the first dataset, using our supervised classification method, we highlight complex interplay between different proportional fixations of Ets1 and Heb resulting in different enhancer activations. We also highlight the possibility of active enhancers lacking the H3K27ac histone mark, challenging the conventional view about its ubiquity (Creyghton et al., 2010). Dictionary analysis was used to study TFs combinations by class, meanwhile it analyses the *k*-means clusters previously computed. We show that there is a strong diversity of profiles per class, and that *k*-means clustering conceals this diversity; indeed *k*-means clusters enriched in Strong enhancers were found to have a similar, rather composite word usage profile.

Concerning E-promoters, given that they are usually active promoters, we compare them to a control set of promoters with equivalent activity : otherwise, we would have separated active and inactive promoters, not promoters and E-promoters. The decision tree structure is found to be onion-layered, with small, particularly class-enriched groups “peeling off” from the bulk at each step. Previous analysis by et Dao al. (2017) showcased TFs enrichment for E-promoters, but only for each individual TF. In our work focusing on TFs combinations, we find that although many E-promoters have an EP300 and JUN-rich profile, a distinct subset is enriched in YY1 instead. There exists minor variations on these profiles that we dubbed “accents”.

We are currently working on the ENCODE dataset. Unsupervised *k*-means clustering results in very impure clusters that do not exhibit different profiles, mostly grouping together regions with respectively high and low total TF fixation, although active promoters and insulators tend to regroup into a cluster of their own. It should be noted that there is a considerable number of regions for each class completely lacking in TF peaks; those regions are removed from the analysis. Further analysis is pending.

Conclusions

Our work allows us to highlight the diversity of TFs combinations profiles found within and between classes of cis-regulatory elements. It is a heuristics-based method, which can identify TFs tuples of arbitrary length. We discover both new and complex TFs combinations, but also reveal those to be characteristic of the CRE class they are found in. We are now looking to apply our method to the dataset compiled by (Muerder et al., 2018) which presents a whole-genome STARR-Seq, in order to experimentally evaluate enhancer activity across the human genome, as opposed to prediction by ChromHMM.

Our next endeavor will focus on the identification of Cis-Regulatory-Modules (CRM) using a deep learning approach. Previous work (*DanQ*, Quang et al., 2015) used deep learning with convolutional filters (CF) to classify genomic regions as enhancer or promoters, and found out that the CFs spontaneously learned correspond to many known TFBSs. We shall use a similar approach based on the distribution of ChIP-Seq TF peaks, considering for each position in the genome the presence of absence of a TF peak instead of its nucleotide (like DanQ). Then we shall analyze the filters learned by our model. A Long Short Term Memory layer should allow us to integrate positional dependencies.

References

Pedregosa *et al.*, « Scikit-learn : Machine Learning in Python » JMLR 12, pp. 2825-2830, 2011.

Dao, Lan T. M., Ariel O. Galindo-Albarrán, Jaime A. Castro-Mondragon, Charlotte Andrieu-Soler, Alejandra Medina-Rivera, Charbel Souaid, Guillaume Charbonnier, et al. « Genome-Wide Characterization of Mammalian Promoters with Distal Enhancer Functions ». *Nature Genetics* 49, nº 7 (juillet 2017): 1073-81. https://doi.org/10.1038/ng.3884.

Kleftogiannis, Dimitrios, Panos Kalnis, et Vladimir B. Bajic. « Progress and challenges in bioinformatics approaches for enhancer identification ». *Briefings in Bioinformatics* 17, nº 6 (novembre 2016): 967-79. https://doi.org/10.1093/bib/bbv101.

Vanhille, Laurent, Aurélien Griffon, Muhammad Ahmad Maqbool, Joaquin Zacarias-Cabeza, Lan T. M. Dao, Nicolas Fernandez, Benoit Ballester, Jean Christophe Andrau, et Salvatore Spicuglia. « High-Throughput and Quantitative Assessment of Enhancer Activity in Mammals by CapStarr-Seq ». *Nature Communications* 6 (15 avril 2015): 6905. https://doi.org/10.1038/ncomms7905.

Vandel, Jimmy, Oceane Cassan, Sophie Lebre, Charles-Henri Lecellier, et Laurent Brehelin. « Modeling Transcription Factor Combinatorics in Promoters and Enhancers ». *BioRxiv*, 2 octobre 2017, 197418. https://doi.org/10.1101/197418.

ENCODE Project Consortium, « An integrated encyclopaedia of DNA elements in the human genome » *Nature* 2012 Sep 6;489(7414):57-74. https://doi.org/10.1038/nature11247

Vanhille, Laurent, Aurélien Griffon, Muhammad Ahmad Maqbool, Joaquin Zacarias-Cabeza,

Lan T. M. Dao, Nicolas Fernandez, Benoit Ballester, Jean Christophe Andrau, et Salvatore Spicuglia. « High-Throughput and Quantitative Assessment of Enhancer Activity in Mammals by CapStarr-Seq ». *Nature Communications* 6 (15 avril 2015): 6905. https://doi.org/10.1038/ncomms7905.

Zhu, Zhou, Jay Shendure, et George M. Church. « Discovering functional transcription-factor combinations in the human cell cycle ». *Genome Research* 15, nº 6 (juin 2005): 848-55. https://doi.org/10.1101/gr.3394405.

Kreiman, Gabriel. « Identification of sparsely distributed clusters of cis-regulatory elements in sets of co-expressed genes ». *Nucleic Acids Research* 32, nº 9 (2004): 2889-2900. https://doi.org/10.1093/nar/gkh614.

Terada, A., M. Okada-Hatakeyama, K. Tsuda, et J. Sese. « Statistical Significance of Combinatorial Regulations ». *Proceedings of the National Academy of Sciences* 110, nº 32 (6 août 2013): 12996-1. https://doi.org/10.1073/pnas.1302233110.

Li, Yifeng, et Alioune Ngom. « Fast Sparse Representation Approaches for the Classification of High-Dimensional Biological Data ». In *Bioinformatics and Biomedicine (BIBM), 2012 IEEE International Conference on: 4-7 October 2012*, 2012. https://doi.org/10.1109/BIBM.2012.6392688.

Chen, Xiaoyu, et Mathieu Blanchette. « Prediction of tissue-specific cis-regulatory modules using Bayesian networks and regression trees ». *BMC Bioinformatics* 8, nº Suppl 10 (21 décembre 2007): S2. https://doi.org/10.1186/1471-2105-8-S10-S2.

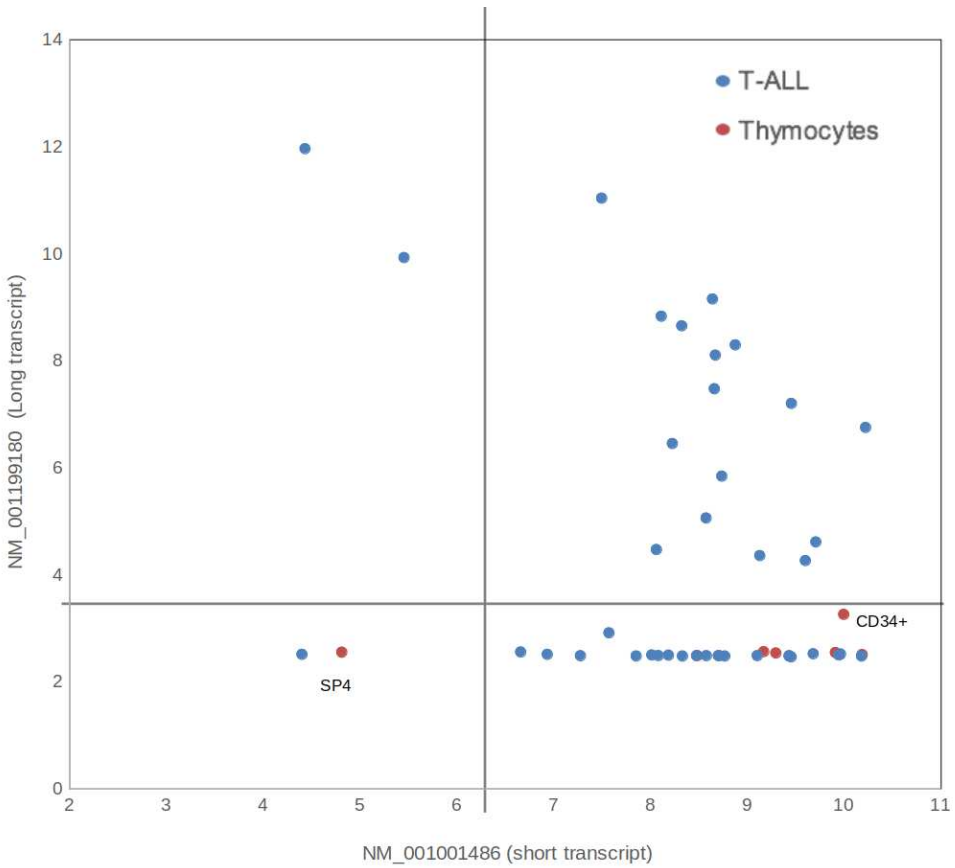


Figure 2.8. – The long transcript (alternative) is not expressed in thymocytes. However, 16/42 of T-ALL express both transcripts and 2/41 express only the long transcript.

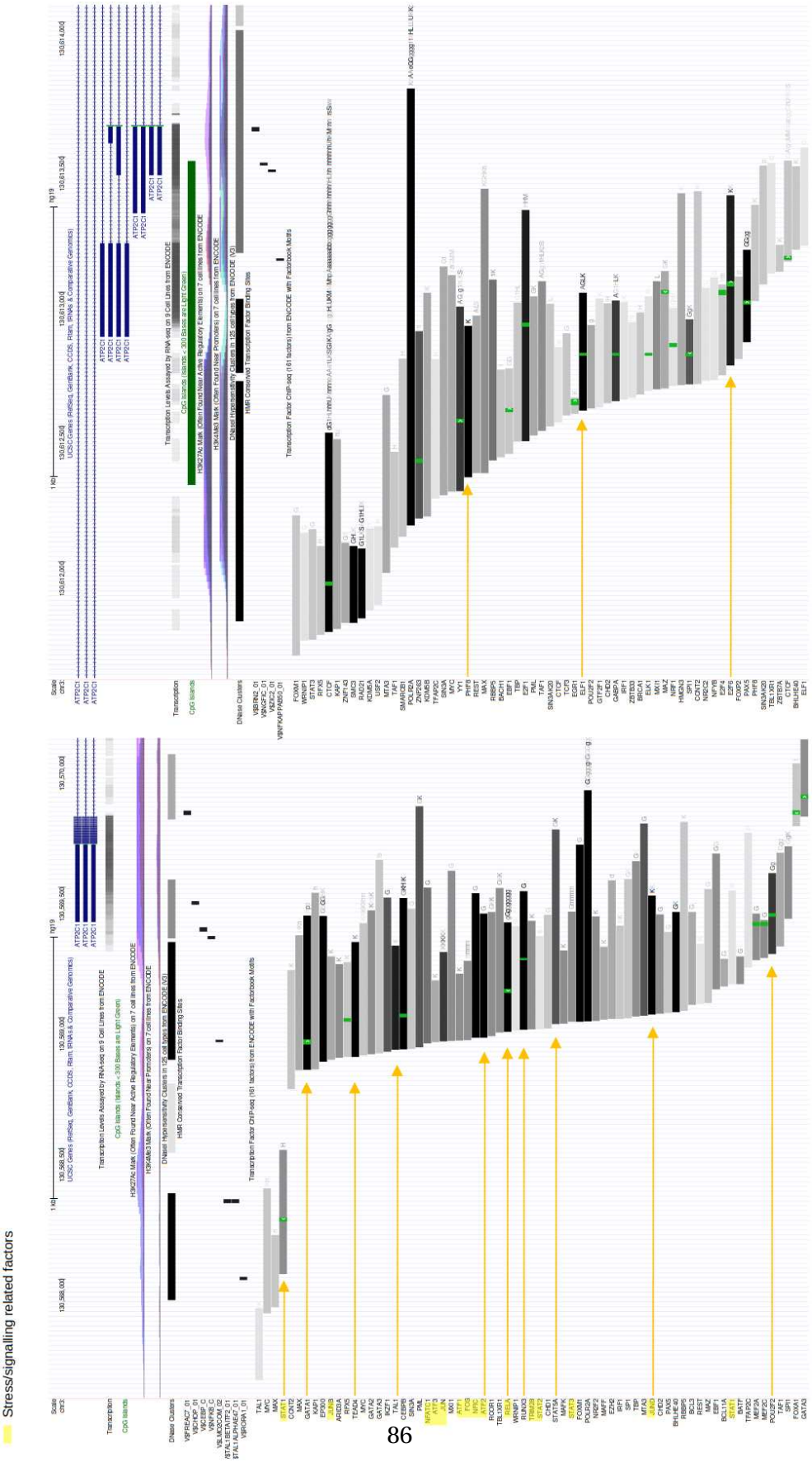


Figure 2.9. – JASPAR TF binding motifs for ATP2C1 promoters. The left is the promoter for the long transcript, the right for the short transcript.

3. Leveraging combinations for anomaly detection with *atyPeak*

Sommaire

This section presents the **atyPeak** project. Here, I endeavored to perform anomaly detection in genomic catalogues using unsupervised multi-view autoencoders. We begin with a short presentation of the underlying mechanisms of Deep Neural Networks, and the impetus behind this research. We then present the new techniques developed as part of the afferent paper, and their possible extensions.

3.1. Principle of Deep Neural Networks

Deep Neural Networks (DNNs) are powerful non-linear parametric systems in machine learning. In broad strokes, they can be envisioned as an assembly of logistic regressions, whose output is fed to subsequent other regressions. They were first proposed in the 1940s. Their stated ambition was universal learning, ie. using a single underlying principle to represent any mathematical function. As such, they were meant to mimic the only other known universal learner: the human brain.

Backpropagation was invented in 1986 (Rumelhart, Hinton, and Williams 1986), although it has predecessors dating back to the 1970s (Werbos). This allowed the networks to learn by adapting their weights. But they remained little more than curiosities until the 1990s, where advances in computing power made large scale application of DNNs possible. It is remarkable that most of the advanced architectures presented in this chapter are less than ten years old. For a review from a historical perspective, see Schmidhuber 2015. For a general review of deep learning techniques with a focus on their application on large genomic datasets, see Eraslan, Avsec, Gagneur, et al. 2019.

3.1.1. Basics

An individual neuron in an neural network is a logistic unit, which outputs a result depending on its inputs. Let us introduce some notations: a_i^j is the activation of the neuron i in the layer j and Θ^j is the matrix of weights [1] controlling the function mapping from the layer j to the layer $j+1$. The function used to calculate the result is called the activation function, with the most common ones being the sigmoid function where $h_\theta(x) = \frac{1}{1+e^{-(\theta' * X)}}$ and the Rectified Linear Unit (ReLU) function where $h_\theta(x) = \max(0, \theta' * X)$.

Layers of neurons Neurons will typically be regrouped in layers, with the input layer dedicated to representing the input data, followed by hidden layers, and finally an output layer. The general structure is presented in Figure 3.1. The principle is always the same: each neuron takes its input from the layer before it and passes them to the subsequent layer after computing activation depending on its learned weights.

The output layer will always output real numbers as values, but their signification differs depending on the problem being considered. For example, in a classifier network using a classical one-hot encoding, each output neuron is associated to a class and its output is the model's certainty that the given example belongs to this class. In another situation, the output layer could also be a tensor representing, say, an image that we want the network to rebuild based on its inputs. The possibilities are endless.

Propagation Forward propagation is the process by which the network calculates its output, based on the given input. The values of the activations for each layer are computed by applying the activation function g to the result of a matrix multiplication:

$$\boldsymbol{z}^{(n)} = \Theta^{(n-1)} * \boldsymbol{a}^{(n-1)}$$
$$\boldsymbol{a}^{(n)} = g(\boldsymbol{z}^{(n)})$$

1. A frequent alternative notation for Θ^j is $\boldsymbol{W}^j$.

. Figure 3.1 - Adapted from Sheehan and Song 2016, license Creative Commons Attribution 4.0 International.

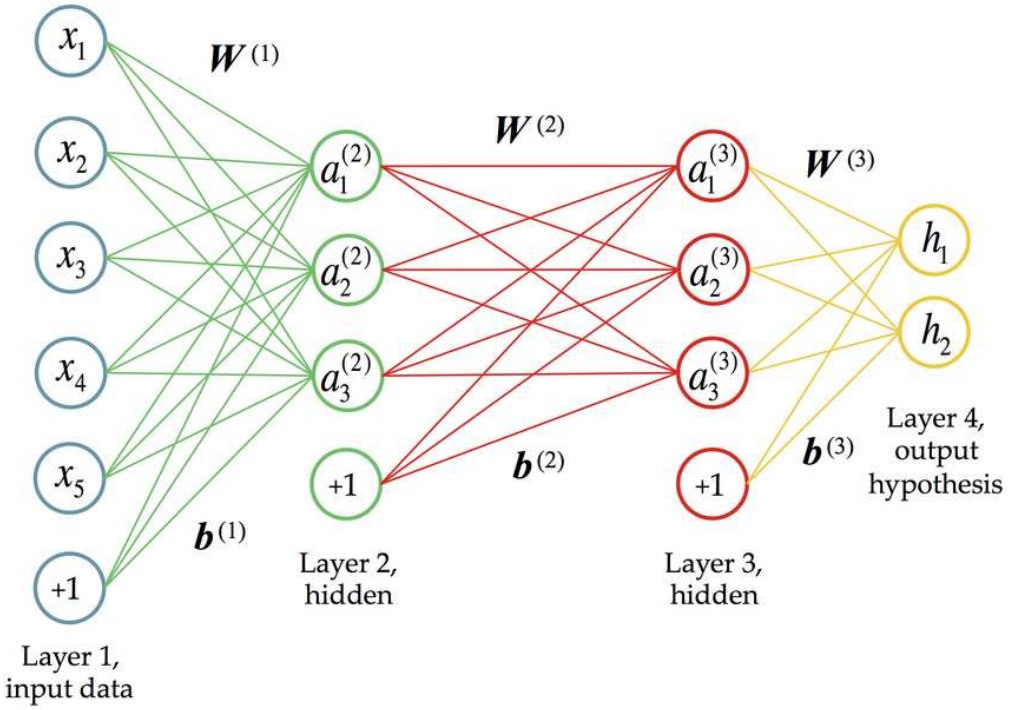


Figure 3.1. – General structure of a Deep Neural Network. The output of each layer is fed to the subsequent layers, constituting an assembly of regression. There can be as many hidden layers as desired.

Back propagation, conversely, is the process by which neural networks learn and adjust their weights. Without going into much details, as this is not the scope of this thesis, backpropagation involves adjusting the coefficients of Θ one layer at a time, from the last to the first. Let δ_j^l be the error of node j in layer l, so that for the final output layer, δ^m is the difference between desired output and actual output ($\delta^m = \boldsymbol{a}^{(m)} - y$). For the next-to-last layer, it will be the difference between the true activation and the desired activation. During a step of back-propagation, coefficients are adjusted via gradient descent to get closer to the desired activation. The process is then repeated for the layer before, and the layer before, etc. until the input layer is reached.

The gradient descent used can be customised, such as using Adam for sparse data. The choice of loss function is important, and the classical Mean Squared Error is not always the best choice.

3.1.2. Representation learning

In representation learning, a model learns representations of the input data (typically by transforming it) that makes it easier to perform a task like classification or prediction. For a general review (on which this section is based), see Bengio, Courville,

and Vincent 2014. A good representation should be smooth (small changes in x lead to small changes in the representation) and integrate information from a as-high-order-as-possible combination of the underlying features.

These conditions are generally satisfied by DNNs. Broadly speaking, Neural Networks perform representation learning as a natural consequence of the hypothesis function they learn during training. Indeed, each layer of the network performs its own embedding of the original data, based on a somewhat complicated non-linear transformation, into a new space. In the particular case of autoencoders (see section 3.1.3), we impose constraints on at least certain layers, namely that the space must be of lower dimension, making autoencoders suited for compression. This idea that a lower-dimension embedding can be an accurate representation of the higher-dimension inputs is also at the heart of the manifold hypothesis [2].

One of the challenges in representation learning is the difficulty of establishing a target for learning, since the ultimate objective is typically improving the classification of another predictor, or another far-removed objective. This is relevant in our case, since our ultimate objective is also different (see section 3.5.1, p. 101, for an exploration of this).

Ultimately, representations such as word2vec or the representations outputted by the recent BERT and GPT-2 models have been succesfully used in NLP (Natural Language Processing) tasks beyond their original encoder.

3.1.3. Specialized Neural Networks

Convolutional filters Convolutional Neural Networks (CNNs) are DNNs making use of convolutional filters in their layers. Convolutional filters can be seen as particular neurons whose input is based on a restricted section of the previous layer, where the stride of that restriction is the filter size. As such, each individual filter will learn a small, local combination of elements. The subsequent last dimension in convolutional layers' tensors is the number of filters in said layer. See Figure 3.2 for more details.

A convolutional product serves to highlight how much a given region of the image "matches" the filter. The purpose of such networks is to learn *local* combinations of elements across the dimensions covered by the filter. Convolutional layers are usually followed by pooling [3] layers, aimed at reducing the dimension of the previous layers. This is usually done by making local averages.

2. For example, if considering black and white images of size 64x64, only a fraction of the 2^4096 possible images would be naturally occurring images. This would manifest in complex correlations between the pixels, with some leeway due to noise. This means a lower-dimensional manifold is sufficient to describe the space of naturally occurring images.

. Figure 3.2 - Top is adapted from "Deep learning for complete beginners", Cambridge Coding Academy, https://github.com/PetarV-/TikZ/tree/master/2D%20Convolution. Bottom is adapted from Aphex34, English Wikipedia.

3. Sometimes also called subsampling layers, since they effectively reduce the dimensionality by mashing local elements together.

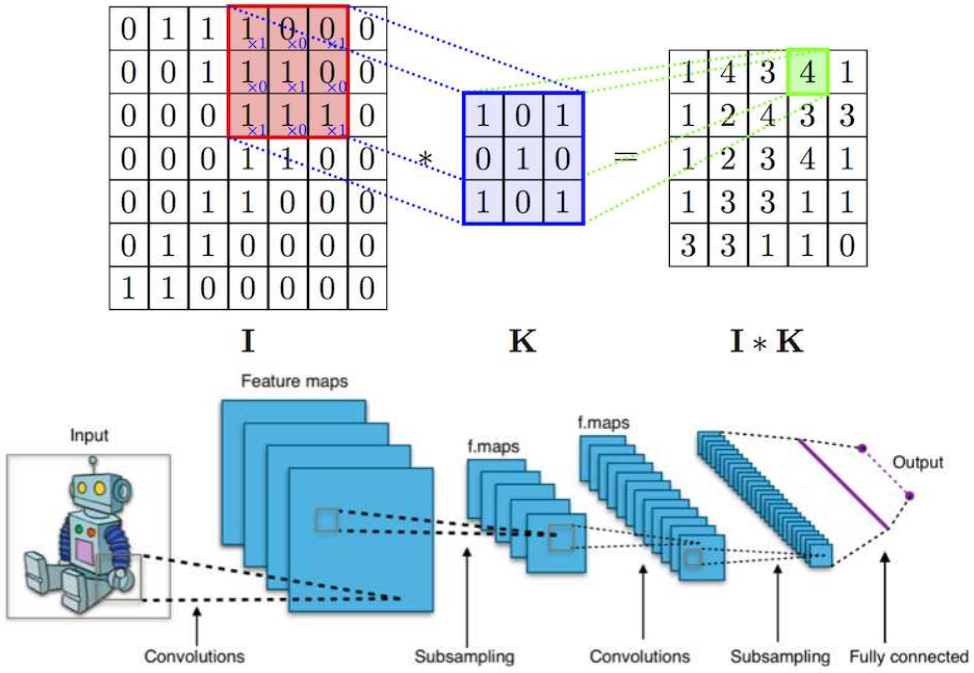


Figure 3.2. – Convolutional Neural Networks. The top part presents the operation of convolution. The representation of an input tensor, as seen through a convolutional filter, gives the closeness of the match between the local pattern and the convolutional filter. The bottom part presents the usual structure of a convolutional network, several convolutional layers interspersed with pooling layers, followed by layers to treat the information.

There also exist graph convolutions, which follow the same general principle except that they consider a node's neighbors in the graphs (nodes with connecting edges) instead of its spatial neighbors.

Autoencoders Autoencoders are a specific type of architecture for neural networks. Their goal is to learn a compressed representation (ie. an encoding) of the input data with a lower size (in bits). Following this compression, the model learns to rebuild the original input data based on the encoded dimension [4]. It seeks to minimize the difference between the input and the rebuilding, hence the name. This compression entails discarding signal noise, as was discussed in section 1.4.3.2. See Figure 3.3 for their general structure.

Variants to this basic model exist. For example, denoising autoencoders will instead be trained to rebuild a clean image from a "salted" version of the image, obtained by adding noise to the clean image. This is not relevant for our particular usage because, as we explain later, we did not have access to clean data in the first place. Another

4. The encoded dimension is sometimes referred to as a *latent*, or *hidden* mathematical space.

popular variant is the variational autoencoder, where the encoded dimension instead represents the moments/parameters of assumed underlying random variables.

Autoencoders can be used to learn useful representations (by extracting the encoded dimension), as generative models, etc. As such, they have been used in a wide variety of tasks, from facial recognition to natural language processing. Since the encoded dimension is usually smaller than the input tensor, a lynchpin application of theirs is *dimensionality reduction*.

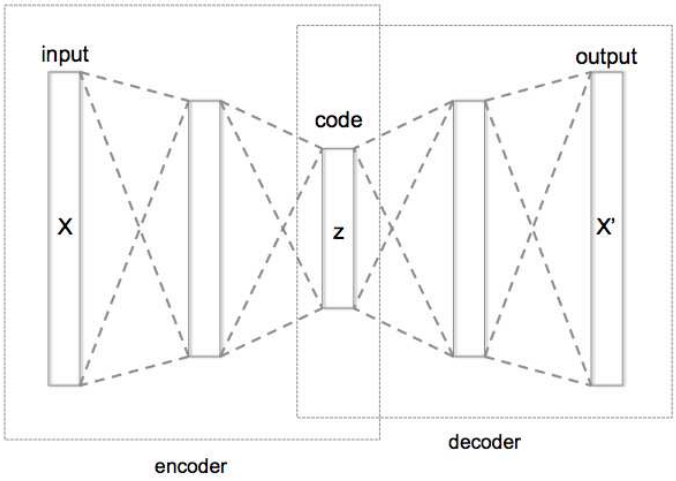


Figure 3.3. – Structure of an autoencoder. Its goal is to learn an encoded representation sufficient to closely rebuild the original input. In this figure, $\boldsymbol{z}$ is the encoded dimension, $\boldsymbol{X}$ the input tensor and $\boldsymbol{X'}$ its reconstruction by the autoencoder.

Advanced elementary structures such as convolutional filters, LSTM and whatnot (see later) can be used with autoencoders, since an autoencoder is an architecture *type*, not a building block *per se*. For more information about autoencoders (and convolutional networks) and their applications to similar problems, please see the relevant Methods section of the attached atyPeak paper.

Other advanced architectures The family of Deep Learning models is very vast, however, and rapidly evolving. The list of architectures is endless, and keeps growing. [5]

Some structures allow the model to focus on specific parts of the previous tensors. For instance, LSTM (Long Short Term Memory) cells are used for language processing

. Figure 3.3 - Adapted from Chervinskii, English Wikipedia

5. If you are interested in learning more about existing architectures and components, I would recommend visiting https://www.asimovinstitute.org/neural-network-zoo/ which, as of writing, does an excellent job of presenting the various members of the Deep Learning family.

and are part of the family of Reccurent Neural Networks, characterized by intra-layer connections. In terms of architecture, they are used in a distributed manner where each cell considers the value of its input (previous layer) tensor for one time stamp [6] at a time. They are themselves made of smaller Neural Networks. However, each LSTM cell also has access to the output of the previous cell in the chain. They further introduce forget gates to discard this previous information, which helps with the exploding and vanishing gradients problem. More recently, attention mechanisms have been proposed, where each neuron is granted access to another intermediary layer consisting of the outputs of the previous layer (or previous states for NLP models) multiplied with a learnable mask of attention weights, to help the neurons focus only on relevant parts of a sequences. More modern NLP models like the *Transformer* use attention exclusively.

This notion of custom connections has been used in other architectures such as the U-net. This is a model that consists of two branches, a contractant branch followed by an expansive branch. While the layers in the contractant branch have only access to the outputs of the previous layers in their branchs, the layers in the expansive branch also have access to the output of certain layers in the contractant branch, allowing them to access some informations from the first steps of the processing.

Models can also be combined, such as with Generative Adversarial Networks, where two models are in competition: a generator network is trying to generate examples to fool a discriminator network, tasked with determining whether an example is from the real data or was generated by a network. The goal is that this competition will help both models learn well, but necessitates that each has a reasonable baseline efficiency.

More recently, capsule networks have been proposed for image processing where certain neurons are connected with multiple weights (a vector) instead of just one weight (a scalar). In other words, they are grouped in *capsule cells* returning an activity vector. As a result, they can transfer more information, such as transferring a feature's color, deformation, etc. along with simply its nature.

Finally, we ought to mention that regularisation (for both activity, and kernel values when relevant) can be applied, and that more generally the loss can and should be customized to impose any penalty of the activities one might desire, for example by enforcing sparsity in encoded dimensions for an autoencoder.

Information flow between parts of a Deep Learning model From what can be learned from this short panorama of Deep Learning methods, I would like to put a particular emphasis on the notion of *information flow*. In deep learning, different inputs can be combined and split as one might desire. The key notion is that connections between layers in general allow for information flow between parts of the model, which will process the information it was given with each successive layer. This is especially relevant when using DNNs in multi-view application. Indeed, I would

6. As with our tensor representation, the "time" axis can be a spatial position axis. It just means there is one axis designated as the position axis for a timeseries-like data.

argue that modern new DNN architectures are built by combining those pre-existing structure in a new manner to create a structure with a tailored flow of information, as exemplified by the creation of LSTM, U-nets, attention, etc.

3.2. Impetus

The impetus behind this project was the realization that collating an ever-increasing number of experiments, each of which have a certain probability to contain errors at each given position, brings an ever-increasing number of total errors. By extrapolating this argument, we can suppose that an exhaustive collating and compiling of the cyclopean amounts of data available would be a nightmarish undertaking, necessity rigorous reprocessing. Fortunately, we shall wonder no more: Jeanne Chèneby did exactly that, and realized that indeed the ReMap catalogue now covers significant proportions of the genome with at least one ChIP-Seq peak, notwithstanding the filtering methods she applied. For instance, ReMap 2020 covers 34% of the genome with 5 peaks or more and 65% of the genome with one peak or more. This realization spurred a collaboration between me , Jeanne Chèneby [7], and Benoît Ballester.

To paraphrase J. Chèneby, this is not surprising since ReMap integrates very hetereogeneous data with different protocols, proteins of interest, biotypes, etc. Furthermore, ChIP-seq experiments produce errors, as we copiously explained in section 1.3.3.1 (p. 1.3.3.1). Since there are so many experiments integrated, there is a high likelihood at any given position on the genome that at least one anomalous peak will be present. In vanilla ReMap, this is alleviated by rigorous quality controls, and through the sheer number of integrated experiences which permits a sort of majority vote increasing confidence in regions with a large number of peaks. Indeed, to paraphrase her, ReMap's root goal was to identify Cis-Regulatory Elements through ChIP-Seq data. Only CREs containing peaks with large number of peaks, presumably from multiple independent experiments, would be considered true CRE.

But is it possible to go one step further in this reasoning and take it to the level of **each individual peak**? Since epigenomic marks tend to work in collaboration (another point we have emphasized *ad nauseam* in section 1.1.3) we expect that ChIP-Seq peaks from regulators that are known to collaborate biologically would be correlated. Similarly, we may observe that two different datasets tend to be correlated. In either case, lonely peaks would be suspicious [8].

Relatedly, ReMap already incorporates such a notion of strength in numbers, as the *non-redundant peaks* are supported by many fixation events from experiments in different laboratories. But since the integrated data can be very heterogeneous (different protocols [9], ...) this is harder to apply. This is why we sought to leverage

7. She was, at the time, a PhD student in the TAGC lab; but has since graduated and moved on to greener pastures (of data)

8. In this context, completely different experiments can be seen as biological replicates if they target the same epigenomic regulator.

9. For the most recent versions ChIP-exo and DAP-seq is used. This data was not used in atyPeak,

the combinations between sources, since in this case the heterogeneity of the data is a non-factor: the approach would simply learn that "A and B correlate" or "C and D do not correlate", regardless of the nature of A, B, C or D, and does not imply they represent exactly the same event at all times.

Using the methods championed by ENCODE (presented in section 1.3.3.1) would amount to classical statistics, such as the Pearson correlation. This, however, lack resolution as those methods generally only considers pairwise correlations instead of n-wise, and compare entire datasets instead of individual peaks. Furthermore, we lack any sort of supervision or database of "clean" peaks to compare against or train a model with. This is akin to the more general case of database curation, where manual review or labelling of the data is usually very expensive or downright impossible. The only resource we can leverage to perform curation are the aforementioned combinations, which lands us square in the domain of **unsupervised anomaly detection** (see section 1.4.3, you know the drill by now). Due to the massive amounts of data involved, and the complex correlations to be learned in the data, Deep Learning seemed appropriate.

Additional details about the problem impetus and the reasons behind our choice of approach can be found in the introduction of the atyPeak paper.

3.3. Adapting network architecture to the research question

To do so, I elected to use an *autoencoder* to perform a lossy compression of candidate CRMs [10]. The idea would be that the model would learn sources (TR, dataset pairs) as groups of correlating sources and not as individual ones, losing *details* in the process. The details lost would be the peaks that do not respect the usual correlations between sources, which are likely to be anomalies.

I would invite the reader to see the attached atyPeak paper for additional details.

General architecture As *atyPeak* is designed to perform a compression, we use comparatively small and simple models that do not require large amounts of CPU resources. This is in accord with my personal philosophy when it comes to Machine Learning, which is to start small and work upwards in terms of complexity. I find that gigantic models are often overfitting their data, and are tantamount to brute force approaches, with the corresponding lack of interpretability. As such, the autoencoder used has only one layer of convolution per dimension (see below) and only a few deep layers. For more on how to choose the dimensions of the layers, see section 3.5.1 (p. 101).

but this notion of *different protocols* can be extended to them too, and still used by atyPeak, as we explain later.

10. A CRM, or Cis-Regulatory Module, is a region on the genome that contains at least one candidate CRE, Cis-Regulatory Element, which is a local element.

3.3.1. Transverse successive convolutions

The tensor representations we used, as presented in section 1.4.1.2 (p. 55, have 3 axes: genomic position, ID of the dataset of origin, and name of the Transcriptional Regulator considered (or more generally regulator ID). See Figure 1.19, at 55. We did not process the entire genome: instead, each tensor given to the model represents a candidate CRM, among the 65K with the most peaks across all cell lines.

Today, most of the literature on Deep Learning is focused on images, and multimedia in general. Hence, even the approaches recently developed such as capsule networks fit the constraints of analyzing an image represented as a matrix (with the values corresponding to RGB colors). In an image, however, the spatial dimensions (X and Y) have an ordered meaning and proximity of two items in the matrix is important. In the tensor representation we propose, there is only an ordering in the X axis (genomic position), unlike the Y and Z axes. Instead, datasets and TRs are sorted alphabetically, and as such there is no greater association between two datasets that are neighbors in the tensors that there is between the first and the last.

However, we did use some structures that are originally designed for image processing, such as convolutional filters. To resolve this paradox, I used fully transverse filters for the Y and Z dimensions. As a result, the filters consider the tensor in its entirety for those dimensions, negating any proximity influence. For the X dimension however, where there is an importance to the ordering, the filters are short (dozens of base pairs) as is usual. Instead of using 3D filters, I used successive convolutions for the two different axes, convolving across the Y and then the Z axis. This was done to alleviate training problems observed with the larger 3D filters, but it also serves to establish a hierarchy by first learning combinations of datasets, and subsequently learning combinations of TRs on latent variables representing combinations of datasets, in this order. This can be linked to a middle fusion approach.

Squishing While the tensor representation cover windows of 3200 base pairs, I downscale the tensor by a factor 10 along the X axis ("squishing") since the data has low granularity along that axis: the peaks are long, so we can afford to have a resolution of 10bp only without losing much information.

This also helped by diminishing the computing costs, since the convolutional filters could be made 10 times shorter. Beyond simply speed, the longer filters often failed to converge, a problem which disappeared with the use of shorter filters and is potentially linked to the sparsity of the tensors (see below).

3.3.2. Crumbing and countering sparsity

A sparse tensor is defined as a tensor where most of the elements are equal to zero. Exactly how many varies depending on the definition. Sparsity in the data is a challenge in Machine Learning. Note that having sparse data is not the same thing as having a sparse model. In the latter case, it means the vector of parameters and/or

internal vectors/tensors representations are sparse [11].

Sparsity can also be due to missing or incomplete data [12], a different but related problem. Sparsity can also mean that you have few representative examples for each class. Finally, as is the case here, the data representations itself might be sparse.

Fundamental consequences of sparsity The cornerstone of fitting Machine Learning is efficient computing of a function's gradients. But in practice, sparse data tends to be stores in very large matrices, and the zeros of those matrices are non informative. This necessitates the development of different methods to efficiently compute the gradients of sparse data [13]. Furthermore, the curse of dimensionality [14] is an even more pronounced problem.

More generally, sparsity dilutes the informative variables. When a model is trained on sparse data, the proportional information content is lower and results in a model that is much less stable, meaning that small variations in the next training examples will have a high impact on the learned parameters.

Model behavior on sparse data LASSO (Tibshirani 1996) and other regularizations are seen as effective at dealing with sparse data, since the regularisation will discard the useless zero variables. Indeed, in a logistic regression with regularisation, the resulting vector of weights θ will be sparse.

When the sparsity is due to missing data (like an user forgetting to enter data), some algorithms will handle it natively, while others require probabilistic imputation to resemble dense data (Alasalmi, Koskimäki, Suutala, et al. 2015). For example, Naive Bayes simply ignores missing data, since it works by inferring Bayesian rules (X. Li, Ling, and H. Wang 2016). However, algorithms such as decision trees of SVM will use probabilistic imputation to fill missing values, usually through mean mode imputation (Josse, Prost, Scornet, et al. 2020). The convergence behavior of classification algorithms with sparsity was explored by X. Li 2017.

More general data augmentation techniques to fill the missing data have also been proposed, such as the use of SMOTE, or Generative Adversarial Networks to create new data. Another possibility for sparse data in general is to use feature hashing or matrix factorization methods to obtain a dense representation of the sparse matrices/tensors.

Sparse data in deep learning More specifically, Deep Learning models have difficulties with sparse data. For large tensor-processing architectures such as CNN (or RNN), the model relies on spatial attributes of the data to learn. If the data is highly

11. Unlike sparsity in the data, sparsity *in the model* is often sought out and considered beneficial.
12. We could argue that this is also relevant here for the ReMap data to some degree. Some datasets may be missing information, depending on how we group them (ie. if an experiment for this TF had been carried by this laboratory, it would have shown a peak.
13. We suffered partly from this and alleviated it using squishing, see section 3.3.1.
14. When the data is so large and has so many variables compared to the number of samples that there exist many spurious rules translating the variables to the class of the samples.

sparse, the network may learn the zeroes as the commonality. Coincidentally, the use of encoder-decoder (ie. autoencoder) models combined with a sparse training strategy where the density is adjusted during training has been found effective at countering this [15] (Jaritz, Charette, Wirbel, et al. 2018).

More generally, in deep learning the activation function that we wish to approximate should be as smooth as possible over its domain [16], with soft gradients. When the data is sparse, not only do we not have enough points to fit on, but we are more likely to miss minima and maxima because the gradient descent never brought us there. Another related problem comes from the fact that our tensor representations can indeed have a step-wise behavior, alternating between a continuous line of zeroes and a continuous line of ones [17] and that values at the same position in different tensors are also binary/form a step difference and do not vary smoothly either.

Crumbing When working on atyPeak, the sparsity problems did not manifest at first on the dense artificial data, but the real representations are clearly sprase, since at any given position only a fraction of all possible TRs will be present, and not all datasets contain peaks for all TRs.

Empirically, I observed that using the fully sparse data would often result in the model being unable to learn even a partially correct result, and only rebuilding completely empty tensors. This is not surprising, as autoencoders focus on rebuilding the average observations. If most of the data is zero, gradient descent is hard and the necessary improvements are seen as too marginal to be conserved in the next iterations. I used the Adam gradient descent, which aims to permit working on sparse data by using smaller, adaptive learning rates Kingma and Ba 2014. However, Adam alone was insufficient.

As a consequence, I developed and used a method I called *crumbing* [18]. This consists of adding, for each non-zero value x in the original tensor representation, a small $\frac{x}{10}$ value at all positions along the Y or the Z axis, forming a cross pattern centered on the original value. See details in Methods. These values do not represent new data, their purpose is to prevent the model from learning only zeroes and direct its attention towards the regions of interest. It is properly accounted for when calculating the anomaly score, where we divide by the total value in the crumbed tensor.

I observed that this seemed to solved the learning difficulties of the model. Quantifying the exact effect of crumbing as a function of sparsity in the original data would be an interesting perspective. For similar sparsity reasons, I used a large batch size (48 samples per batch minimum) to smooth the gradients and avoid the *batch effect* [19],

15. This density tweaking is reminescent of the crumbing we implemented (see below) where the density is increased at selected positions on the tensor.
16. The set of inputs we wish to approximate it over
17. As opposed to a smoother evolution along the time axis, such as $\begin{pmatrix}2 & 3 & 4 & 4 & 5\end{pmatrix}$.
18. "Crumbing" as in "leaving a trail of bread crumbs".
19. Where the model skews exaggeratedly its coefficients depending on what it saw in the latest batch, never converging to the true average

which as we explained is a problem with sparse data.

3.4. Anomaly based on the absence of known collaborators

The end result of our approach is that each peak gets a score from 0 to 1000. This score represents whether *its gregariousness is lower than average*. Broadly speaking, this meant that if the sources A, B and C are correlated, the model will not learn them individually but instead learn an "$\{A, B, C\}$" brush for the entire combination γ. Recall that a source refers to a (dataset, TR) pair. For example, (ENS12345, CTCF) is a source designating the peaks for CTCF in the dataset ENS12345. Going back to our example, if at a given position there is only a peak for one of these three sources, the entire ABC group will be rebuilt with a value of $\frac{1}{3}$ for each source. In this case, the added B and C peaks are called *phantoms*.

The score given to each peak corresponds to the reconstruction error in the final tensor. If we did not apply the normalizations presented below in section 3.5 (p. 100), a score of 1000 would mean *all* collaborators are present. So, we normalize by considering how many of those collaborators are present on average. If, say, A is correlated with B and C but on an average CRE there are only 2 of the 3 present, then an $\{A, B\}$ situation gets a score of 1000, and is seen as perfectly normal. Conversely, this means that the peaks which get a low score, and as such as seen as anomalous, are peaks whose correlation group (at their position) is less complete than it is on an average CRM. More details are presented in section 3.5.2 (p. 102).

Limitations There are some limitations in the nature of the correlation groups that can be find. For instance, overlapping groups are possible (ie. learning an ABC group and BCD group sharing B and C) but hard to reach in practice, as we explain in the paper. Furthermore, sources that are usually rather lonely (not particularly correlating to other sources) tend to be learned as their own groups. This can be a problem, because there is a possibility the model may instead see those lonely sources as the anomalies we want to remove. This can be alleviated by using a weighted loss: we show in the paper that sources with a higher weight in the loss will be focused on and learned as more precise groups[20]. The proposed normalizations (section 3.5) also help with the problems presented in this paragraph.

In any case, generally any judgment made by atyPeak is made in *probabilistic* terms. Without supervision, we cannot say with certainty which is indeed due to noise or to any particular source of anomaly. Instead, we rank the peaks by *plausibility*, depending on how many known collaborators they have, giving them a score. The score threshold to be used, and even what it means for a peak to be *anomalous*, will change depending on the user's needs (see section 3.6).

20. The recommended usage is to increase the weight of the rarer sources.

However, I would argue all the limitations presented here are inherent to any unsupervised approach, and that we cannot do better without supervision.

3.4.1. Artificial data for approach confirmation

To validate this approach, I used artificial tensors representing artificial CREs. Details about the generation of those tensors can be found in the Methods of the paper. Their usefulness stems from the fact that the correlation groups formed by the artificial sources are perfectly known and precisely controlled. Indeed, we know exactly that the source A correlates with B and C, but not D and E and F, for example. As a result, we know that if the model produces phantoms for D when A is present, it is an error. Noise that does not respect the usual correlation is then randomly generated and added.

All phenomena described here, such as group completeness being the main factor determining the score, overlapping groups, independence of score from group abundance, etc. were first assessed using artificial data. This is paramount, as we lack ground truth for the biological data and as a result cannot calibrate our approach. Furthermore, having demonstrated our model on artificial data allows us to go further and say that the *atyPeak* approach is applicable to all problems that can be modeled in a similar manner. The necessity to use artificial data to validate bioinformatic approaches is starting to be emphasized in the literature (Daber, Sukhadia, and Morrissette 2013).

3.5. Proposed normalization techniques for black-box models

The normalization techniques presented here were a necessity as we used the autoencoder model for anomaly detection, which necessitates a degree of interpretability, and is a departure from their usual applications (compression).

I would point out that those techniques do not attempt to understand the inner workings of the model. Indeed, they work by creating an input tensor with certain characteristics and studying the output returned by the model, compared to this input tensor. I believe this is a strength, and makes them applicable not only to all DNNs but more generally to *all black box models* (where the inner workings are poorly understood, be it in ML or otherwise) and to all models making a compression to a lower dimensionality space with hidden variables.

Background on the interpretability of ANNs Interpretability of DNNs is an active area of research. They are commonly considered black box models, meaning we have little to no control or readability on their inner workings, and must rely on tuning the parameters during training until the output resembles the desired output. The standard interpretation method consists of picking a layer and generating, for each

neuron of the layer, an artificial input example that would maximally activate this neuron using gradient ascent [21]. Some examples of this approach are presented in the paper. This is also applicable to more exotic components like LSTMs. Another method is to directly look at the learned weights, but this is only informative on simple models or when looking at the weights of special layers (such as attention layers).

3.5.1. Q-score for model evaluation

Hyperparameter choice is a significant problem in DNN design. In this context, *hyperparameter* designates variables such as the number of layers, the number of neurons or filters in each layer, etc. As a result, the field of meta-learning has emerged, applying machine learning algorithms to the metadata and hyperparameters for ML experiments. The basic principle is to consider the hyperparameters as the arguments of a function to be optimized, which outputs a quality score depending on them. However, this requires formulating a relevant score function.

Classically, when working on images, the criterion is often the rebuilding quality [22] (Ordway-West, Parveen, and Henslee 2018). Here, however, approaching a lossless compression is not the objective. We instead have what I would call a Goldilocks objective of compressing "just right": we want to lose the details which do not match regularities in the data, because they correspond to the anomalous peaks we want to flag, while while maintaining a reasonably accurate compression.

So, we need a different meta-learning objective. To that end, I introduced the Q-score (short for Quality score) which focuses on *conservation of correlations*. For two sources A and B, the presence of A at a given position should affect the score of B **if and only if** A correlates with B. This is a logical proposition: the Q-score simply evaluates for all A, B pairs if it is true. If it is false, it assigns a penalty based on the relative abundance of A and B. A more precise formula is given in the paper. The goal is to reach a model with small penalties, so as to properly learn the correlations present in the data. But the correlating sources should be learned as groups, instead of over-learning the sources as individual components.

The current formulation of the Q-score is as a sum of binary terms, and as a result does not have a continuous derivative. A continuous version was being considered and has not been completed due to lack of time. This makes an analytical optimization complicated and forced me to resort to a manual grid search. Relatedly, a lack of available computing resources forced me to make it coarser than I would have liked, with a bigger emphasis on manually tuning the parameters and instead confirming my choices with the Q-score.

21. Just like gradient descent, but we maximize a function instead of minimizing it.
22. aka. the reconstruction error

3.5.2. Normalization of correlation groups

This normalization stems from the following realization: *an autoencoder will tend towards learning the average*. This is touched upon in section 3.4 (p. 99). This means that, in practice, the value of certain sources will tend towards an average if they have been learned as part of a very large group, a group which is usually at least partially incomplete in the real data.

Making it more concrete: if A has been learned as part of the $\{A,B,C\}$ combination/group, the value given to a peak of A in the rebuilt tensor will be equal to 1 if and only if A, B, and C are all present. In practice, this is very rare, which means the rebuilt value of A will peak at, say, $\frac{2}{3}$ if on average two out of the three are present at a given position [23]. The goal of the normalization is to counter this.

It should be noted that it is not merely a matter of cardinality. This must be pondered by the weight given to the source in the learned group: while we have given simple examples, groups are learned as something resembling $\{0.2\times A, 0.5\times B, 0.7\times C\}$, due to factors explained below and presented in the paper, but the general gist is the same. Furthermore, overlapping groups must be accounted for: if we have the groups $\alpha=\{A,B,C\}$ and $\beta=\{C,D,E\}$, then C receives contributions from α and β and this must be pondered.

The detailed normalization procedure is presented in the paper. It is done source by source, and returns a different coefficient for each. It basic principle is to prepare an input tensor containing a peak only for the considered source, and look at the phantoms added in the rebuilt tensor. Broadly speaking, it consists of three steps, corresponding to three different biases:

1. **Intra-group bias**, where different sources within the same correlation group are biased due to relative abundance differences within the group, or a bias in learning (usually, a too high learning rate). This is corrected through the difference between the sum of the original and rebuilt tensors.
2. **Inter-group bias** as described in the previous paragraph. Recall that the rebuilt value of a peak (value in the rebuilt tensor) is proportional to how complete its correlation group. Since the groups have different sizes, our goal is instead that peaks get the same score when their group's completeness relative to its average completeness it the same. This is corrected through a Monte Carlo approach by iterating over a subset of all CRM representation and calculating the mean occupancy (ie. completeness) for this source's group. Its group is estimated simply by looking at all the phantoms added in the rebuilt tensor.
3. **Overlapping group bias**, where if a source belong to several groups, phantoms from them can accumulate and will not be seen at step 2. This is corrected similatly to the inter-group biais, but we evaluate instead how much all the sources that are not present in the correlation group can contribute to the phantoms, in the average case.

23. Actually,

The principle of estimating the correlation group a source belong to by looking at the phantoms added in the rebuilt tensor is important for interpretability, since it means the correlations groups can be deduced. However, due to the overlapping groups problem and more generally the complex non-linearity of the model, it is recommended to consider this heuristically. Meaning, a source B may not appear as a phantom for the source A, but A may appear as a phantom for B, so all sources should be considered, like we do in the steps 2 and 3.

Finally, I believe the normalization techniques presented here would be useful to any problem leveraging autoencoders for anomaly detection based on the respect of combinations, and more generally useful to any autoencoder black-box approach.

3.6. Biological interest

Confirmation of biological meaningfulness To confirm the relevance of our approach, the results of atyPeak were cross-referenced with known biological correlations.

The first confirmation presented in the paper involves comparing the scores given to peaks for certain TFs when biologically known correlators are present. This is the same elementary procedure as the Q-score. We show further evidence of unidirectional influence with the case of CTCF: factors which are always found with CTCF will get a worse score when CTCF is absent, but conversely CTCF can be found independently so its score is not as affected by the absence of other factors.

We performed an additional confirmation through a comparison with the ReMap 2020 update (the 2018 update was used when training atyPeak). There are two relevant observations: first, peaks with a bad score in atyPeak 2018 tend to be less confirmed in 2020, meaning the number of peaks for the same TF in ReMap 2020 tends to be lower. This effect is marginal, but very real, as peaks that had a abysmal score ($s < 250$) will very rarely be replicated in ReMap 2020. However, for peaks with a higher score, this replication coefficient can vary[24]. Second, when considering CRMs as a whole and not peaks, we find that CRMs with low average score will instead be confirmed more in ReMap 2020; our interpretation is that this does not mean that the CRMs were full of erronerous peaks, but rather that there should have been peaks present for the missing correlating TFs that have been added in the 2020 update.

Relevance for biological studies In conclusion, the atyPeak approach is of interest to the identification of Cis-Regulatory Elements. This requires high-confidence estimation of the Transcriptional Regulator Binding Sites. Through atyPeak, it is possible to at the very least flag those sites that are considered suspicious, resulting in an

24. We must remember that not all experiments have been replicated a constant k times in the new datasets integrated in ReMap 2020. By which I mean it is very possible that a peak got a good score but the corresponding TR was not replicated many times when totalling all the new datasets, leading to a lower *replication number*.

increased confidence in the CRE estimated through high-quality (as ascertained by atyPeak) peaks.

CREs with high-scoring peaks are assumed to be correctly characterized. Unfortunately, a low average score may be due to either the presence of anomalous data, or conversely the lack of peaks for a correlating source that was supposed to be present, and we cannot determine which without supervision.

For the peaks themselves, the take-home message is that our method signals *peaks that have less correlating sources present than they usually do*. Those anomalous peaks can be errors that one wished to remove, but they could also be specifically sought out, for example as potential oncogenic mutations. The usual correlation groups used to make this determinantion are learned by the model itself, and our normalization helps correct for the variety of possible groups.

In conclusion, atyPeak is an unsupervised *side-channel* approach, in that it allows us to bypass the need to perform a supervised curation for each and all type of noise. Indeed, the lack of supervision was an interesting challenge.

3.7. Perspectives and extension of the approach

Multi-omics As discussed when presenting our tensor representations in section 1.4.1.2, this approach can be applied to any data that can be represented in a similar tensor. While the X axis should still represent a genomic position, the Y and Z axes do not necessarily need to represent "dataset/experimenter ID" and "TRs". It is possible to extend this to any multi-omics approach. For instance, at one step in the project we considered merging all peaks for all datasets for a given TRs, and instead comparing the bidnings across different cell lines, so that the axes would be respectively "TRs (merged)" and "cell line".

Non-binary data The approach was first developed to work on binary data, where a 1 signals the presence of a peak and the 0 an absence. This facilitates interpretation and the calibration of the normalizations presented above: if the only possible values in the original tensor were 0 and 1, the meaning of a value of 0.5 is immediately apparent.

However, my original idea was to have a value of 1 only at the peak center, and decrease it according to a Gaussian law over the distance. This part of the the reason why the anomaly score formula includes a division by the value at each position in the original tensor[25], the other part being crumbing. Such non-binary data could be used now that the techniques have been calibrated, but it would complexify the interpretation.

Narrower peaks As discussed in section 1.3.3.1, the true binding site for each Transcription Factor is likely only a few base pairs long. However, ChIP-seq peaks are much

25. If the only possible values in the original tensor are 0 and 1, such a division would be redundant.

longer than this. This apparent drawback is actually leveraged by atyPeak: since the model only used convolutional filters, a value of 1 for the margins of the ChIP-Seq peak can be instead taken to mean that there is a TF binding in the vicinity. This idea of over-extending beyond the biological event is related to an idea presented later, where one simply extends (ie. *slopping*) the regions of interest and quantifies the statistical enrichment depending on the slop (see section 4.4.5).

If we were to use methods such as ChIP-exo, where the peak is centered on the (short) true TFBS, it would become necessary to add a temporal integration layer to atyPeak, as is explained in the paper. But I do not believe this calls into question the fundamentals of the approach: the goal would merely be to allow the model to access again this "in the vicinity" information discussed in the previous paragraph.

I believe the most promising way to do so would be to add bidirectional LSTMs after the first convolutional layer but before the middle (encoded dimension) layer. Or perhaps to use convolutional filters with a larger step on the X (position) axis.

Performance Performance-wise, the current biggest bottleneck of the approach does not come from the autoencoder itself, which is relatively straighforward. The biggest performance cost comes from reading the BED files giving the ChIP-Seq peaks positions and transforming them into tensors, as ascertained by microbenchmarks I performed.

To alleviate this, I split the original BED file (which clocked in at 4 GB) across multiple smaller BED files, adding an index to determine which one contained each candidate CRM. As a result, file access is handled by the operating system and spares us from having to browse the entire file each time we query a CRM.

Indeed, this was necessary because there is no way to directly access a given line in a text file. One must instead read it from the start and count the "\n" characters encountered which means indexing the single large file would not have worked.

3.8. Article

atyPeak is, as of writing, submitted to the *BMC Bioinformatics* journal upon editorial recommendation. It has been previously presented at ISCB 2019 in Basel.

Anomaly detection in genomic catalogues using unsupervised multi-view autoencoders

Quentin Ferré[1,2], Jeanne Chèneby[1], Denis Puthier[1], Cécile Capponi[2]*, Benoît Ballester[1]*

[1]Aix Marseille Univ, INSERM, TAGC, Marseille, France
[2]Aix Marseille Univ, Université de Toulon, CNRS, LIS, Marseille, France

*Corresponding authors : Cecile.Capponi@lis-lab.fr, benoit.ballester@univ-amu.fr

Abstract:

Accurate identification of Transcriptional Regulator binding locations is essential for analysis of genomic regions, including Cis Regulatory Elements. The customary NGS approaches, predominantly ChIP-Seq, can be obscured by data anomalies and biases which are difficult to detect without supervision. Here, we develop a method to leverage the usual combinations between many experimental series to mark such atypical peaks. We use deep learning to perform a lossy compression of the genomic regions' representations with multiview convolutions.

Using artificial data, we show that our method correctly identifies groups of correlating series and evaluates CRE according to group completeness. It is then applied to the ReMap database's large volume of curated ChIP-seq data. We show that peaks lacking known biological correlators are singled out and less confirmed in real data. We propose normalization approaches useful in interpreting black-box models. Our approach can be extended to other similar problems, and can be interpreted to identify correlation groups. It is implemented in an open-source tool called atyPeak.

Introduction

The decreasing cost of gene sequencing and other genomic assays localizing various regions of interest (epigenomic features, TF binding regions) has resulted in a wealth of experimental data from the broader scientific community as well as from large consortia (eg. ENCODE[1]). This data has been collated in warehouses such as the GEO database[2] or

ArrayExpress[3] to facilitate inference and functional annotation of genomic regions. This includes Cis Regulatory Modules (CRMs), which regulate gene expression through binding Transcriptional Regulators (TRs), including Transcription Factors (TFs) binding directly to DNA, and co-factors binding to other TRs forming a regulator complex. Localized clusters of TR bindings form Cis-Regulatory Elements (CREs). In this paper, we focus on improving CREs detection and characterization through better identification of TR binding locations.

While TF binding sites (TFBS) may be predicted based on DNA sequence, statistical precision is low[4] and the use of experimental data, such as ChIP-seq[5] that combines chromatin immunoprecipitation with massively parallel DNA sequencing, is preferred[6]. ChIP-seq can detect both TFs and co-factors binding regions. Each region is associated to a peak in the signal, wider than the region. However, such large scale NGS approaches are known to contain errors and biases resulting in artifactual or anomalous elements. Low complexity regions complicate mapping[7], and ChIP-seq presents several known difficulties[8] including immunoprecipitation quality[9], inadequate controls and other factors complicating peak calling[10]. False positives can be introduced for biological reasons[11,12] and through peak callers (FDR of 1-5 % or more[13]). Besires errors, anomalous peaks can be caused by other biological and technical specificities (eg. different protein fixation kinetics), systematic experimentator biases, mutations creating new TFBS, TRs having rare secondary roles, etc.

Such problems are difficult to correct *a posteriori*. In some cases, human manual curation is possible to label artifacts that can be subsequently used to train supervised machine learning models, some of which also leverage deep learning and combinations between series[14]. But this is seldom available. There is currently no systematic detection method or database of known false-positive regions, besides the ENCODE blacklist[15]. To mitigate this, one can enforce quality criteria at each step, from sequencing (read quality, existence of replicates) to mapping (proportion of mapped reads, low number of regions mapped by unique reads) to peak calling (IDR). The IDR[16] measures consistency between peaks called for two replicas of the same biological condition. However, it is only pairwise, singles out entire series and not individual peaks, and if one of the two replicas considered has poor quality both replicas will get a low score. It also cannot be applied to corroborate data from two different protocols or conditions (eg. different laboratories or TRs).

While quality criteria can be computed for every step of the data processing, which ENCODE does, large amounts of data nevertheless increase the risk of at least one false positive being present and would be detrimental to CRE analysis. Here, we seek to work at a higher scale and use correlations between processed data. As supervised curation is unavailable, the question is whether we could use another property of the data to identify inconsistent elements. Since TRs most often act in combination through complexe formation[6], it follows that biologically significant CREs would likely be clusters of TRs. Similarly, different experimental series should have patterns of corroboration (e.g. where series A says there is a peak, series B says so as well). Here, we define a dataset as one ChIP-seq experimental series for a given TR (in a given cell line, those can be technical replicates or different experiments). For example, RAD21 is significantly associated with CTCF in insulator regions[17]. As such, RAD21 and CTCF form a correlation group and finding CTCF alone would be suspicious. Such combinatorics are indeed considered of major interest to CREs detection[18] and meta-analysis of datasets is emerging[19]. As such, we turn to the more general problem of anomaly detection, meaning to identify elements that do not

conform to the expected normal patterns[20], as a substitute for curation. In this study we focus on detecting anomalous or atypical peaks, where atypical is defined as not respecting the typical TR and/or dataset combinations learned from the data. Removing anomalous peaks will, in turn, fulfill our objective of improving CRE quality. As CREs have high peak density, anomaly detection methods can be used.

We consider the ReMap[21] database, whose initial curation and uniformized reprocessing workflow provide sufficient quantity and quality to use the outlier detection approaches we propose. Given the volume of data and the potential complexity of the combinations, Deep Neural Networks (DNN) models are a natural solution and have been used before on genomic data[22]. They are able to learn complex distributions, not achieved by methods such as PCA, by using multiple layers of increasingly-abstracted representations. Specifically, autoencoders are known to be effective in unsupervised anomaly detection. They also allow us to work at the level of individual loci. Furthemore, this is also a multiview problem[23] as each TR can be thought of as one view composed of several datasets. Thus, our approach integrates correlations between datasets and/or TRs, leveraging another strength of NNs.

To remove atypical peaks, we propose atyPeak, a stacked convolutional autoencoder, and supply processed data files for selected TFs and cell lines from ReMap. Since no gold standard dataset exists to perform cross-validation we demonstrate our approach with artificial data to ensure robustness of our model[24]. Our approach is applicable to any series of intervals, not only ChIP-seq regions. It also offers some interpretability and can be used to extract and interpret the aforementioned correlation groups, as identification of clusters of TRs is of great interest[25].

Results

Representation and processing of Cis Regulatory Modules

To apply our method, each CRM is first converted to a 3D tensor representation of the peaks it contains, where the X,Y,Z axes represent respectively genomic position, datasets of origin, and TR of interest (Figure 1A). We then use a convolutional autoencoder to perform a lossy compression. The representations are viewed by the model through convolutional filters. They focus first on the correlations between datasets and then between TRs, in a stacked multiview approach (Figure 1B). This produces an encoded representation of the CRM, passed to a decoder attempting to reconstruct the original. In the end, each peak is given an anomaly score corresponding to its difference in value in the original and rebuilt representations.

Our approach can be applied to any type of sets-of-intervals data from multiple sources in the same format, not only omics. In subsequent figure legends, “deep dimension” is the number of neurons in each Dense layer, while the “filters number” is the number of kernels in each Convolutional layer. LR stands for learning rate.

Validation using artificially generated data

Without gold-standard data and in the absence of precedent readily available comparable methods, we generate artificial data designed to approximate real CRMs to confirm the model’s ability to correctly discover correlation groups of sources (a peak’s source is its {TR, dataset} pair). Our goal is to simulate biological complexes of TRs, each one being a correlation group. To generate artificial regions, we stack a random number of peaks around a given position. The sources of those peaks belong to one of two (or more) predefined sets of sources representing correlation groups (Suppl Fig 1). The choice of set is made once per CRM. As a result, peaks from each set will significantly correlate only with other members of the same set, forming a correlation group. We also add to the CRM random noise representing atypical peaks not respecting existing correlation groups.

We tested our model’s ability to learn which predefined correlation group each peak belongs to, as opposed to the individual peak itself. Indeed, when peaks from a given correlation group are present, the model rebuilds the entire group in the neighborhood of the peaks and not the individual peaks (Figure 2A, 2 groups). A peak’s value in the rebuilding depends on how many sources from its group are present. The more complete the group is, the higher the value (although fully complete groups are unlikely to occur in the actual data). Peaks added to the rebuilt tensor by the model are called phantoms (Figure 2). Biologically, this means that the model will identify common TRs and/or dataset combinations. Each peak will get a score proportional to the number of correlators present in its vicinity, and the missing correlators will be added as phantoms.

Stability to different group characteristics

Our model is still effective with more complex and realistic artificial data models. For

instance, the number of binding sites across TRs differ but we show that the model is not biased by differences in abundances between correlation groups (Suppl Fig 3A). Furthermore, biological correlation groups are not mutually exclusive (eg. for the TRs A, B and C, there can be both an “ABC” and an “AB” group) and the model is also capable of learning such groups (Suppl Fig 4B and 11) although this is not always reliable and comes with caveats and precautions described in Methods. Generally, phantom peaks from overlapping groups will be generally present but less pronounced than they should be.

It is possible to extract the correlation groups learned by the model to mine for biologically relevant combinations of sources. It can be done by interpreting the encoded dimension (Suppl Fig 6), or instead by identifying the correlators of a given source by looking at which phantoms peaks are added when it is present (Figure 2, and Suppl Fig 11 in real data).

Scalability and the information budget

Any compression is characterized by its aggressiveness: one that is not aggressive enough can afford to learn details (for us, the noise we want to remove) but a too aggressive compression might lose too much information. In our method, this depends on the relative information budget, which is the ratio between the data’s information and the model’s entropic capacity. Biologically, this means the budget to be used depends on the size of the database: the number of sources considered and the number of relevant correlation groups in the data. For example, in Figure 2B the model with a too large budget learns too precise, smaller, non-significant correlation groups but the phantom peaks added are still not from sources outside the correlation group.

To rigorously choose the information budget, we propose to verify that the model correctly learns pairwise correlations. We design and propose a quality score (Q-score) which is also used on real data. In Figure 3, we demonstrate scaling budget upwards to accommodate richer data with 8 correlation groups. The model tends to focus on the most frequent sources when learning the groups, which can often result in grouping the least sources together in background groups and ignoring the very rarest ones. These differences in focus can be alleviated (Suppl Fig 4A), but mean that a correlation group does not necessarily represent a single or complete biological complex, which should be remembered when interpreting them.

Systematization on many observations

In Supplementary Figure 1, we show that the observations made above hold true when considering a larger number of artificial CRMs. The presence of a peak results in phantoms peaks from the same correlation group of sources, but not from the other groups. Noise peaks, lacking their usual correlators, get rebuilt with a lower value. Models with a too large budget will learn smaller non-significant groups.

The learned group themselves can be subject to certain biases, such as giving higher values to comparatively more abundant elements within a group, the fact that all correlation groups do not have the same average completeness, and contributions to the rebuilt values

from several overlapping groups. We performed normalization to counter such biases, by evaluating the score given by the model in controlled artificial CRMs when only peaks from the considered source are present, and normalizing by TR in the end. Non-normalized results are also available, but the score must then be interpreted relative to the average score for its source. More details about the experiments conducted on artificial data and the conclusions drawn are available in Methods.

Application to real biological data

Having evaluated our model on artificial data, we processed biological data from the ReMap project. We used selected CRMs for the Jurkat, Hela, K562, MCF7, CD34 and ESC cell lines. This data was sparser than the artificial datasets, as not every dataset contained every TR. This necessitated “crumbing” (Suppl. Fig. 9) and adapting the information budget. Parameters are detailed in Supplementary Table 2 and were chosen thanks to the Q-score. We covered a variety of cell line profiles, ranging from Jurkat’s sparse genomic binding data with many datasets concerning only one TR, to high-dimensionality examples such as K562 proving our model’s scalability potential.

Figure 4 shows two representative examples of CRMs in HeLa along with their rebuilding by the model. The difference in rebuilding shows that the model does not always rebuild the average CRM and has learned different correlation groups. In Figure 4A, BRD4 has a low score as the model was expecting more correlators (cf. the estimated correlation groups presented in Suppl Fig 11), unlike AFF4 and ELL2. Conversely, in Figure 4B the BRD4 peak was expected and is added as a phantom.

In HeLa, we observed 3-4 different correlation groups learned by the model, in line with what is likely biologically significant. In practice more groups are learned as overlaps of groups: this is visible in HeLa with AFF4 and ELL2 still benefiting from the presence of other sources (Suppl Fig 8), and BRD4 or SFMBT1 being part of several correlation groups (Suppl Fig 11). Across most cell lines, the model usually needs to learn over only a few thousands of CRMs until the loss begins to stabilize. This suggests that most CRMs, among the selected ones, have similar configurations, which is confirmed by the fact that averages calculated over different sets of 10K random CRMs will be very similar. Early stopping was often needed to prevent overfitting in cell lines with less sources such as CD34, and HeLa to a lesser extent.

The final score is normalized so that a peak in the average configuration for its source in terms of presence of known correlators will get a score of 750/1000. We provide both normalized and raw scores, at the user’s convenience.

To confirm the predictions of our model, we use the ReMap 2020 update[44] (Suppl. Fig. 14). Presumably, atypical TR fixations as identified by the approach would be less emphasized considering the larger amounts of data processed in this update. We find that the CRMs with the highest update ratio (number of peaks in 2020 divided by 2018) were the ones with lower atypeak scores, suggesting that they were incompletely characterized in 2018 and needed more data. They could also be high-noise regions, although this is unlikely as they were among the richest CRMs in 2018. Supervised data would be required to

determine the correct hypothesis.

At the individual peak level however, we consider the update ratio for peaks of the same TR within the same CRM, which would confirm the binding of this TR here by drawing from other data. While peaks at any score can have any update ratio as ReMap2020 does not simply replicate all previous experiments a given constant number of times, we find that peaks with a higher update ratio and thus more robust confirmations seldom had a score under 250 in atyPeak.

Confirmation of the biological meaningfulness of identified correlations

We also use pairs of Transcriptional Regulators for a second confirmation. These pairs are known to co-occur on the genome, either via their high Jaccard indexes (example in Suppl. Fig. 12) or from the literature such as GABPα with ERG in Jurkat[26], ELL2 with AFF4 in HeLa[27], or SFMBT with RCOR1 also in HeLa[28].

For each interesting pair A,B of TRs, we consider the distribution of the scores given by our model for A when B is also present in the same CRM, and when it is not. We provide some examples in Figure 6 with high and poor correlations. We observe that, as we demonstrated earlier, the score given to a peak of a given TR is increased when another TR that correlates with this one is present, and vice-versa for the non-correlating ones. This means that, when properly calibrated, atyPeak learned on its own significant correlations.

Singleton peaks tend to have a lower score compared to peaks found in richer CRMs. This is expected, since CRMs are regions with multiple cooperating TFs, singleton peaks are generally suspicious. This is not simply linear (Figure 6B), which further illustrates that the model learns biologically meaningful correlation groups and not simply that richer CRMs are better.

Some interpretation of the model is possible on real data as well (Suppl Fig 10). When performing combination mining (Suppl Fig 11) the learned groups match the expectations about correlating TRs discussed for Figure 6, such as AFF2 and ELL4 being present in the same group. BRD4 and others are learned together as a more “background” group. SFMBT1 and RCOR1 were learned alone with low phantoms from other sources, although it is justified by their relatively low Jaccard index with the other TRs (Suppl Fig 8), since as we established overlapping groups are hard to learn. In general, careful interpretation of the learned groups is necessary for cell lines having high frequency imbalances such as Jurkat, or high dimensions such as K562 or MCF7.

Availability of data and code

The source code and data are available at <https://github.com/qferre/atypeak>. Treated data files with scores for the considered ReMap peaks are available as a UCSC session at <http://genome-euro.ucsc.edu/s/qferre/atyPeak_hg38> (Figure 5), or as BED files with diagnostic data at <https://github.com/qferre/atypeak-files> and on the ReMap website at <http://remap.univ-amu.fr/> [Tab not currently available, under construction].

Discussion

We designed an anomaly detection method to identify regulatory peaks that are not part of a cluster of regulatory elements. Our method finds outliers which do not respect the usual sources (TRs and/or experimental series) combinations. Peaks get a higher score when more of their correlators are present, forming a richer cluster. This allows for CRE detection taking TR composition into account. Crucially, our unsupervised approach does not require an *a priori* set of known anomalous experimental peaks, which is seldom available and could bias a model towards the particular kind of anomalies it represents.

atyPeak learns usual source combinations patterns, while the noise (anomalous peaks as defined in Introduction) is discarded. By focusing on combinations instead of a particular type of anomaly, we *de facto* indiscriminately correct most of the errors discussed previously. The combinations learned by the model will be based on what is typical in the regions provided in training (for example, our selection contains many gene promoters). We do not fixate on a single type of error, nor do we emit a definitive judgement on peak quality, as it is impossible without supervision. This is made possible by using high-quality ReMap data; indeed, unsupervised anomaly detection presupposes a low proportion of anomalies.

We have validated our approach using artificial data designed to model correlating elements and a noise of atypical peaks. The model autonomously learns multiple *n*-wise correlation groups of sources in both artificial in real biological data. As the underlying task is compression we use comparatively small, simple networks which nevertheless perform well. Hence, our method can be readily used on a laptop from training to application.

Usage

Our approach estimates how “typical”, with respect to source combinations, each peak is when compared to all the CRMs in a single given cell line, since we currently work within one cell line at a time. As the model is unsupervised, anomaly score thresholds are at the user’s discretion depending on their needs. For example, a large scale analysis might exclude lowest scoring peaks (ie. assumed False Discovery Rate). However, a focused study of a single or selected experimental series may rely on low-scoring peaks as they might be caused by certain events of interest (mutations, etc.). It is also possible to use high-scoring atyPeak peaks to detect candidate regulatory regions of interest and use that selection as a filter when looking at other genomic data. A low average score for a given CRM also suggests it might be incompletely characterized and missing information about other peaks, instead of noise.

Scaling the information budget is crucial to learn the appropriate correlation groups, and databases with more experiments will require larger models. To do so, we propose a Q-score based on whether known correlations influence the rebuilding. We introduce a group-based normalization to correct rebuilding biases and introduce interpretability. We believe these contributions could be applied to other latent variable models, and more generally to any black-box model with arbitrary complex correlations. Both are first steps and warrant further research.

While we provide scores for selected ReMap data, our model can be reused to denoise any kind of region database with multi-view sources such as in the ReMap data format (where peaks are independant between cell lines and each have a TR and a source dataset, cf. documentation).

Our hierarchical multi-view approach is a type of intermediate fusion, where a first latent space is learned based on one type of combination, followed by learning combinations of it across another dimension. The full tensor represents the CRM; the latent learning by the convolutional kernels is focused on local combinations analogous to CREs (local clusters). By looking at the added phantoms, it is also possible to interpret the learned correlation groups to find combinations of corroborating experimental series, and regulatory clusters of collaborating TRs.

Generalization

ChIP-seq protocols, and subsequent quality, can vary wildly. Since our approach learns how experiments are corroborated by others, such differences in quality are self-correcting. Hence, we do not require unified protocols like large consortia (ENCODE) would, and can work with heterogeneous data from multiple provenances. That being said, having more and larger genomic datasets for each Transcriptional Regulator will help.

While our study focuses on ChIP-seq data, our approach can be generalized to any type of data consisting of a series of peaks, or more generally corroborating time-series intervals from multiple datasets. In genomics, this includes ChIP-exo, ATAC-seq, as planned for future ReMap releases, or even otherwise determined regions like promoters of overexpressed genes in a certain condition. But it could also be used, for example, to compare weather forecasting models.

The atyPeak approach could also be applied to many multi-omics problems by changing the meaning of the dimensions, ie. integrating different assays for different cell lines instead of different datasets for different TRs. More generally, we propose to leverage typical combinations between sources to perform anomaly detection by representing multi-view data as K-dimensional tensors (for K views) and using structures designed to consider those combinations.

To our knowledge, our approach shows the first use of a large-scale meta-analysis of ChIP-seq datasets to corroborate them with each other, using deep learning methods to integrate them in complex combinations. This allowed us to identify and eliminate atypical peaks that do not respect such combinations, resulting in higher-quality data available for genomic analysis.

Methods

Materials

Data sources

In this study, data is provided by ReMap 2018[21]. ReMap endeavors to identify and characterize regulatory regions from a large-scale integrative analysis of DNA-binding protein experiments. The 2018 human update uniformly annotated and processed 3,180 ChIP-seq experiments, including some biological replicas, creating a catalogue from the analysis of 35.5 million peaks (after merging) for 485 TRs in a variety of cell types and tissues. The regions of interest or Cis-Regulatory Modules (CRM) selected for this study are defined as a region binding at least two different regulatory proteins in all the cell lines and tissues of ReMap, in order to mitigate variation coming from non-standardized sources. A CRM can contains from two to a few thousand peaks, in one (or more) Cis-Regulatory Elements(s). In ReMap, this adds up to 1.6 million CRM; however current estimates point to in the order of magnitude of a few hundred thousands biologically significant ones only.

Data selection

We used as a query a subset of the aforementioned CRMs, keeping those with at least 100 peaks across all cell lines for 65,535 CRMs in total, to focus on the densest genomic regions. We processed only a subset of representative cell lines and selected only certain relevant TRs, to reduce the sparsity of the resulting tensor representations. The list of selected sources is present in Supplementary Table 1. Our goal was to consider TRs with high biological significance, comparable abundances, and interesting combinations. In practice, for each cell line, we get the TRs with the most experiments, but if a selected TR has a known collaborator further down the line, said collaborator may take the place of a previously selected but isolated TR (eg. MYC/MAX).

Autoencoder model

Artificial neural networks (ANNs) are assemblies of neurons, which are logistic units outputting a result dependent on a linear combination of its inputs. In a network, the output of one layer of neurons is fed to the next layer. The weights of each neuron are learned by backpropagation. More specifically, an autoencoder is an ANN whose goal is to learn for each provided example a compressed representation sufficient to rebuild it in the most efficient manner, which entails discarding signal noise[29]. Applications of autoencoders include dimensionality reduction, anomaly detection, information retrieval and image processing.

Here we performed a lossy compression of our CRM representations, ie. transforming them into shorter vectors capable of returning similar information. When performing a lossy compression, noise and other non-information are the first elements to be lost, but so are fine-grained details. More interestingly, anomalies (in our case atypical peaks) are lost because no regularity involving them is found. Compressing also introduces

artifacts, which for us are phantom peaks it was expecting to see.

As any compression algorithm implicitly maps the compressed vector into a feature space, and learning such mappings based on certain criteria (ie. minimized loss) is a quintessential machine learning task, there is a close connection between machine learning and compression. Deep (convolutional) autoencoders are particularly suited to it[30] and can be tailored efficiently to variations of the problem such as group anomaly detection[31]. The rebuilt image is not a cleaned image, but a compressed one, unlike in denoising autoencoders[32], but those cannot be used here since we have no ground truth, ie. no *a priori* information on which peak is good or not.

Existing anomaly detection approaches solve slightly different problems and ours (anomaly detection based on sources/dimension combinations) is not directly comparable. The closest existing parent is the detection of anomalous vertices in a dynamic graph[33] giving precedent to our approach of giving a score to each vertex at each time step depending on their behavior. Here we use an autoencoder to learn such a score, for which the use of graph convolution is precedented[34] although instead of edges we have multi-view bags of items. Another related approach is multivariate time series anomaly detection, but here we seek to label anomalous features, not anomalous points.

Data representation

Each putative CRM is represented as a 3D tensor $T \in \mathbb{R}^3$. This tensor of peak presence contains a representation of ChIP-seq peaks falling into this region: the x, y, z dimensions are respectively the nucleotide position, the experiment/series ID, and the Transcription Factor involved in the ChIP-seq. Each cell line is analysed separately. The value at each position of the tensor is 1 if there is a peak, 0 otherwise. CRM longer than 3200 bp are truncated, and those shorter are padded with zeros (3.2kbp was the 9th decile of length).

We then downscale the tensor by a factor 10 along the X axis ("squishing") since the data has low granularity along that axis to allow the use of smaller, easier to train convolutional kernels. Also, to counteract lower rebuilt values at the margins of the tensor in CNNs, we add a padding of meaningless zeroes at the beginning and end of the X axis instead. Its length is twice the convolutional kernels' length on each side. In real biological data only, to help the model learn in spite of the sparsity, we also add crumbing (Suppl. Fig. 9) where for each tensor element where there is a nonzero value of v, we add $0.1 * v$ in each position in the same Y or Z axis as a hint. Crumbing is cumulative.

Model architecture

The structure of the model is detailed in Figure 1B. Our model has two parts: an encoder creating a latent representation and a decoder retrieving the original tensor. As with all autoencoders, the model is trained to try to rebuild the original CRM representation as its output. The full model parameters are available on GitHub. Our model was implemented using Keras 2.3[35], with Tensorflow 1.15 and NumPy 1.18.1[36].

Convolutional encoding

The CRM representations are viewed through sliding convolutional filters, to focus on

correlations between the TRs and datasets. A convolutional filter gets as input a slice of a matrix and outputs a weighted sum of its elements, with the weights forming the filter proper. The first layers are two successive convolutions with two different types of kernels (combinations of datasets, then combinations of TRs).

Let n be the number of TRs in the cell line, m the number of datasets and k the size of the convolutional kernels. As a result, the kernel shapes are $(k, m, 1, channel = 1)$ for the first, and $(k, 1, n, F)$ in the second where F is the number of filters in the previous layer. As there is no ordering to the datasets or TR, we perform a depthwise convolution and read the entire dimension at once. Default k is 20. We use only one layer per dimension. We use few kernels, lower than the later Dense layer size, creating a bottleneck[37], but larger Dense layers still improves rebuilding (Figure 3).

The combinations are learned over a short window across the region given by the variety and stride of the convolutional kernels. Convolution filters have a kernel regularisation of 2.5E-3 by default and Dense layers (see below) have low Dropout regularisation (10%), except for the encoded layer which has none so it can specialize. We observed that a stacked approach of one dimension at a time can lessen training problems associated with large number of dimensions.

Convolutional filters are known to be useful in finding combinations of elements across dimensions, including in biological sequences[38]. Multi-view integration is traditionally done by using different strides for the filters, or by processing each view followed by a feature fusion[23]. In contrast, as our dimensions are incomparable, we express a hierarchy between our two dimensions by integrating datasets combinations first, learning a first latent space which is then passed to another convolutional layer, which learned its own latent space based on TR combinations (across another dimension) of the values of the first latent space.

Integrative layers and decoding

The convolutional layers are followed by 4 regular (Dense) integrative layers, to learn complex combinations. On each layer, only the last dimension (filters) provides weights, resulting in Time-Distributed layers with no communication along the X dimension. The ReLu activation function is used.

We obtain an encoded dimension in the fourth Dense layer. For the decoding, we consider each element of the non-human-readable encoded dimension as a latent variable. A first layer of convolutional filters reads the entire dimension once to produce a first decoding, and a second and final layer has one filter per source (TR-dataset pair). This is done because unlike classical images, there is no order to the features. The final layers perform a reshaping of the result back to the original tensor shape.

There is no communication along the X axis, unlike NLP models such as Transformer or LSTMs, as we focus on local combinations. However, along the other axis of the encoded dimension, the encoding layer has access to the state of all other learned neurons making this partly reminiscent of a transposed attention mechanism[39]. Custom time-based layers and constraints could be added here. This is not necessary to work with large, overlapping ChIP-seq peaks, but might be needed to integrate the ReMap 2020 new ChIP-exo and DAB-

seq data.

As such, even though the full tensor represents the CRM; the latent learning is focused on local combinations analogous to CREs (local clusters) in a window the size of the convolutional kernels.

Loss used

The loss used when training the model is the Mean Squared Error, or L2 loss. Hence, when the model adds phantoms from the same correlation group, it must lower the value of the original peaks. This forces compromise compared to a L1 loss, since $\| \begin{pmatrix} 0 \\ 2 \end{pmatrix} - \begin{pmatrix} 1 \\ 1 \end{pmatrix} \|_2^2 < \| \begin{pmatrix} 0 \\ 2 \end{pmatrix} - \begin{pmatrix} 2 \\ 0 \end{pmatrix} \|_2^2$. Using L2 loss on an autoencoder has been previously shown to be effective to perform denoising even when no cleaned images, only corrupted/noisy versions, are available[40].

The best results were obtained with the Adam ("Adaptive learning rate for sparse data") optimizer[41], which is effective on problems featuring large data and very noisy and/or sparse gradients. It adapts the learning rate based on gradient moments. Our base Learning Rate for the Adam optimizer is 1E-4, which is 0.1x its default. We keep the model with the lowest loss of all the epochs during the training process. Training is stopped when losses stops diminishing with a patience of 5 and a minimum delta of 2.5E-4 (chosen empirically) to consider it to have diminished significantly. Batch size is 48 CRMs with 10 batches per epoch in artificial, 48 in real data. We can also use a weighted loss by apply weighting to the loss for each dataset or TR separately.

Anomaly score

Finally, each peak gets an anomaly score based on the autoencoder reconstruction error. The better the peak according to the model, the higher the score. Such an approach has precedent in signal processing[42].

We define an anomaly score to compare a tensor X to its rebuilding by a model noted $h(X)$. We have A the anomaly score tensor defined as :

$$A[x,y,z] = \begin{cases} 0 & \text{if } X[x,y,z] = 0 \\ 1 - \frac{X[x,y,z] - h(X)[x,y,z]}{X[x,y,z]} & \text{else.} \end{cases}$$

Dividing by the original score accounts for the potential crumbing. The score of each peak is the maximum value in A across all nucleotides of the peak. This is necessary to correct cases where near vicinity peaks will get high score only on the parts where they overlap and because peaks smaller than the convolutional filter's length will get a lower rebuilt score as a result. Peaks can sometimes be divided between two (or more) of our 3200-bp windows, getting one score for each rebuilt matrix: we merge them by giving them a score that is the mean of each.

Normalization of correlation group biases

This normalization aims to correct bias in rebuilding based on the learned correlation groups. We calculate a weight for each source (meaning each {dataset, TR} pair), based on the following steps, to be applied to all anomaly scores computed for this source.

$\mu_X(T)$ designates the 2D matrix of the mean across the X axis (region size) of the 3D tensor T, and $max_X(T)$ its maximum. $M[s]$ designates the value of the matrix M for the current source. $h(T)$ is the tensor obtained as output when passing as input the T tensor to a trained atyPeak model. We note F_t a "full CRM" which is a 3D tensor representation where all possible sources with abundance higher than zero are present along its length with a value of 1. Let $F = \mu_X(F_t)$. The correlation group of a source can be estimated (see Interpretability) by preparing a CRM containing only a peak for the given source along its entire length denoted U_s. We get such a request mask as $R = \mu_X(h(U_s))$.

The first step corrects for intra-group biais in rebuilding, due to learning bias (usually too high learning rate) or abundance differences within the group. There, the sum of the rebuilt CRM will be biased too. We get the first weight $k_1 = \sum h(T) / \sum T$.

For inter-group bias, recall that the rebuilt value of a peak (value in $h(T)$) is proportional to how complete its correlation group is. But group have different sizes, cf background group and different groups. Our goal is that peaks gets the same score where their group's current completeness relative to its average completeness is the same. We define occupancy for a CRM for the current source as $M * max_X(U_s) * IW$ where $M = R/max(R)$ and IW is the matrix of intra group weights for all sources. These are pointwise multiplication, not matrix products. We use a Monte Carlo approach by iterating over a portion of all CRMs and calculating the mean of all their occupancies $\mu(\theta)$, excluding zeroes to get only the correlators when the source is actually present, and self-correct for relative loneliness. The final second weight $k_2 = \theta_F / \mu(\theta)$ where θ_F is the occupancy calculated of F_t.

Thirdly, if a source is in several groups, phantoms from several groups can accumulate and will not be seen at step 2. We evaluate how much the sources that are not in the request will contribute. We calculate the mean and max negative occupancies (η) exactly as above, except we use a negative mask M_n instead of request R, where $M_n = (F - R/R[s]) * F[s]$. We ponder by the average presence of these other peaks to get $k_3 = 1 - ((h(F)[s]/F[s]) * (\mu(\eta)/\eta_F)$

The final weight is $k = k_1 * k_2 * k_3$. To prevent overcorrection of sources that were not learned by the model, all k are capped at 10. For now, having a CRE with more peaks than average results in higher rebuilt values, as we consider that for CREs in particular more TRs mark denser/better CREs. This assumption could be changed here by penalizing values above the corrected average quality.

The final step consists of centering and reducing/normalizing the scores by TR, under the assumption that no TR is inherently of a better quality than the others. Having

more correlators (ie. data less sparse for the same dimension, more datasets per TRs) is a benefit. For each peak, if their score at this step is s their final score is $s_f = 750 * (1 + \frac{s - \mu_{TR}}{2\sigma_{TR}})$, where μ_{TR} and σ_{TR} are respectively the mean and standard deviation of scores observed for this source's TR at the previous step. Note that scores are usually not normally distributed.

We center around 750 to use a larger part of the score scale for cases where the local cluster (CRE) is less complete than average, which are the cases we want to mark. If you choose to use a non-normalized score, compare each score to the median score for its source. This normalization is a step in the right direction that independently moves score averages for different TRs closer (Suppl Fig 13) but warrants further research.

Training and interpreting the model

We provide scripts to directly process a BED file in ReMap format with diagnostic figures and usage instructions in the Readme.

Impact of data characteristics/scaling on required information budget

As we discussed in Results, the information budget determines the aggressivity of the compression. It depends on the relative information budget. It is the ratio between the quantity of information to be learned in the data (itself a function of the number of TR and/or dataset combinations) and the model's entropic capacity (how much information can be stored in the compressed representation). Adequate hyperparameter tuning is a widespread problem in deep learning as a higher information budget will predictably increase the model's Vapnik–Chervonenkis dimension and make it more prone to overfitting.

In our case, the entropic capacity is mostly increased by increasing the dimension of all Dense layers and the number of convolutional filters on one side (more is higher). But also by diminishing the learning rate (LR) on the other which was often necessary to reach lower losses, even with all other parameters constant. We saw in Figure 3 that to achieve the same aggressivity, the required entropic capacity scales up with the quantity of information in the data. Figure 2B, on the contrary, is an example of overprecision. Larger dimensions (more datasets and/or TRs in the database) require a higher information budget, even with no additional information (Suppl Fig 3B). However, Lower Learning Rates are more of a necessary condition than larger models to reach higher precisions with higher dimensions (Suppl Fig 3C). Learning larger correlation groups (composed of more sources) is also harder.

The most frequent sources are learned in more precise groups, while the rarer ones appear often grouped together in more "background" groups. All groups, but especially the latter, are not expected to be fully complete (meaning all the sources are present) in the real data. More generally, sources that are comparatively too rate (empirically ⅕ difference) may be completely disregarded by the model as they are seen as systematic noise. All those tendencies are more visible in high-dimensionality examples or those with higher imbalances, and can be alleviated by using a weighted loss: Suppl Fig 4A shows that dimensions with a higher weight will be focused on and get more precise groups.

We also show in Suppl Fig 4B that the model is capable of learning overlapping groups (where the groups are “G1” and “G1+G2” instead of “G1” and “G2” like in Suppl Fig 1) However, it required learning adjustments with higher weighting on the rarest dimensions to direct the learning, and more importantly early stopping. With a variety of other parameters, peaks in G2 produce only marginal phantoms for G1, or we get too precise or non-homogeneous groups (Suppl Fig 4C). Note that G1 will often not produce phantoms of G2 (although it should and does sometimes happen, like in HeLa) so be careful to look at the estimated groups for all sources when interpreting the model. Relatedly, even in non-overlapping groups the watermark (ie. the lonely control peak we added at the same position to most of the CRM that does not particularly correlate with other groups, see Artificial data) does not create phantoms anywhere else. However, watermark phantoms are produced by peaks from (certain sources in) the G1 and G2 groups. The rarer of those two groups often erroneously produces stronger phantoms, a tendency reduced when this rarer group is weighted more.

Loss and training

Due to the high dimensionality and sparsity of our data, we used lower Learning Rates (Suppl Table 2) and large batches to counter overfitting and batch effects. We also used early stopping in most cases, in most cases stopping even before a loss low plateau is reached to prevent the model from adding bias in a futile attempt to improve.

With different random seeds, we observed over several runs small but real deviations in scores and estimated correlation groups. As with most machine learning approaches, we recommend averaging over several runs (2-3) for both these applications.

Training the model takes around 10-30 minutes per cell line for smaller models (HeLa) and 1-5 hours for larger ones (K562 and MCF7). However, reading and processing the source BED files is a large part of this time and the approach is not CPU bound. Times given on an i7-7820HQ and on an SSD drive. GPU use did not significantly improve running times. Production of the resulting BED file after training is also time consuming (around 12 hours for K562 but 40 minutes for HeLa), so it is advised to check some rebuilt matrices before proceeding.

Interpretability

To interpret the latent variables in the encoded dimension (Figure 1, Suppl. Figs. 6 and 10), we use a gradient ascent method to build an hypothetical CRM tensor that would maximally activate each individual row in the encoded dimension layer[43].

We seek $TM = \{\forall i \in [1; \#(E)], \underset{T}{\operatorname{argmax}}\, a_{E,i}(T)\}$. We add some blur at regular intervals on the Y and Z axis during gradient ascent for more natural looking results. By default we use a learning rate of 1, 50 steps in gradient ascent, and the blur standard deviation is $(\sigma_x, \sigma_y, \sigma_z) = (0.2, 1E-2, 1E-2)$ applied every 5 steps. For each latent variable of the encoded dimension the gradient is calculated across the entire length. As the Dense layers are not connected across the X axis, we are considering local combinations. Since this is not the next-to-last layer, the final result will be a complex non-linear

combination of those variables. This should instead be seen as a highlight of the model's focus during learning.

Another type of interpretability is based on the same procedure used in the normalization (Suppl Fig 11). We create a CRM representation U_s that is empty except for one peak for a given source along all its length. By looking at $R = h(U_s)$, we see what phantoms are added by the model, and deduce these are part of the same correlation group as the source we are considering. Due to the peculiarities mentioned above when learning overlapping groups, look at all the sources' estimated groups, as a source A may impact the score of B without B appearing in A's estimated group. Passing R does not always result in values of 1 due to complex nonlinearity, but it is a good approximation.

Note that a learned correlation group of "ABCDE" does not necessarily mean ABCDE are always found together, as seen in artificial data where the model learned the entire G1 and G2 group, which almost never found complete in the artificial CRMs. As such, rarer sources can be grouped in more background groups without necessarily being a complex. For both interpretabilities, negative weights are likely due to sum averaging and should not be focused on. Indeed, the rebuilt tensor is not simply the sum of the estimated correlation groups for the sources present.

Q-score quantifies the quality of the reconstruction

To rigorously choose the information budget, we propose to verify that the model correctly learns generated pairwise correlations. On one hand, if two dimensions (datasets or TF) correlate, finding them both together in the region of interest should result in a higher score for them than when they are found alone; on the other hand if they do not correlate, this should have no impact. To estimate this we design a Q-score, which is lower in better models.

$$\forall(i,j) \in \Lambda, Q = \sum_{i,j} \sqrt{A_i * A_j} * (P + R)$$

The Q-score is defined as where,

$C = [R(i,j) > \hat{\mu}(R)]$
ϵ=$0.05^{*}\#(\Lambda)$
$P = (C - [P(\mu(alone) = \mu(both)) < \epsilon])^2$
$R = (C - [P(\mu(phantom) = \mu(none)) < \epsilon])^2$

Here Λ is a set of all TRs and all datasets (so all possible Y and Z dimensions, excluding the X dimensions of peak position) and the brackets are Iverson brackets denoting indicator variables. Note that we only compare TRs with other TRs and datasets with other datasets, because a dataset and a TR are not mutually exclusive and issues can arise when considering a dimension that is only present as noise when another is present. For the same reason, we consider only positive correlation coefficients later.

C asserts whether the Pearson correlation coefficient between the two considered dimensions is higher than the mean correlation coefficient. It is calculated on the tensor

representations of the CRMs at the nucleotide level.

For P and R, we take 10K CRM tensor representations T and their rebuilding $h(T)$. For each of them, we record the values for the (i, j) dimensions of interest (averaged across X axis). We compare the average rebuilt value of A in different scenarios: For P, when a peak of i was present in T, does presence of j in the same CRM result in a higher rebuilt value for i ? And for R, when i was absent, does the presence of j result in higher phantom values than when j is absent ? To perform these comparisons, we use a Welch test to determine whether the means are different. We use a Bonferroni correction by using a p-value of $0.05/\#(\Lambda)$. We then weight the result by the relative abundance of the dimensions A_i and A_j. We do not normalize the scores with the procedure discussed before because we compare a source with its own values.

Artificial data

We use artificial regions to confirm the model can discover correlation groups. They are meant to approximate real genomic CRMs, hence the generation process and parameters are based on true data.

We define a probabilistic model to generate the artificial data. The output of this model is an ensemble P of peaks, whose characteristics are: their start and end, the IDs of the TF they represent and the experiment they belong to. Hence we have $P = \{(s_i, e_i), act_i \in \{0, 1\}, TF \in N, series \in N\}$ which is then converted into a 3D tensor representation, as explained in Data representation. The generation itself consists of three steps detailed below. Each step is run once per generated artificial CRM. Unless specified otherwise, all random variables used are Poisson R.V. of $\lambda = 1$. See Suppl Fig 1 for an illustration of the dimensions.

First, we place a control peak called a watermark along all the length of the CRM for the 1st TF in the 1st dataset, representing ubiquitous TRs. It will be very frequent but not particularly correlated with other sources and so form its own correlation group. It has a customizable probability (default 75%) of appearing, to prevent the model from learning it and only it when it is too frequent.

Second, we want to place a stack of correlating peaks from different TRs and datasets, at roughly the same positions. The stack will belong to one of two or more TR "correlation groups". The groups are made by splitting the set of all in TRs in two, or more, or by making groups of 4. Group choosing probabilities are equal by default but can be weighted.

Only one such group is picked per generated artificial CRM. We then pick a common center for the peaks, uniformly randomly across the region. Now, we pick $K + 1$ datasets without replacement among all predetermined reliable datasets (by default, the last half of them). In these datasets we will place $N + 1$ peaks. For each peak to be added, we randomly select $P + 1$ TRs from the current correlation group with replacement. N, K and P are random variables. For each TR selected, separately move the center by a distance j_d (uniform R.V. between -200 and +200), take a peak length randomly of L (L is a log-normal

RV of $\mu = 250$, $\sigma = 0.25$) and finally, write the exact same peak among all the datasets selected previously. Note that since artificial data draws peaks at random, there is a larger number of possible combinations than there is usually in real data of the same dimensionality.

Third, noise peaks are placed uniformly randomly from all datasets and TRs to represent false positives and atypical peaks, which by nature do not respect existing correlation groups. To represent false negatives, each peak has a probability $t = 1/4$ of being removed at this tep. Then, we randomly position $F + 1$ peaks (F is a R.V.) by drawing randomly their characteristics like previously. Noise cannot be placed in the watermark.

References

1. ENCODE Project Consortium. An integrated encyclopedia of DNA elements in the human genome. *Nature* **489**, 57–74 (2012).
2. Barrett, T. *et al.* NCBI GEO: archive for functional genomics data sets—update. *Nucleic Acids Res.* **41**, D991–D995 (2013).
3. Parkinson, H. *et al.* ArrayExpress update—an archive of microarray and high-throughput sequencing-based functional genomics experiments. *Nucleic Acids Res.* **39**, D1002–D1004 (2011).
4. Bulyk, M. L. Computational prediction of transcription-factor binding site locations. *Genome Biol.* **5**, 201 (2004).
5. Johnson, D. S., Mortazavi, A., Myers, R. M. & Wold, B. Genome-Wide Mapping of in Vivo Protein-DNA Interactions. *Science* **316**, 1497–1502 (2007).
6. Lambert, S. A. *et al.* The Human Transcription Factors. *Cell* **172**, 650–665 (2018).
7. Li, H. Toward better understanding of artifacts in variant calling from high-coverage samples. *Bioinforma. Oxf. Engl.* **30**, 2843–2851 (2014).
8. Landt, S. G. *et al.* ChIP-seq guidelines and practices of the ENCODE and modENCODE consortia. *Genome Res.* **22**, 1813–1831 (2012).
9. Kidder, B. L., Hu, G. & Zhao, K. ChIP-Seq: technical considerations for obtaining high-quality data. *Nat. Immunol.* **12**, 918–922 (2011).
10. Wilbanks, E. G. & Facciotti, M. T. Evaluation of Algorithm Performance in ChIP-Seq Peak Detection. *PLOS ONE* **5**, e11471 (2010).

11. Jain, D., Baldi, S., Zabel, A., Straub, T. & Becker, P. B. Active promoters give rise to false positive 'Phantom Peaks' in ChIP-seq experiments. *Nucleic Acids Res.* **43**, 6959–6968 (2015).

12. Teytelman, L., Thurtle, D. M., Rine, J. & van Oudenaarden, A. Highly expressed loci are vulnerable to misleading ChIP localization of multiple unrelated proteins. *Proc. Natl. Acad. Sci. U. S. A.* **110**, 18602–18607 (2013).

13. Chitpin, J. G., Awdeh, A. & Perkins, T. J. RECAP reveals the true statistical significance of ChIP-seq peak calls. *bioRxiv* (2018) doi:10.1101/260687.

14. Koh, P. W., Pierson, E. & Kundaje, A. Denoising genome-wide histone ChIP-seq with convolutional neural networks. *Bioinformatics* **33**, i225 (2017).

15. Amemiya, H. M., Kundaje, A. & Boyle, A. P. The ENCODE Blacklist: Identification of Problematic Regions of the Genome. *Sci. Rep.* **9**, 1–5 (2019).

16. Li, Q., Brown, J. B., Huang, H. & Bickel, P. J. Measuring reproducibility of high-throughput experiments. *Ann. Appl. Stat.* **5**, 1752–1779 (2011).

17. Hanssen, L. L. P. *et al.* Tissue-specific CTCF-cohesin-mediated chromatin architecture delimits enhancer interactions and function in vivo. *Nat. Cell Biol.* **19**, 952–961 (2017).

18. Kleftogiannis, D., Kalnis, P. & Bajic, V. B. Progress and challenges in bioinformatics approaches for enhancer identification. *Brief. Bioinform.* **17**, 967–979 (2016).

19. D. Chikina, M. & G. Troyanskaya, O. An effective statistical evaluation of ChIPseq dataset similarity. *Bioinformatics* **28**, 607–613 (2012).

20. Chandola, V., Banerjee, A. & Kumar, V. Anomaly Detection: A Survey. *ACM Comput Surv* **41**, 15:1–15:58 (2009).

21. Chèneby, J., Gheorghe, M., Artufel, M., Mathelier, A. & Ballester, B. ReMap 2018: an updated atlas of regulatory regions from an integrative analysis of DNA-binding ChIP-seq experiments. *Nucleic Acids Res.* **46**, D267–D275 (2018).

22. Eraslan, G., Avsec, Ž., Gagneur, J. & Theis, F. J. Deep learning: new computational modelling techniques for genomics. *Nat. Rev. Genet.* **20**, 389–403 (2019).

23. Li, Y., Wu, F.-X. & Ngom, A. A review on machine learning principles for multi-view biological data integration. *Brief. Bioinform.* (2016) doi:10.1093/bib/bbw113.

24. Daber, R., Sukhadia, S. & Morrissette, J. J. D. Understanding the limitations of next generation sequencing informatics, an approach to clinical pipeline validation using artificial data sets. *Cancer Genet.* **206**, 441–448 (2013).

25. Teng, L., He, B., Gao, P., Gao, L. & Tan, K. Discover context-specific combinatorial transcription factor interactions by integrating diverse ChIP-Seq data sets. *Nucleic Acids Res.* **42**, e24–e24 (2014).

26. Sharma, N. L. *et al.* The ETS family member GABPα modulates androgen receptor signalling and mediates an aggressive phenotype in prostate cancer. *Nucleic Acids Res.* **42**, 6256–6269 (2014).

27. Lin, C. *et al.* AFF4, a Component of the ELL/P-TEFb Elongation Complex and a Shared Subunit of MLL Chimeras, Can Link Transcription Elongation to Leukemia. *Mol. Cell* **37**, 429–437 (2010).

28. Lin, S. *et al.* Proteomic and Functional Analyses Reveal the Role of Chromatin Reader SFMBT1 in Regulating Epigenetic Silencing and the Myogenic Gene Program. *J. Biol. Chem.* **288**, 6238–6247 (2013).

29. Ponomarenko, N., Lukin, V., Zriakhov, M., Egiazarian, K. & Astola, J. Lossy Compression of Images with Additive Noise. in *Advanced Concepts for Intelligent Vision Systems* (eds. Blanc-Talon, J., Philips, W., Popescu, D. & Scheunders, P.) 381–386 (Springer Berlin Heidelberg, 2005).

30. Theis, L., Shi, W., Cunningham, A. & Huszár, F. Lossy Image Compression with Compressive Autoencoders. *ArXiv170300395 Cs Stat* (2017).

31. Chalapathy, R., Toth, E. & Chawla, S. Group Anomaly Detection Using Deep Generative Models. in *Machine Learning and Knowledge Discovery in Databases* (eds. Berlingerio, M., Bonchi, F., Gärtner, T., Hurley, N. & Ifrim, G.) vol. 11051 173–189 (Springer International Publishing, 2019).

32. Vincent, P., Larochelle, H., Bengio, Y. & Manzagol, P.-A. Extracting and composing

robust features with denoising autoencoders. in *Proceedings of the 25th international conference on Machine learning - ICML '08* 1096–1103 (ACM Press, 2008). doi:10.1145/1390156.1390294.

33. Ranshous, S. *et al.* Anomaly detection in dynamic networks: a survey. *WIREs Comput. Stat.* **7**, 223–247 (2015).

34. Zheng, L., Li, Z., Li, J., Li, Z. & Gao, J. AddGraph: Anomaly Detection in Dynamic Graph Using Attention-based Temporal GCN. in *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence* 4419–4425 (International Joint Conferences on Artificial Intelligence Organization, 2019). doi:10.24963/ijcai.2019/614.

35. *keras-team/keras*. (Keras, 2020).

36. van der Walt, S., Colbert, S. C. & Varoquaux, G. The NumPy Array: A Structure for Efficient Numerical Computation. *Comput. Sci. Eng.* **13**, 22–30 (2011).

37. Newell, A., Yang, K. & Deng, J. Stacked Hourglass Networks for Human Pose Estimation. *ArXiv160306937 Cs* (2016).

38. Quang, D. & Xie, X. DanQ: a hybrid convolutional and recurrent deep neural network for quantifying the function of DNA sequences. *bioRxiv* 032821 (2015) doi:10.1101/032821.

39. Mishra, N., Rohaninejad, M., Chen, X. & Abbeel, P. A Simple Neural Attentive Meta-Learner. *ArXiv170703141 Cs Stat* (2018).

40. Lehtinen, J. *et al.* Noise2Noise: Learning Image Restoration without Clean Data. *ArXiv180304189 Cs Stat* (2018).

41. Kingma, D. P. & Ba, J. Adam: A Method for Stochastic Optimization. *ArXiv14126980 Cs* (2014).

42. Malhotra, P. *et al.* LSTM-based Encoder-Decoder for Multi-sensor Anomaly Detection. *ArXiv160700148 Cs Stat* (2016).

43. Simonyan, K., Vedaldi, A. & Zisserman, A. Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps. *ArXiv13126034 Cs* (2014).

44. Chèneby, J. *et al.* ReMap 2020: a database of regulatory regions from an integrative

analysis of Human and Arabidopsis DNA-binding sequencing experiments. *Nucleic Acids Res.* **48**, D180–D188 (2020).

Acknowledgements

The authors wish to thank Lionel Spinelli for helpful discussion and valuable feedback in writing this manuscript.

Additional information

Authors were supported by recurrent funding from Aix Marseille Université and INSERM. QF, JC PhD Fellowships from the French Ministry of Higher Education and Research (MESR).

Authors contribution statement

Q.F. designed and implemented the method and analyzed the results.
J.C. helped analyze the results and contributed feedback on the method.
C.C. heavily contributed to the design of the method.
B.B. and D.P. helped with the problematic and analysis and presentation of the results.
All authors reviewed the manuscript.

Competing interests statement

The authors declare no competing interests.

Main figures

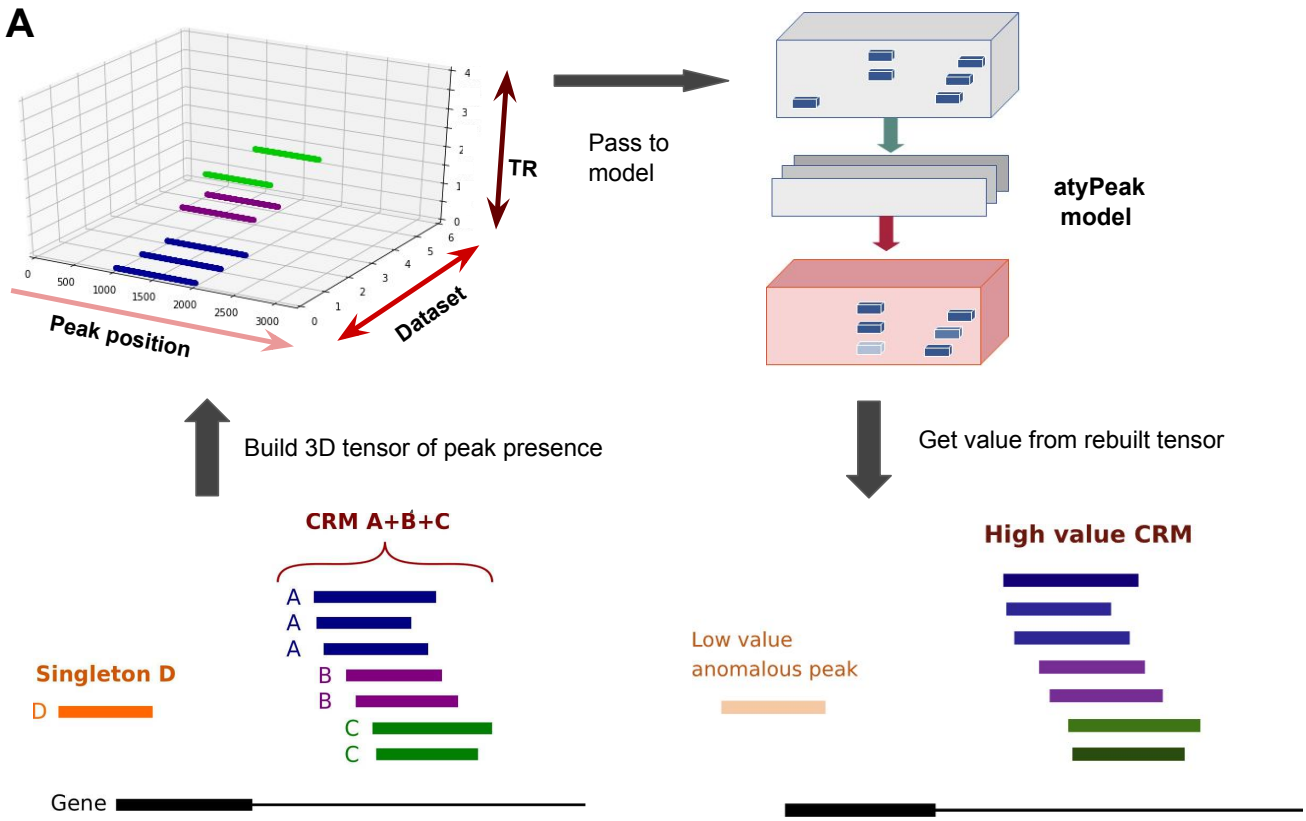


Figure 1A : Workflow and model description. Once candidate regions (ReMap-identified CRMs) are set, we build tensors of peak presence representing them. The X axis represents the position along the genome, while the Y and Z axis are dataset and TR identifiers respectively. The tensor has a value of 1 is a peak for this TF in this dataset (ie. for this source) is present, 0 otherwise.

The atyPeak model will lossily compress this representation. This will result in losing anomalies and other finer details, by learning correlation groups for the rebuilding instead of individual peaks. At the end, each peak is given an anomaly score corresponding to the mean autoencoder reconstruction error, the difference between the original (grey) and rebuilt (red) representation. Scores are then added to the original BED file.

Full source code and documentation are available at <https://github.com/qferre/atypeak>

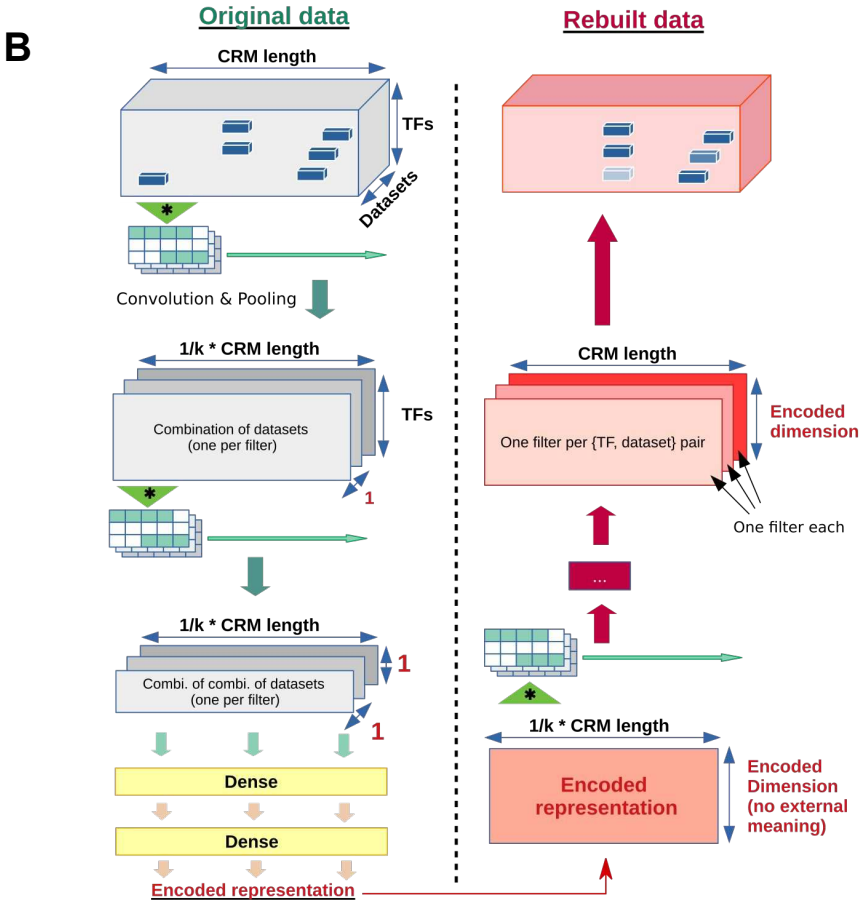


Figure 1B : Model structure. During the encoding, the CRM are viewed by the model through convolutional filters to focus on the correlations between datasets and then between TRs. We use two type of filters (combinations of datasets, then combinations of TRs) successively in a stacked multiview approach. After the subsequent Dense layers, we obtain a smaller encoded representation. This encoded representation is fed to a convolutional decoder with several layers, trying to rebuild the original CRM representation.

In subsequent figure legends, “deep dimension” is the number of neurons in each Dense layer, while the “filters number” is the number of kernels in each Convolutional layer, and LR is the learning rate of the Adam optimizer. More details about the structure are available in Methods.

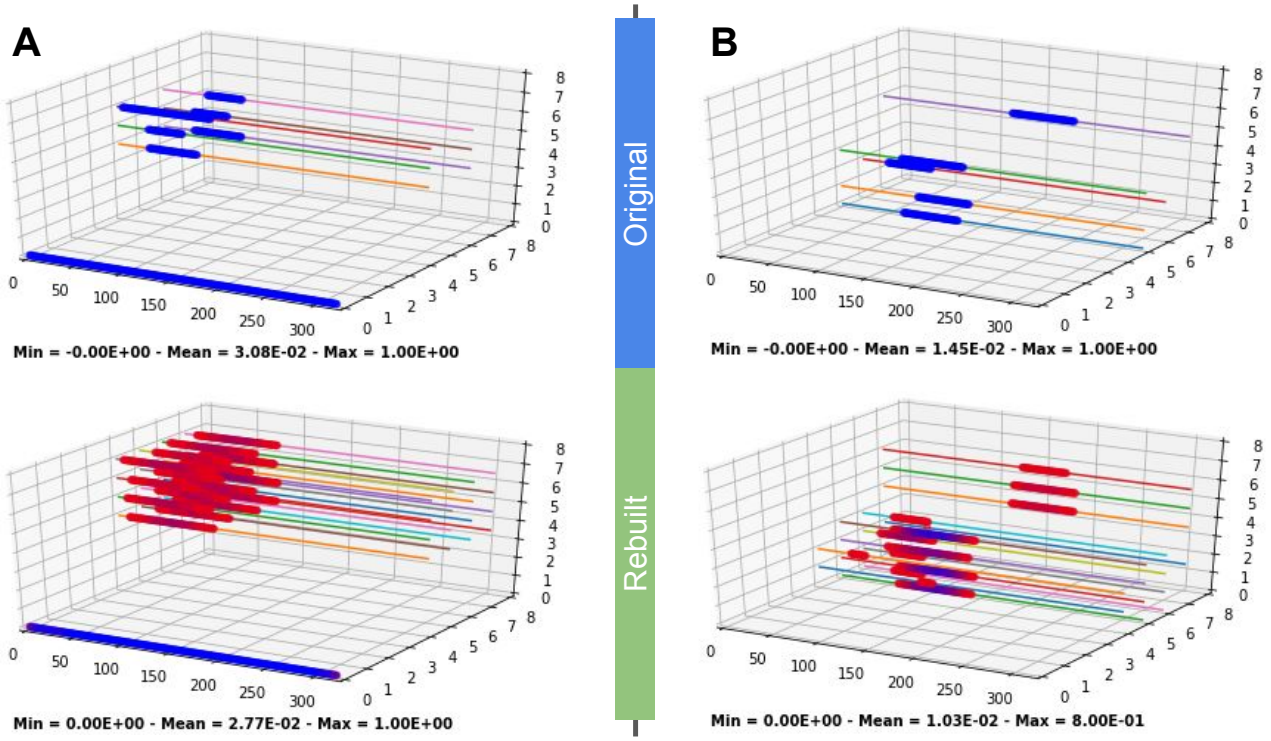


Figure 2 : The atyPeak model learns correlation groups. In each case, the tensor at top is the original representation and the bottom one is what is rebuilt by the model. The model was trained on artificial data. There were 2 predefined correlation groups covering different subsets of dimensions (G1 and G2) defined in Suppl Fig 2. The thin colored lines are only here as a visual aid.

In (A), when the model rebuilds the CRM representation, it rebuilds the entire correlation group when peaks from the group are present. This results in adding the other members that were not originally present as “phantom” peaks. In this case, it is the G1 group. In (B) however, we used a model with a less aggressive compression (too high information budget) and the rebuilding is too precise, learning smaller, non-significant groups instead of the entire G1 or G2 groups.

Model parameters in A were a deep dimension of 32, 16 filters and a Learning Rate (LR) of 1E-3. B used 48 filters, 256 deep dimension, a LR of 1E-4. Note that for B, that increased precision is not achieved with higher deep dim but default LR - we needed a lower LR. 48 epochs for all or early stopping (for A). Groups were equiprobable.

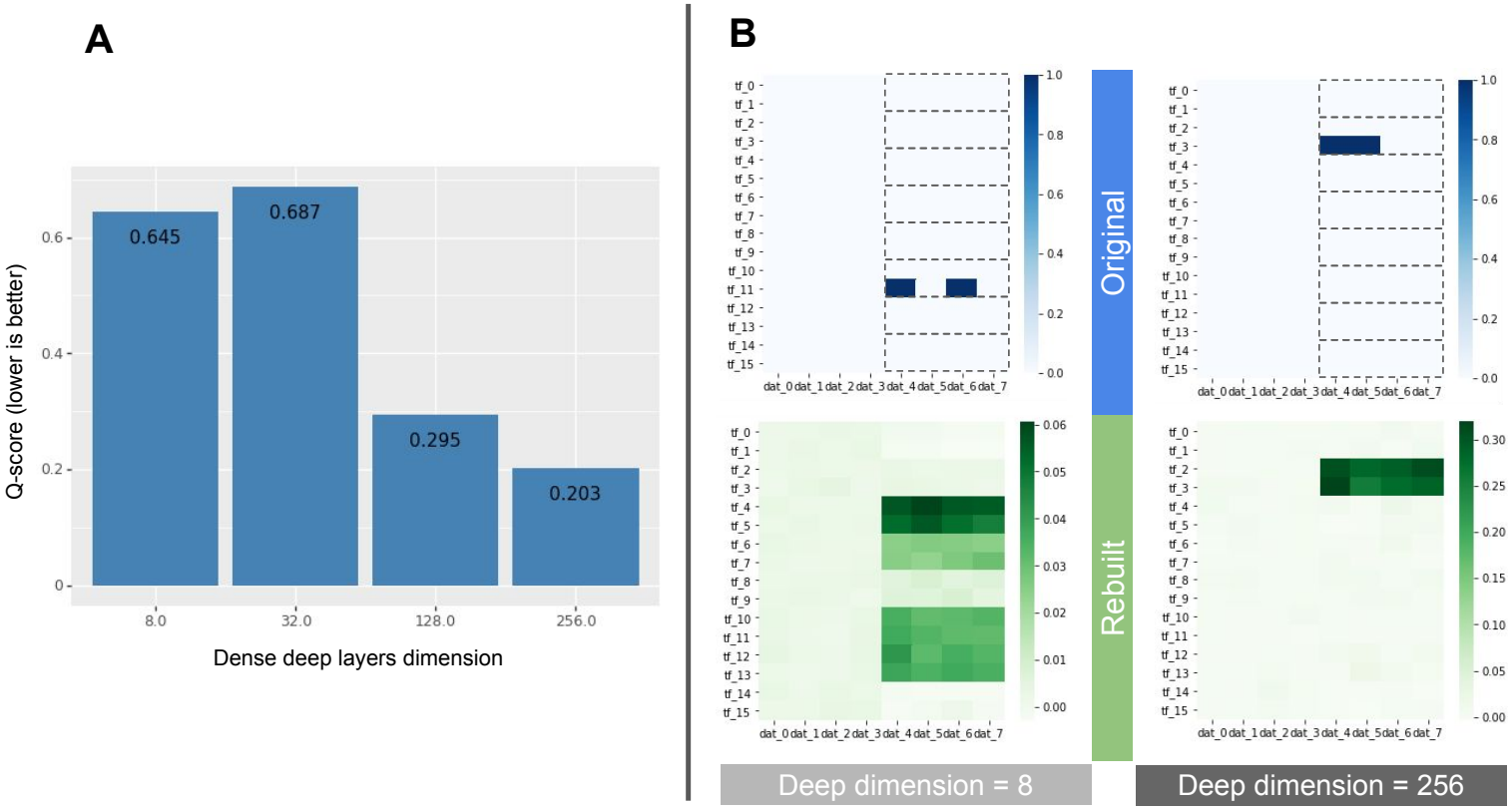


Figure 3 : Scaling of the information budget with the data. We used artificial data of dimension 8x16, but the TRs are subdivided into 8 groups instead of 2 like in Suppl. Fig. 2 (ie. there the TR 0 and 1 are a group, then 2 and 3 are another group, etc. up to 14 and 15. The groups are visually reminded on the figure as one grey box per group). At data generation, the stack is placed in one of the 8 groups. All 8 groups are equiprobable. The model parameters were 24 convolutional filters and a LR of 1E-4. The number of neurons in the Dense layers changes during the grid search.

With lower deep dimensions (and so a lower information budget), the model is unable to learn separately the 8 existing correlation groups (B left) and will instead learn fewer and larger groups. A larger budget was needed to learn the 8 groups (B right). This highlights how the information budget must be adapted to the quantity of information in the data for a satisfactory result. Note that for this larger data, hundreds of neurons are required, compare to smaller models for the smaller data of Figure 2. To help choose the budget, we propose a Q-score to quantify the quality of the rebuilding depending on the budget. This score assesses how well the model learns each existing pairwise correlations. More details about the Q-score of the models involved in this figure is presented in Suppl Fig 5.

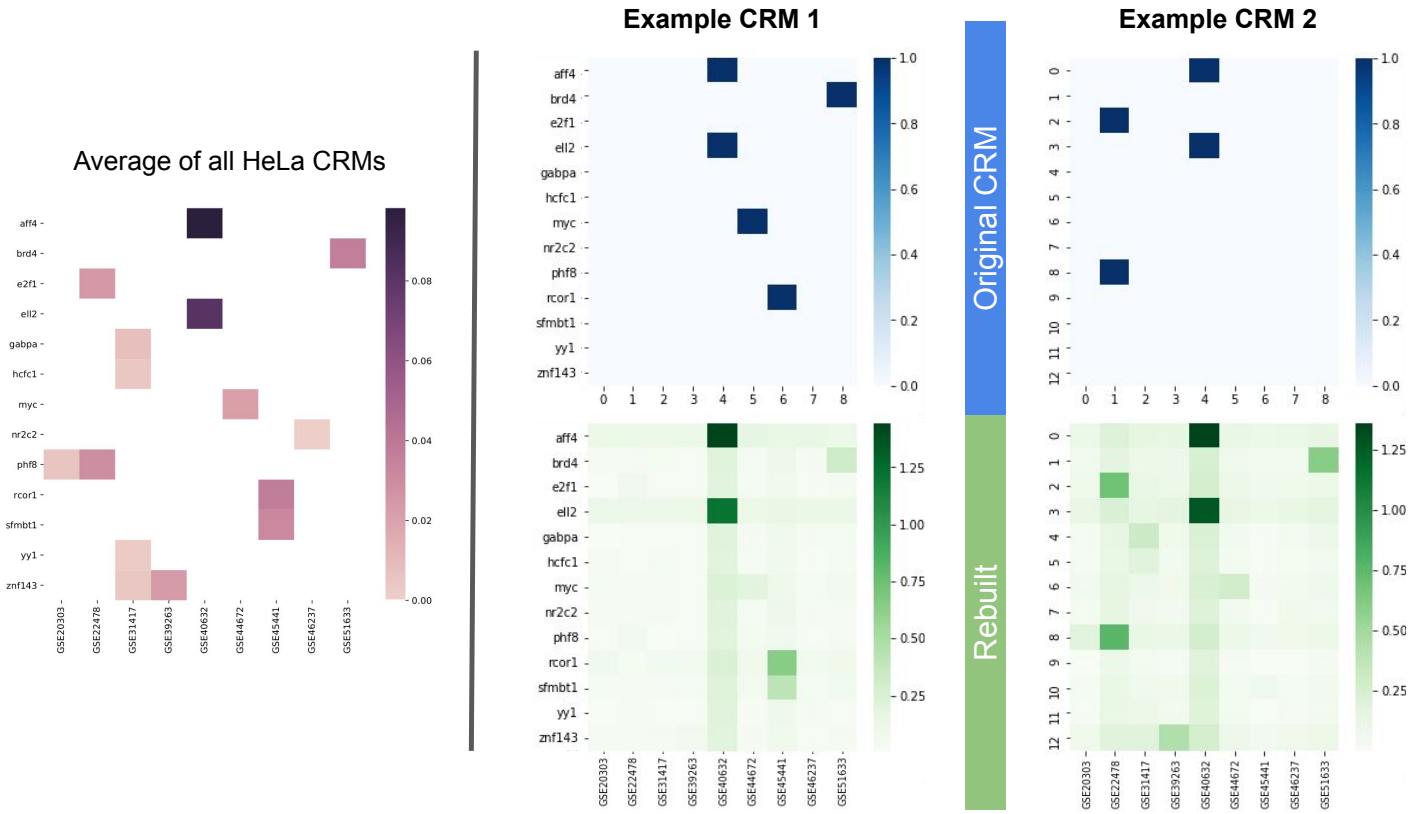


Figure 4 : Two different examples of real CRMs rebuilt in HeLa. All figures give the mean across X axis (region size) of the tensors. The model's parameters are detailed in Supplementary Table 1. As with the other figures, the blue heatmap represents the original representation of the CRM, with the green heatmap giving the rebuilding by the model. The average of all HeLa CRMs is provided for comparison.

The model has visibly learned different correlation groups, and not just rebuilt the average CRM. We can see, notably for BRD4, that an incomplete group results in lower scores, and that phantoms are added to complete the learned groups. Some learned groups are extracted and presented in Suppl Fig 11 for comparison.

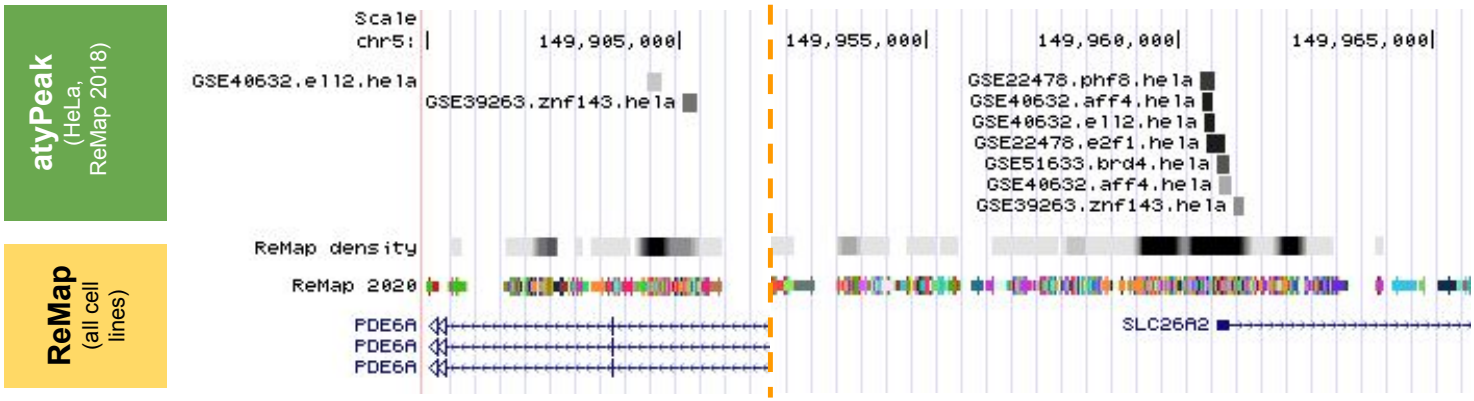


Figure 5 : Example of visualisation of atyPeak results in the UCSC genome browser. The results presented here are for the HeLa cell line for ReMap 2018. A darker peak indicates a higher atyPeak score. The annotated BED data files with the corresponding atyPeak scores are available at <https://github.com/qferre/atypeak-files> or as a UCSC browser session at <http://genome-euro.ucsc.edu/s/qferre/atyPeak_hg38>.

We can see on this figure an example of rich CRE with many peaks, and a poorer CRE where many correlators for those TRs are missing which predictably has a lower score. As detailed previously, our approach estimates how “typical” each peak is, with respect to the usual combinations between sources (TRs and/or datasets) for a given cell line. As the model is unsupervised, anomaly score thresholds are at the user’s discretion. For example, a large scale analysis might exclude the lowest scoring peaks, but a focused study of a single or selected experimental series may specifically seek low-scoring peaks that might be caused by certain events of interest (mutations, etc.). It is also possible to use high-scoring peaks to detect CREs of interest and use that selection as a filter when looking at other genomic data, like we show here with ReMap 2020.

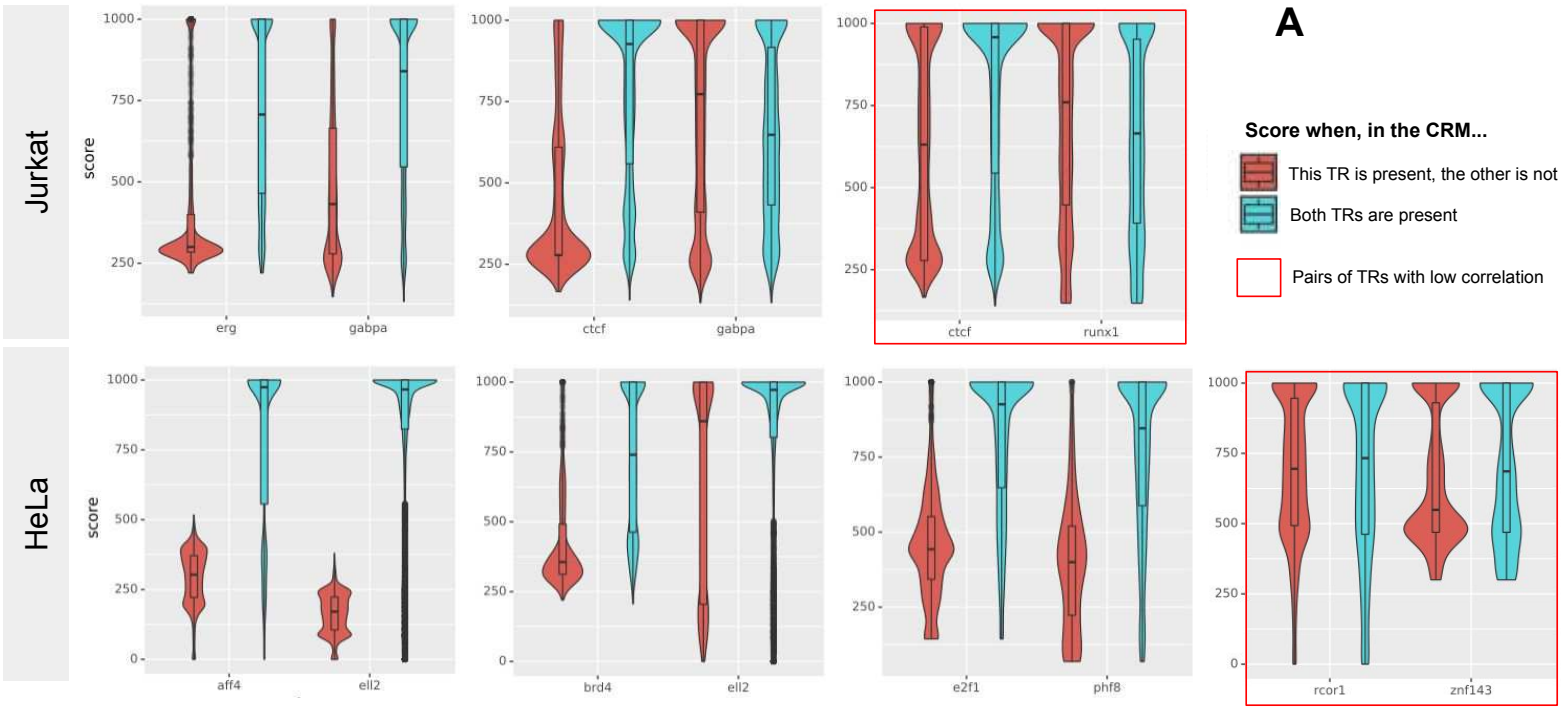


Figure 6A : Confirmation of the biological meaningfulness of identified correlations. Scores are considered after applying all normalizations described in Methods. We consider pairs of TFs. For each pair {A,B} we give the score of peaks from A when B is present too in the same CRM (blue) or when B is absent (red). This is the same elementary operation as the Q-score, except we do not average across the X axis but take the actual peak value.

Most examples presented are of TRs with high correlation, such as GABPA and ERG in Jurkat which have many common binding sites, ELL2 and AFF4 in Hela, or RCOR1 and SFMBT1 in Hela which are both repressors. When TFs correlate, our model will have learned that and assign higher scores to peaks for a TF when one of its correlators is present. We also provide some counter-examples: CTCF and GABPA in Jurkat have a R coefficient of 0.2 which is high for CTCF but low for GABPA (GABPA is often seen with CTCF, but CTCF has other partners than GABPA) and as such the impact on the score is also unidirectional. Finally the pairs framed in red such as CTCF and RUNX in Jurkat or RCOR1 and ZNF143 in HeLa have a low correlation coefficient. For them, the presence of one TR of the pair has little to no impact on the score of the other.

For cases such as AFF4 and ELL2 in HeLa which have one major correlator (namely, each other), the distributions of all scores (blue and red merged) is rather bimodal, as the presence of the other acts as a binary switch.

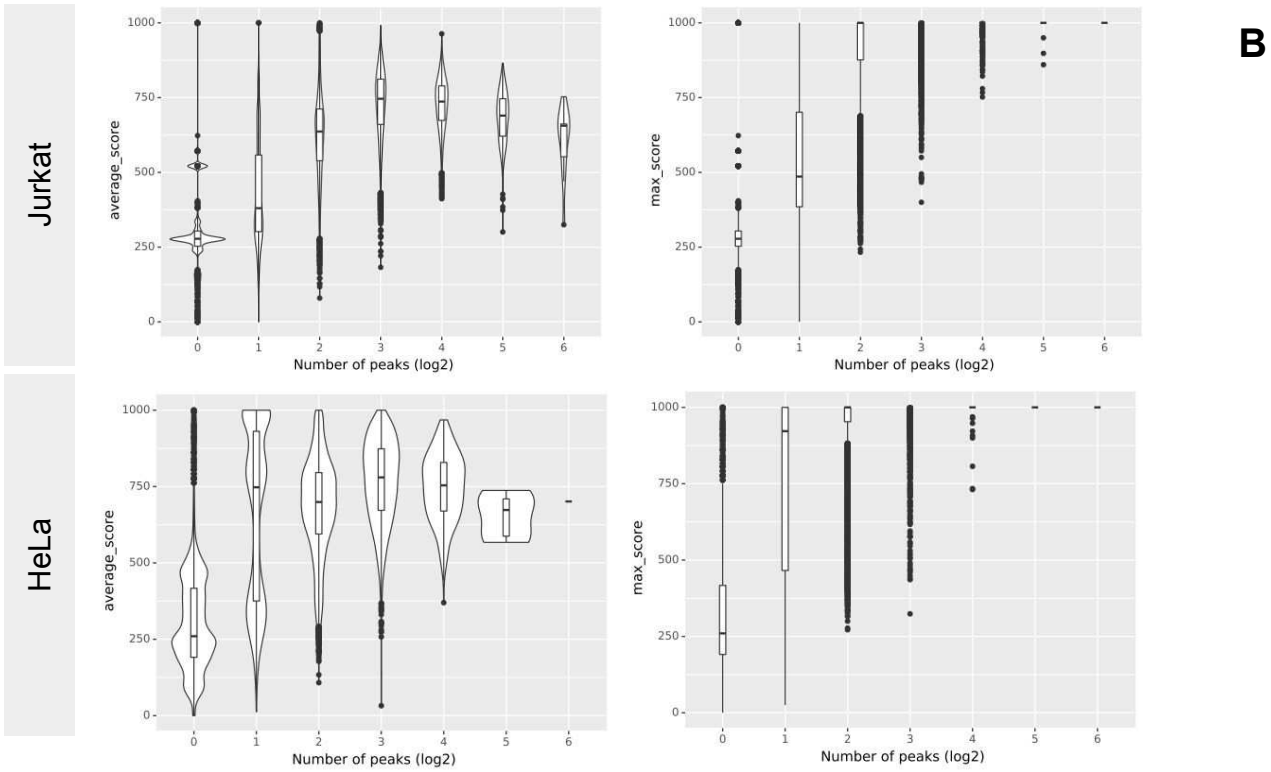
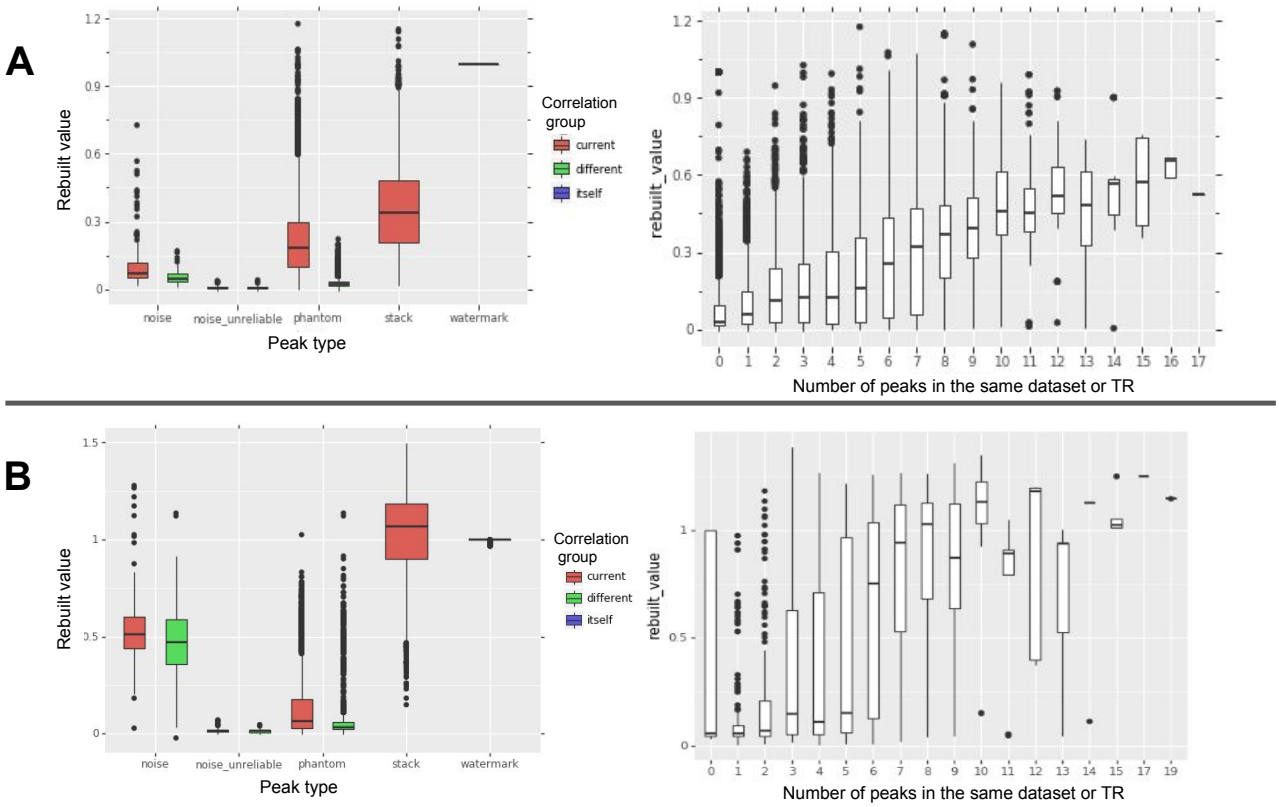


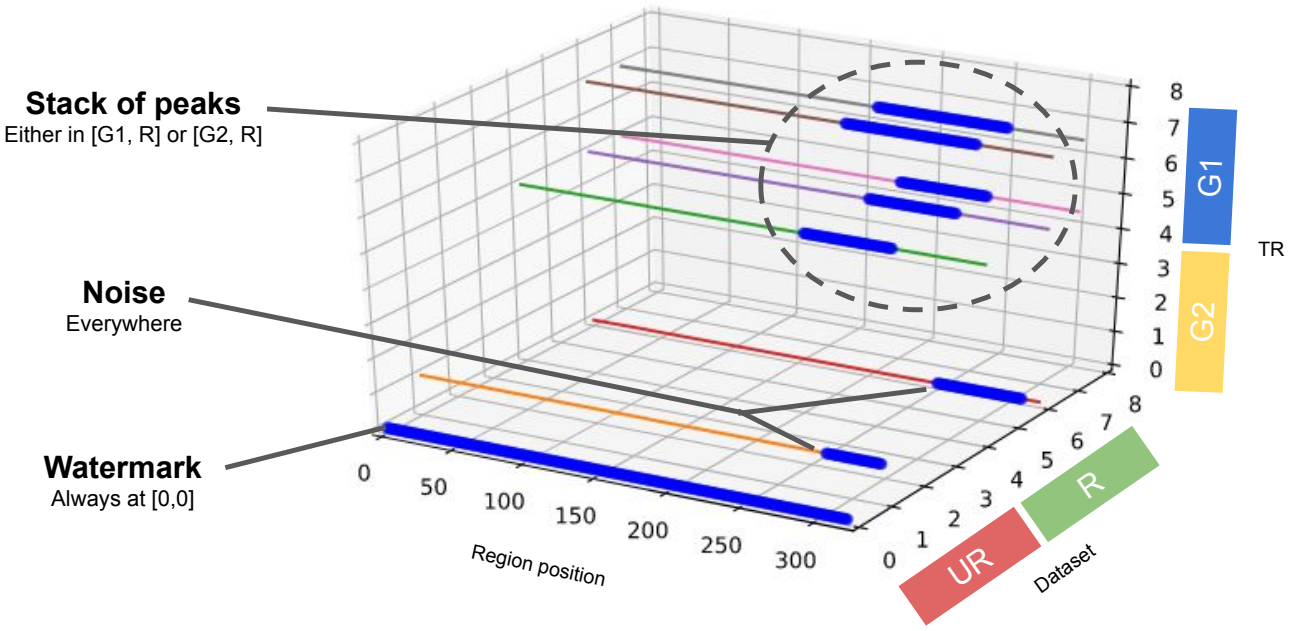
Figure 6B : For each processed CRM, average (left) and maximum (right) score of the peaks present in it, depending on the total number of peaks in the CRM. Number of peaks given axis in log2 scale.

As Transcriptional Regulators tend to work in complexes, it makes sense that richer CRMs would be on average of better quality. However, the relation is not strictly linear: CRMs with supernumerary peaks likely contain noise, which is reflected here in a lower average score.

Supplementary figures



Supplementary Figure 1 : Systematisation of artificial data analysis in 10 thousand CRM. For both (A) and (B) we use a model with deep dimension of 32, 16 convolutional filters, and LR of 1E-3. We compare a situation where such a model is too precise in (B) where the artificial data dimensions are 6x4“ with a situation in (A) where such a model is adequate.The left plot gives the distribution of rebuilt (max across) values of peaks depending on their type: respectively noise in reliable (R) datasets, noise in unreliable (UR) datasets, phantoms (peaks added that were not present in original matrix) and stack (peaks that were in the stack of added peaks). See Suppl Fig 2 for details. The color gives the correlation group of the peak (belonged to the same group as the group where the stack of peaks was placed for this CRM, or different). “Brothers” is the total number of peaks in same line or same column (summed). In both cases, the stack of peaks (and watermark) are correctly rebuilt, and phantoms of a high value are added in the same correlation group as the stack, but not in the other group. The correct rebuilding of the watermark shows lonely peaks can still be learned when frequent. When noise is added in the (R) datasets, it will not be part of a stack hence its usual correlators will not have been added : lacking its correlators, it it atypical by our definition and gets a lower value due to this, not just because it is lonely. Noise in (UR) is discarded by the model due to its rarity. Noise scores are higher in B as the groups have less members, and a single noisy source represents a larger proportion of the total group learned than in A.



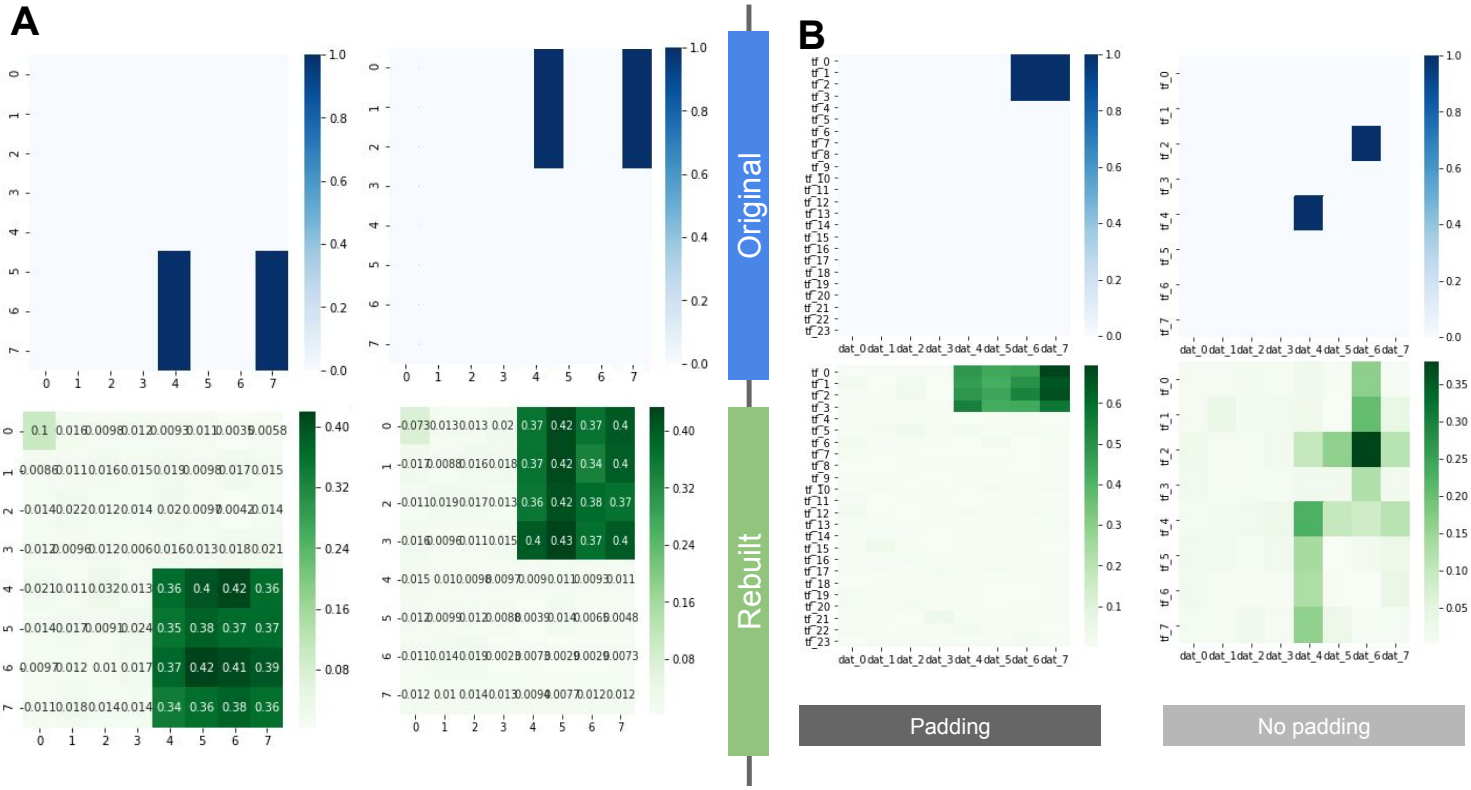
Supplementary Figure 2 : Example of an artificially generated region representation and relevant groups, in 8x8 dimension. The small colored lines are only visual aids.

The X axis (bottom left) is the position along the region, the Y axis (bottom right) is the dataset number and the Z axis (right) is the Transcriptional Regulator number. Datasets are split in half between “Reliable” (R) which will contain both the stack of true peaks and noise, and “Unreliable” (UR) which will contain only noise. TFs are split in the G1 and G2 groups. They can optionally be split in more than two groups.

We first place a stack of peaks around a common position. These peaks will belong to either the G1 or G2 group. As a result, sources within the G1 or G2 group will correlate with each other, but will not significantly correlate with sources from outside their group. We then add noise uniformly randomly that can belong to any dataset and TR, representing anomalies. Finally a control watermark peak is added with, usually, 75% probability, representing ubiquitous TRs. All values in the tensor are 1, denoting presence. More details are available in Methods.

Cell line	TRs	Datasets
Jurkat	brd4, cdk7, ctcf, erg, fancl, gabpa, gata3, med1, myb, myc, runx, runx1, tal1, tal1_scl, tcf12, tcf3, znf335	GSE: 17954, 25000, 29180, 42575, 45864, 49091, 50622, 59657, 68976, 76181, 83116, 83777
HeLa	aff4, brd4, e2f1, ell2, gabpa, hcfc1, myc, nr2c2, phf8, rcor1, sfmbt1, yy1, znf143	GSE: 20303, 22478, 31417, 39263, 40632, 44672, 45441, 46237, 51633
K562	atf1, cebpb, ctcf, ep300, fos, fosl1, gata1, gata2, irf1, jun, junb, jund, max, myc, nrf1, rad21, rest, spi1, stat1, yy1	GSE: 70482, 70764, 74999 ENCSR000: AKO, AQB, ATM, BGW, BKM, BKU, BKV BLP, BMH, BMV, BMW, BPJ, BRQ, DJX, DJY, DKA, DKB, DLZ, DMA, DNZ, DWE, EFS, EFT, EFV, EGE, EGH, EGJ, EGK, EGM, EGN, EGS, EGT, EGU, EGY, EHE, EHJ, EHK, EWF, EWG, EWM, EZT, EZU, EZV, EZW, EZX, FAD, FAE, FAG, FAH, FAI, FAU, FAV, FAZ ENCSR: 091GVJ, 137ZMQ, 159OCC, 239ZLZ, 494TDU, 795IYP, 837EYC, 854MCV, 998AJK
MCF7	ahr, ar, brd4, ctcf, ep300, esr1, foxa1, foxm1, gata3, hsf1, jun, jund, max, med1, myc, ncoa1, ncoa2, ncoa3, rad21	GSE: 35109, 38901, 40129, 40762, 41561, 41820, 45822, 45852, 48930, 51274, 54855, 55921, 59530, 60270, 68355, 68356, 70764, 71276, 72082, 72249, 80808 ENCSR000 : AHD, BST, BSU, BTQ, BTR, BUJ, BUL, DMJ, DML, DMM, DMO, DMP, DMQ, DMR, DMS, DMV, DWH, EWS, EWV ENCSR062HDL, ENCSR176EXN ERP000: 209, 380, 783, 901 ERP001226, ERP002305
CD34	gata1, gfi1b, kmt2a, mllt3, myc, notch1, runx1, tal1	GSE: 52924, 54344, 63010, 64862, 85488
ESC	brd4, ctcf, ep300, ezh2, lef1, myc, nanog, nipbl, pdx1, pgr, pou5f1, smad3, sox2	GSE: 13084, 17917, 18292, 20650, 29422, 33281, 58685, 64758, 69479, 69539, 75297 ENCSR264RJX

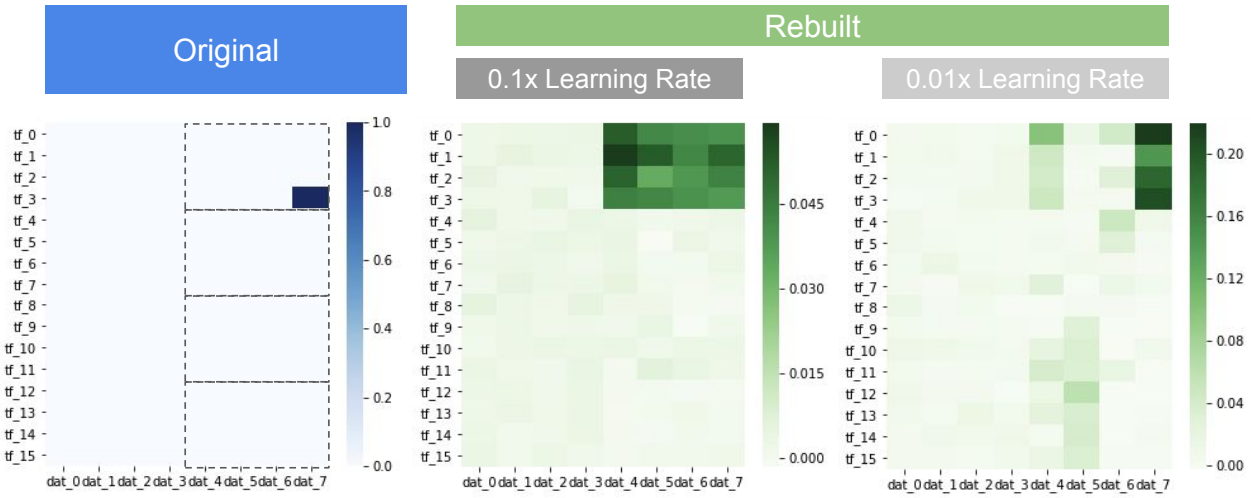
Supplementary Table 1 : List of datasets and Transcriptional Regulators from ReMap 2018 used in this study. Datasets IDs are grouped by prefix in the table: for example, “GSE52924” and “GSE54344” are grouped as “GSE: 52924, 54344”. For the cell lines, if variants are present in the ReMap data, they are not kept. For example, CD34 does not include CD34_condition1.



Supplementary Figure 3 A and B : Learning biases and budgets.

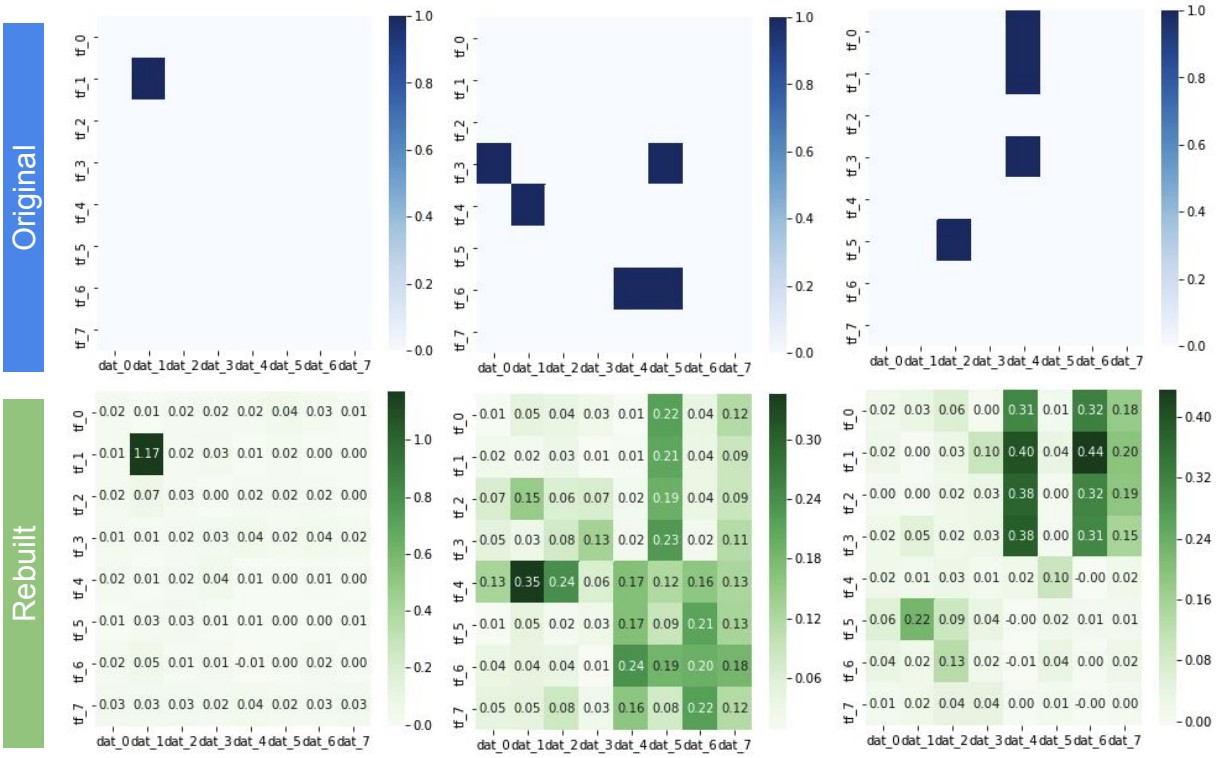
(A) Using artificial data with group selection odds for G1 and G2 set at ⅔ and ⅓ instead of equal. The values given in rebuilding are still only dependant on group completeness, the difference in abundance between the two groups does not influence the result. The model had 16 filters, deep dimension of 32 and LR of 1E-4.

(B) Using two equiprobable non overlapping groups, as per Supplementary Figure 2. The only difference between “padding” and “No padding” is that a padding of 12 lines (TFs) of zeroes were added to the matrices passed to the model. The model had 96 filters, 256 deep dimension, LR of 1E-4. This shows that even where there is no new information (in the left, the two groups G1 and G2 are still in the two top-rightmost 4x4 blocks), the precision is lower for the same model when the data dimensions are larger.



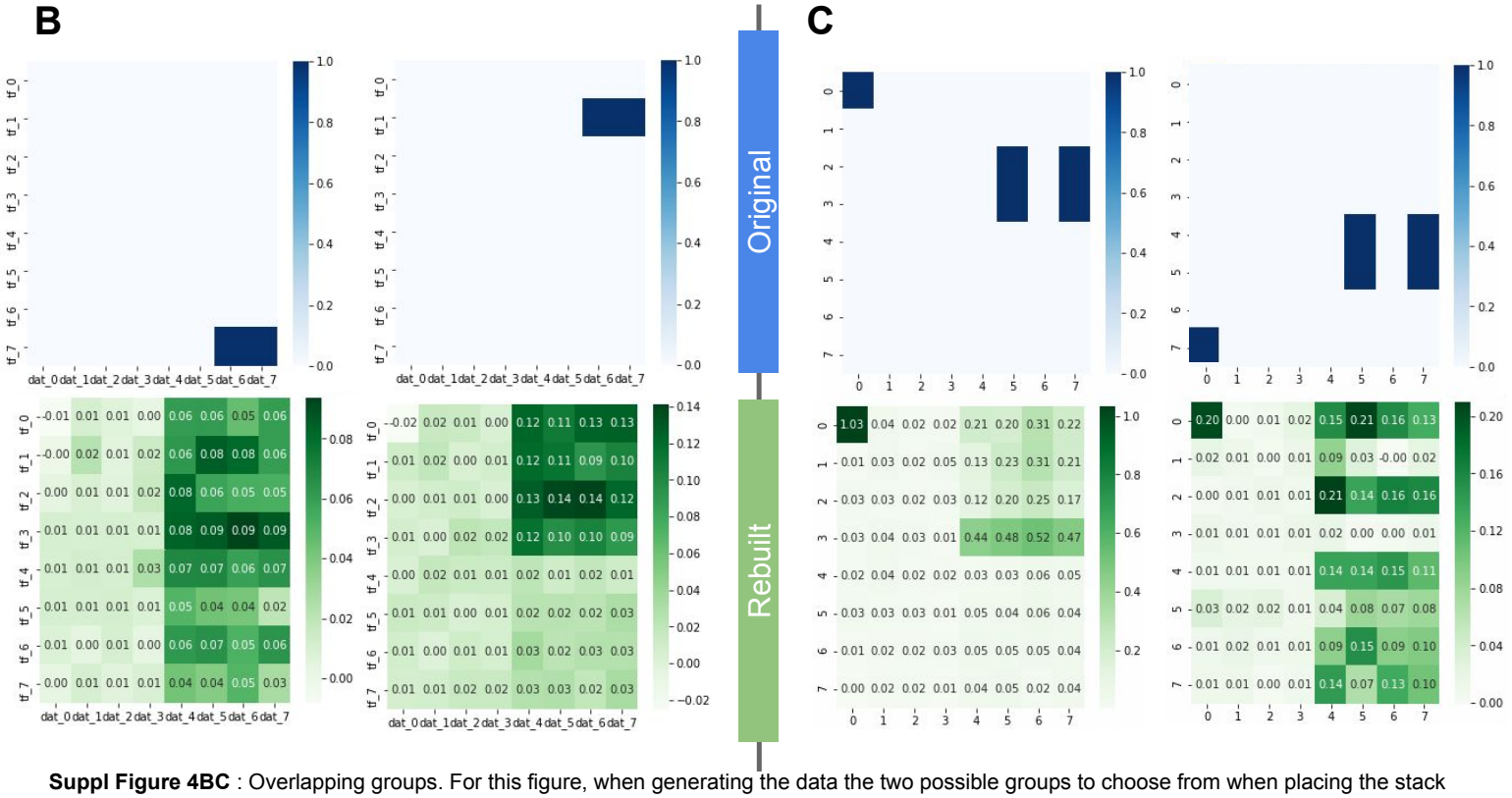
Supplementary Figure 3C : Model trained with artificial data, with 4 correlation groups, 64 filters and 600 deep dimension. The correlation groups predefined at data generation are reminded by the dotted lines.

In spite of the very large information budget of the model, a LR of 1E-4 was not enough to reach an over-precise learning. Overprecision was achieved only with a much lower LR of 1E-5, which demonstrates that to reach increased precision (less aggressive compressions) lower LRs are more of a necessary condition than large deep dimensions.



Suppl Figure 4A : Usage of a weighted loss. We use a model with 24 filters, deep dimension of 64 and a LR of 1E-4. We use artificial data with the same generation process as usual, as detailed in Suppl Figure 2. However, we assign a weight of 10 in the loss to all UR datasets (0 to 4 included), which means that when computing the loss errors on these dimensions count 10x as much.

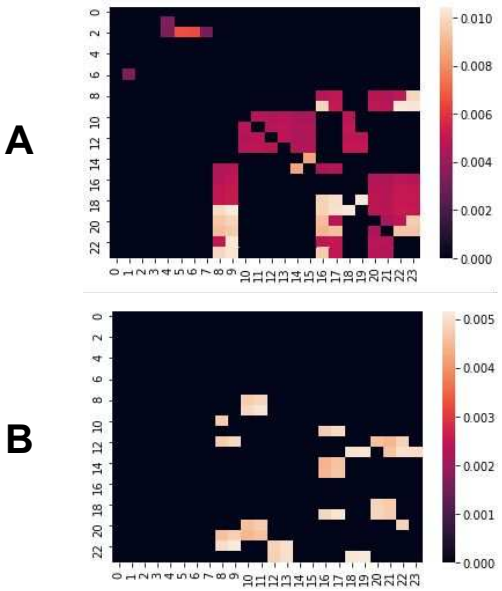
Similar models without weighting would entirely discard peaks from suich UR datasets (Figure 2), but now they are now learned with high precision, almost individually, while the precision is not as great for the non-weighted sources. This highlights the role of weighting in directing the learning towards specific sources and on the process of learning in general.



Suppl Figure 4BC : Overlapping groups. For this figure, when generating the data the two possible groups to choose from when placing the stack were not “G1” and “G2” but “G1” and “G1+G2 = all TRs”. This means the second groups overlaps with the first, and in fact contains all its sources plus its own exclusives. Both groups had 50% odds of being selected.

Such overlapping groups are hard to learn and needs careful parametrization, as we explain in Methods. This requires use of 2x weighting for the sources of G2, and early stopping at 16 epochs. In B, the overlapping groups are learned properly and we see that G2 produces phantoms for G1. The model used had 16 filters and 32 deep dimension and a LR of 1E-4.

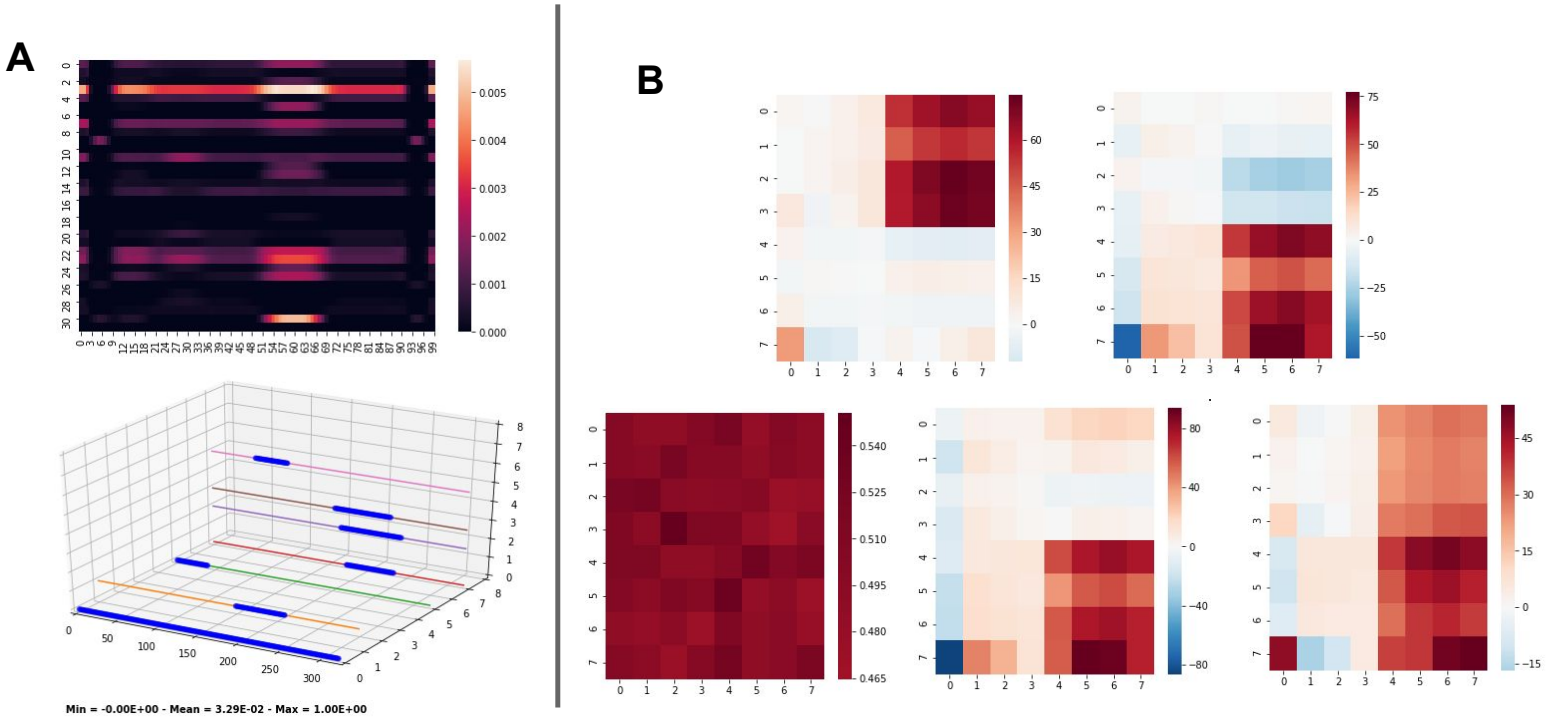
C is an example of difficulties that can be encountered. Done with a model of 24 filters, 64 deep dimension, a higher LR of 1E-3 and crucially, no weighting. The sources of G2 still produce some phantoms for the sources in G1, but those are much fainter, and the rebuilt groups are not homogeneous. Note that a lower LR of 1E-4 for this LR resulted in increased precision as would be expected, with more precise groups for G1 and no overlapping phantoms.



Supplementary Figure 5 : Q-score matrices giving the contributions for each pairs of dimensions for two models from Figure 3. Lower is better.(A) corresponds to the model with 8 Deep dimension and (B) to the model with 256 deep dimensions. The numbers of both the X and Y axis have the same significance : 0-7 represent datasets 0 to 7, and 8-23 represent the TRs 0 to 15.

The Q-score assesses, for each couple of dimensions (datasets with datasets and TRs with TRs) if the presence of one results in a higher score when present for the other, or in higher phantoms. The better model has lower Q-score, as the 8 groups were learned properly.

The Q-score is currently a work in progress but is informative as to the larger trends ot learning. More details are presented in Methods.



Suppl Figure 6 : Example of interpretability in artificial data.

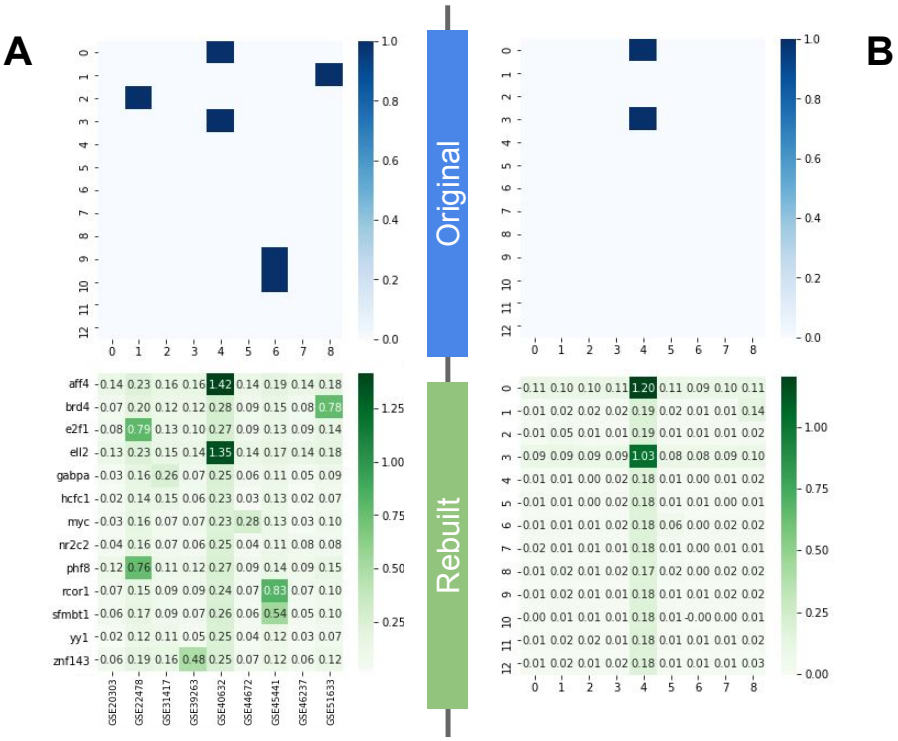
(A) The bottom figure presents an example of artificial tensor on the bottom, and the top heatmap gives the squared activation of encoded dimension when this representation is passed to the model. The parameters used are the same as FIgure 2A.

(B) gives several ur-examples (average across X axis) of CRMs that would maximally activate one row in the encoded dimension (rows in top left). The focus is mostly on the correlation groups as a whole. This is useful as a focus map to see where the model focuses its learning, but the final rebuilt tensor is not simply a linear combination of those, as evidenced by the fact that some ur-examples focus on both correlation groups. Some dimensions are never used, and redundancies were observed. Note that watermark is visible in those ur-examples only when it is not added 100% of the time (and therefore is a variable and not a constant).

Cell line	CD34	Jurkat	MCF7	K562	ESC	HeLa
Learning rate	1E-3	1E-4	5E-5	5E-5	1E-4	1E-4
Early stopping	0.003	No	No	No	No	No
Number of convolutional filters	16	32	64	128	32	32
Dense deep dimension	64	256	1024	2048	256	256
Number of nonzero sources (TF +dataset pairs)	7	22	128	71	17	15

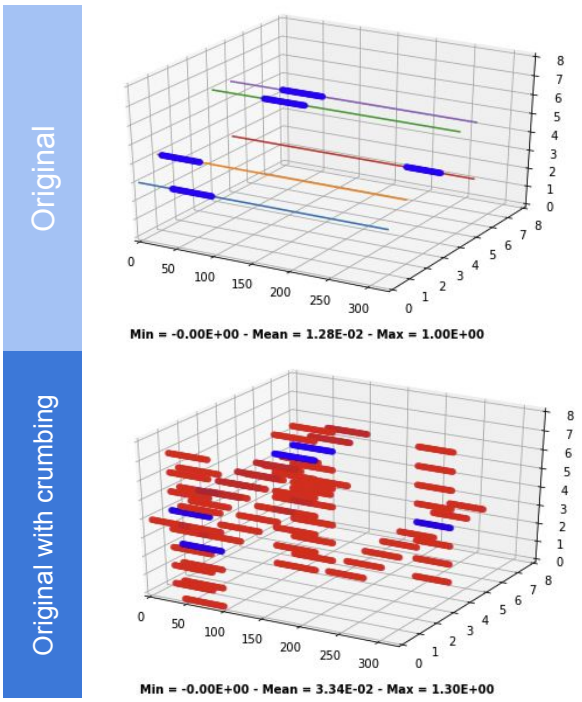
Supplementary Table 2 : Parameters used when processing real ReMap data. This provides a baseline based on the dimensions of the data for our cell lines. Dimensions provided are for comparison purposes, to provide a baseline. Other parameters (regularisation, etc.) have the same value for all cell lines, given in the Methods section in the paper.

The nonzero sources gives the number of sources encountered often enough (at least one in several thousand CRMs depending on cell line) that a normalization coefficient was computed for them.



Suppl Figure 8 : Demonstration of overlapping groups learning by the model. (A) gives a true HeLa CRM and its rebuilding, and (B) is the same CRM after removing all peaks excepted for those belonging AFF4 and ELL2. Even though they are a group almost by themselves (Suppl Fig 11), removing all other peaks results in AFF4 and ELL2 having a lower score in the rebuilding of A than B as BRD4 and others also contribute to their group, even though they are learned in another group. This confirms overlapping groups are possible in real data, but subject to caveats describes in Methods.

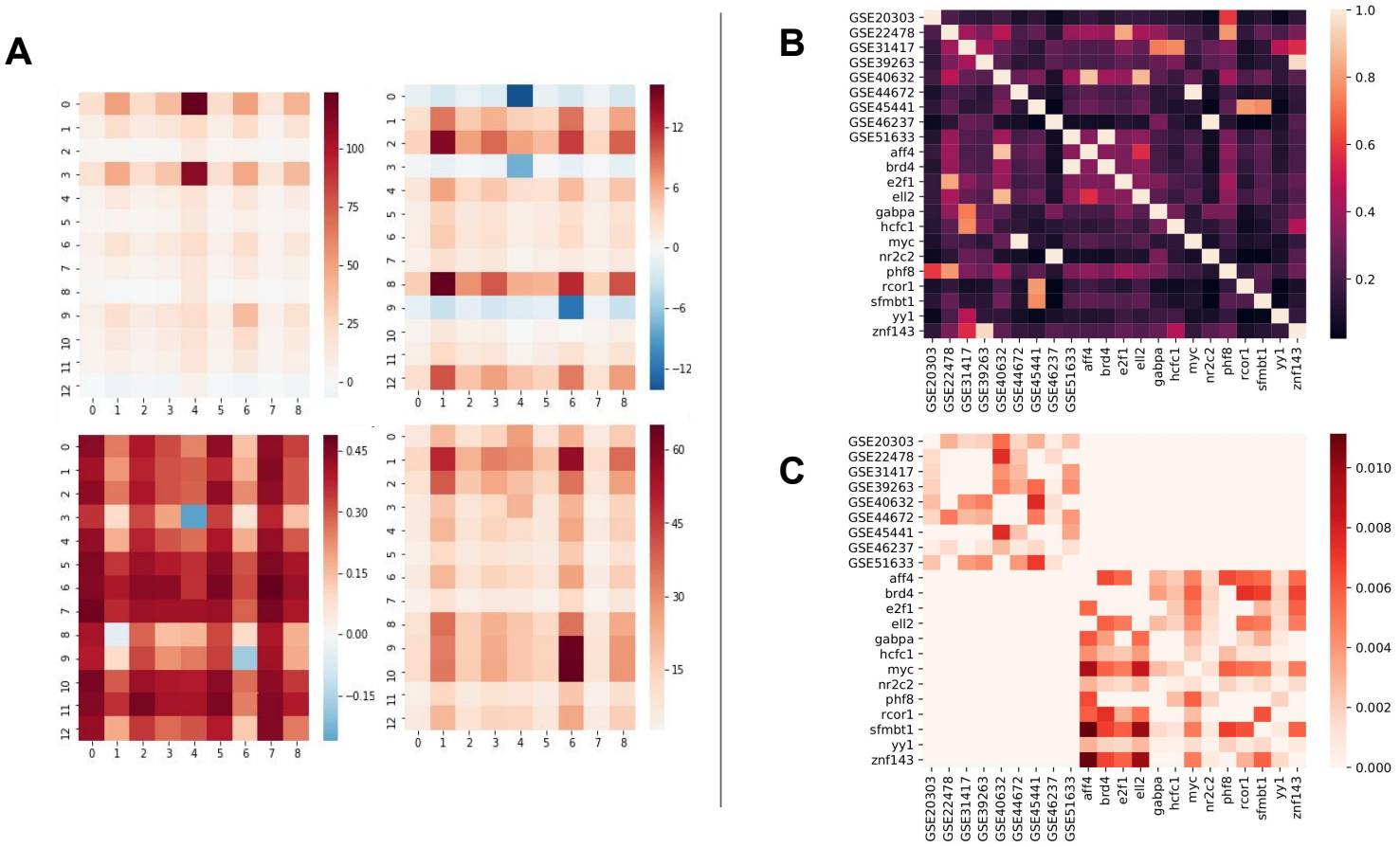
Figures give the maximum across the X axis. Parameters are given in Suppl. Table 2.



Suppl Fig 9 : Crumbing. On the figure, values from low to high and red to blue. Thin lines are a visual aid.

Crumbing is added to real data matrices to fight sparsity. For each nonzero value in the original CRM representation at position [x,y,z], 1/10th of this value is added to all positions at [x,:,z] and [x,y,:], meaning for all datasets sharing the same TR and all TRs sharing the same dataset, forming a “+” pattern.

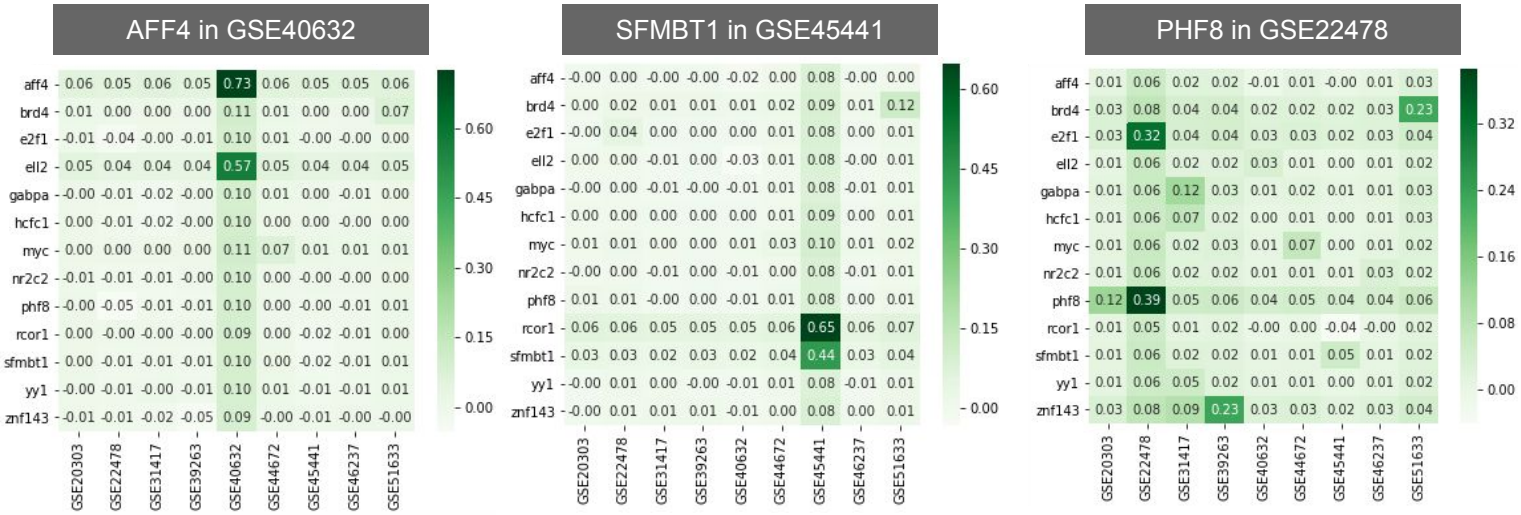
This is necessary because on very sparse data, such as the real data tends to be, the model can easily fall in the learning trap of rebuilding a completely empty tensor.



Suppl Figure 10 : Analysis of a trained HeLa model.

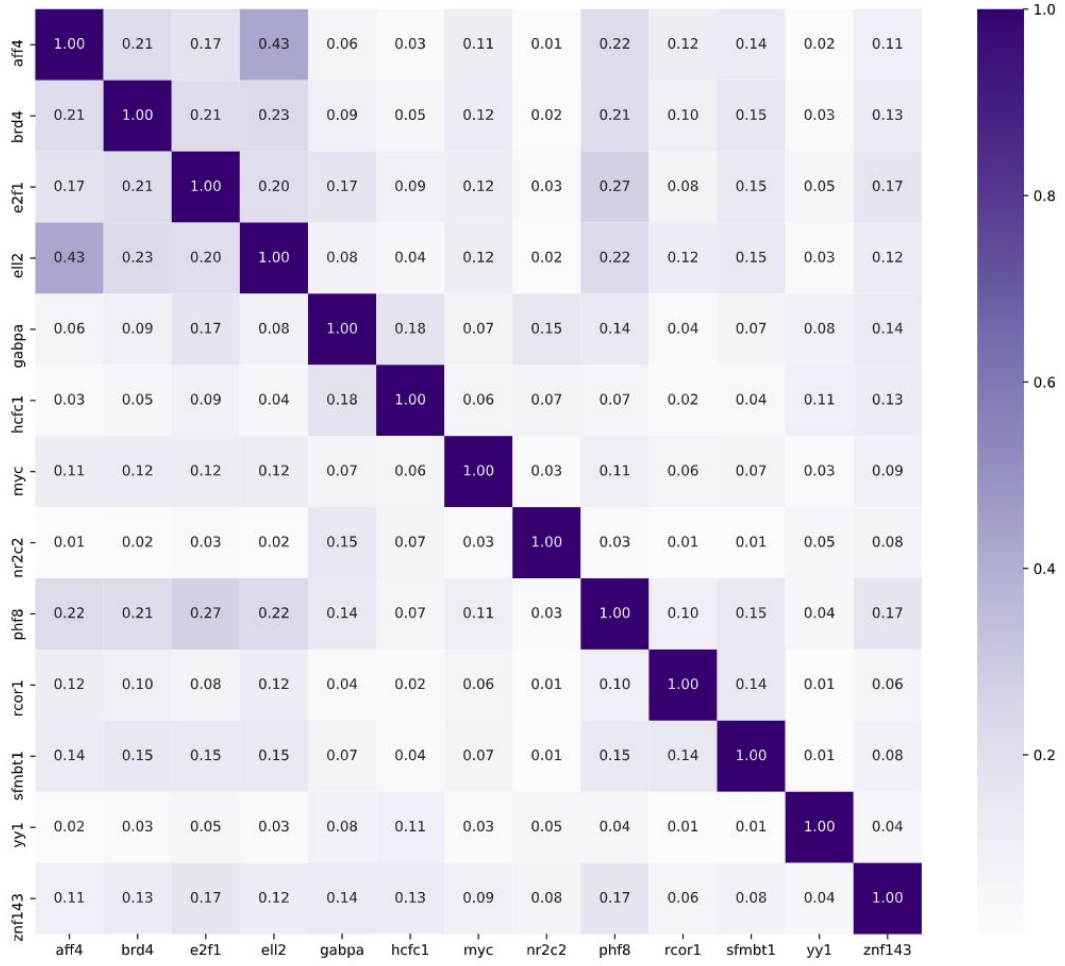
(A) presents some ur-examples (maximally activating CRMs for each element of the encoded dimension) summed across the X axis, calculated on a trained HeLa model. (B) is HeLa correlation matrix between all dimensions like in the Q-score, (C) is the Q-score contributions

We observed that using a higher deep dimension can still help reach a lower loss, even with redundancies in the ur-examples, including on real data.

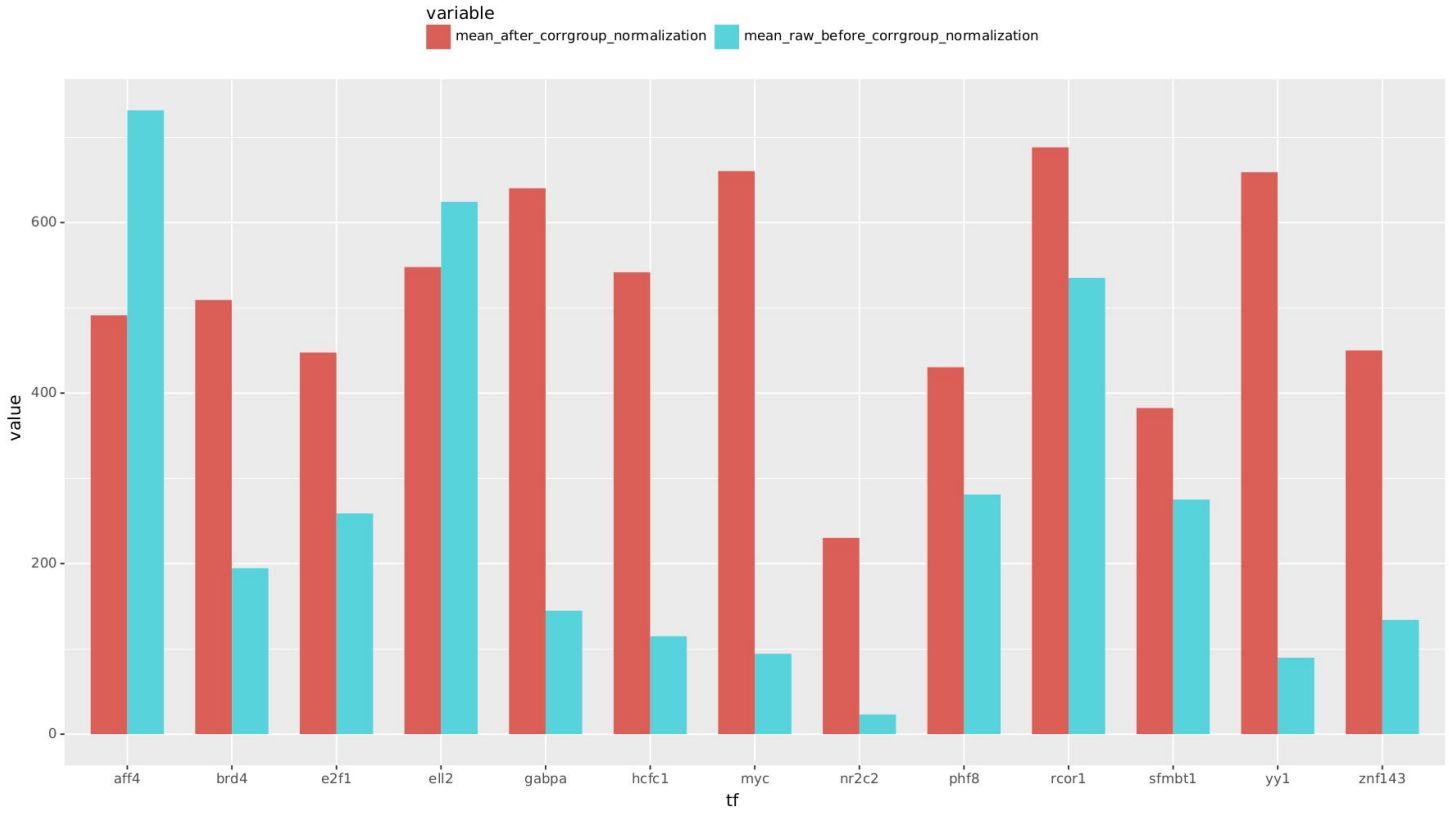


Suppl Figure 11 : Estimating the correlation groups certain sources belong to. This is done in the HeLa cell line, with the legend indicating respectively the Transcriptional Regulator and dataset concerned. As detailed in Methods, for each given source we create an empty CRM representation with a peak along its length for this source only, and pass it to the trained model.. The result, shown above, is the sum across the X axis of the rebuilt tensor.

The cross '+' pattern is due to crumbing, one needs to be mindful of it when interpreting. Note that several sources (BRD4, SFMBT1) are present in more than one group.

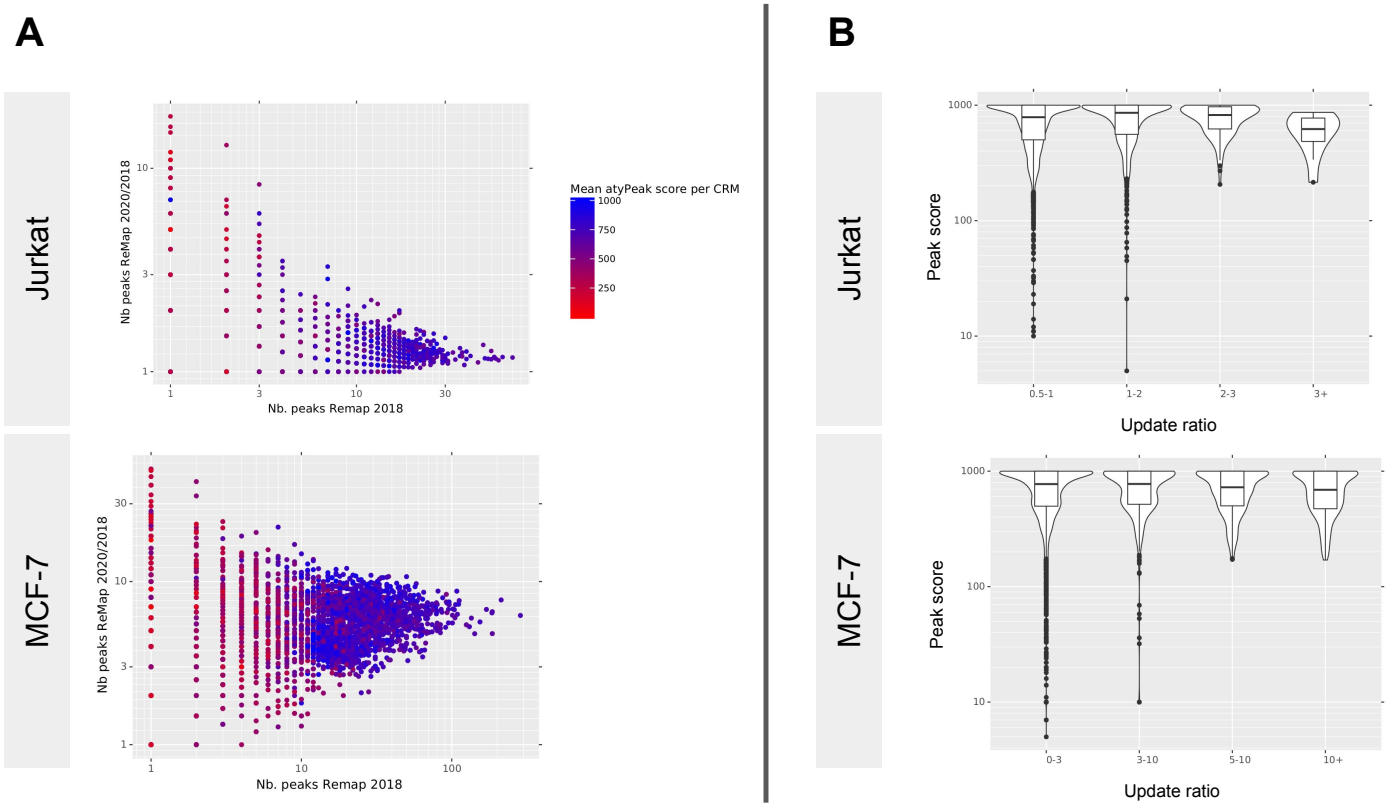


Supplementary Figure 12 : Jaccard index $A \cap B / A \cup B$ for HeLa Transcriptional Regulators. Based on the CRM representations we created for this data. Those are provided for comparison purpose and interpretation of the extracted correlations learned by the model in HeLa.



Suppl Fig 13 : Mean score per TR in HeLa before and after applying the normalization detailed in “Normalization of correlation group biases” in Methods, but before centering and reducing it based on the mean score for each TR. In summary, this normalization corrects biases due to average completeness differences between groups.

After applying this part of our normalization, the means tend to be closer between the different TRs, correcting the various biases we detailed. For example, sources learned as part of larger groups like BRD4 (see groups in Suppl Fig 11) get a needed boost to their score.



Suppl Fig 14 : Comparison between ReMap 2018, ReMap 2020 and predictions made by atyPeak.

(A) For each CRM (random subselection of 10,000), number of peaks in either update and mean atyPeak score in 2018. Low scoring CRMs tend to have less peaks, and have proportionally more peaks added in 2020.

(B) For each peak (random subselection of 5,000), number of peaks in 2020 for the same TF in the same CRM divided by the number in 2018, potentially from any number of databases. Restricted to peaks in CRM with average score of at least 500 to prevent bias described in subfigure A. Log scale is used for the score to emphasize the low-scoring peaks. We see that peaks with a score of under 250 are more rarely confirmed in 2020.

4. Statistical enrichment and combination selection with *OLOGRAM-MODL*

Sommaire

4.1. Impetus

In the previous chapter, I presented *atyPeak*, which leverages combinations of Transcriptional Regulators and datasets to perform an anomaly detection task. Here, we seek to solve a more fundamental problem: *considering a combination* [1] *of genomic*

1. A precise definition of *combination* in this context is given in section 1.4.1, p. 52.

intervals, does it occur more frequently than would be expected by chance? These genomic position intervals will often correspond to estimated Transcriptional Regulator Binding Sites, but may also represent chromatin accessibility, genic on non-genic elements, etc.

This is treated as a fundamental problem in the sense that we do not, at this stage, make any assumptions as to why a combination is found enriched, or what the consequences of such an enrichment might be. Our only concern is rigorously assessing the statistical significance of an enrichment. Indeed, insofar as genomic regulators [2] are concerned co-localisation is usually a sign of functional association, but the precise interpretation will depend on the context, especially if the genomic intervals considered do not represent regulatory proteins binding sites, but something else entirely (see section 4.4.5 below, p. 172).

There is a need of a rigorous statistical framework to answer that question. However, we have found the current approaches to be lacking, for reasons that were broached in section 1.4.5 (p. 64) and expanded upon below, as well as in the attached papers. In this section, we present the **OLOGRAM** and **OLOGRAM-MODL** projects, which sought to address those questions.

4.2. The *pygtftk* toolset

As we discussed in section 1.3.4.4, there are several conventional file formats to represent genomic regions, chief among them BED and GTF. Most of the existing tools dedicated to manipulating GTF files will, at their core, convert them into another format, usually exclusive to them, before further processing. For instance, the R/Bioconductor library *rtracklayer* (Lawrence, Gentleman, and Carey 2009) converts them into a GRanges object, while *gffutils* uses a SQLite database. This is tailored to their specific applications, but is computationally intensive and lacks flexibility.

By contrast, *pygtftk* (see the attached paper) is a toolset designed to be used as a CLI (Command Line Interface) to manipulate genomic annotations, centered around GTF files. It offers efficient annotation and manipulation of the GTF files themselves, and is capable of layering complex commands through command line piping to effect complex requests. For example, splitting a GTF file into several BED files according to the type of genomic feature, and passing to OLOGRAM for enrichment analysis. More examples are presented in the documentation of the tool.

To summarize, the *pygtftk* toolset offers a uniformized workflow to process genomic regions in the GTF format, and eventually the BED format. As a result, it was a natural fit as a platform to implement a tool that would analyze combinations of those regions, and as a result would need sanitized input in the conventional file formats for genomic regions. The tool is available on *bioconda* and merely requires the user to run the command `conda install -c bioconda pygtftk` in a Conda environment to install it.

2. And more generally, genomic elements.

Development practices On a more personal note, this project was an excellent opportunity to practice some of the good practices in software development outlined in section 1.3.4. I worked in collaboration with Denis Puthier, Guillaume Charbonnier, Nori Sadouni and Fabrice Lopez on implementing my approach on this toolkit.

> Res, non verba
> *Actions speak louder than words*
>
> *Titus Livius*

GitHub versioning was instrumental, because both the approach and toolset were gradually improved over the course of my thesis. For instance, I began implementing the approach to multiple overlaps (OLOGRAM-MODL) in code while the standard 2-wise approach was still in the reviewers' hands. Additionally, other improvements and extensions to *pygtftk* itself were developed by my collaborators during that time. This necessitated a robust system of branching to ensure only stable features were put in the hands of the users.

Relatedly, continuous integration (using Travis) and functional testing were also important. By designing small scale testing scenarios where the expected output of the approach is known with precision, we can ensure that no further update to the tools or the toolkit breaks something that was previously working. And when an error is found, this allowed me to quickly find which parts of the code could have been its source, by exculpating the ones where the functional tests signaled there was no error.

Finally, on a more human level, I discovered just how important it was to talk to your collaborators, and have robust pipelines to share the workload and specify who exactly is going to work on what, within which delays. For the tool itself, after the initial implementation of the approach, we entered cycles of feedback. These consisted of: me releasing a new version, collecting the experiences, feedback, suggestions and bug reports of my collaborators, releasing new versions, and so on so forth until we reached an acceptable state. In particular, this helped me fight "tunnel vision" when it comes to user experience, and showed me where my explanations on the inner working of my approach were severely lacking [3].

4.3. Determining the statistical enrichment of combinations using *OLOGRAM*

This study started focused on the problem of 2-wise combinations of regulators (ie. overlap of regulator A with regulator B). For a given combination γ of regulators, consider the following null hypothesis [4]:

3. Hopefully, they are now only *moderately* lacking.
4. Which is also valid for n-wise combinations, when $\text{card}(\gamma) > 2$.

Definition 6. *The null hypothesis* (H_0) *for a combination* γ *is that it is observed in the real data no more than by chance, if the regions of its constituent sets were placed at random on the genome.*

On the problem of statistical significance between two sets of genomic regions, several approaches have been proposed (Simovski, Kanduri, Gundersen, et al. 2018). Their main difference lies in the statistical model used to reject the null hypothesis. More details are provided in the introductions of the attached papers, but I would highlight a few key differences of the OLOGRAM approach here, compared to those previous approaches.

4.3.1. Statistical modeling

The underlying mathematical problem in OLOGRAM is to determine the significance of the intersections between many regions sets. However, those sets' constituent regions have varying lengths and inter-region distances. This makes an analytical determination of the expected number of overlaps highly non-trivial.

Definition 7. *For a combination* γ*, let* $S(\gamma)$ *be the number of base pairs on which it is observed.*

To reject (H_0), one must show that the observed value of $S(\gamma)$ is statistically significant and not the product of random chance [5]. The obvious solution, used by most previous algorithms, is to rely on an empirical p-value. With no assumptions made about the nature of the underlying distribution of S, it is sampled n times by performing n shuffles in which (H_0) is enforced as true. This is followed by counting the number of nucleotides for which γ is observed in those shuffles. As a result, the p-value for the observed value of S in our real data is equal to its frequency in the shuffles. However, this is *very* computer-intensive, and imprecise due to poor sampling: the p-value given can never be more lower than the n^{-1} where n is the number of shuffles, and the estimate will have a considerable standard deviation. Other approaches use a binomial test or a hypergeometric test. But in order to do so, they usually make overtly optimistic assumptions, such as reducing all regions to their center points (length of 1 base pair) which we show in the paper can skew the results towards rejecting (H_0).

Furthermore, when generating the samples (or simply modeling) according to the null hypothesis (H_0), these approaches often assume that the regions in the sets should be placed uniformly randomly on the genome. In contrast, keeping the distribution of inter-region distances, as is done in OLOGRAM, better conserved the structure of the genome; for example, regions that tend to be grouped in clusters where the clusters themselves are more distant (short repeats, etc.) will remain thusly distributed in the shuffles. However, This also makes the mathematical problem highly non-trivial and complicated an eventual analytic solution (the one precedent for this, Genomic

5. Or not explainable by the other priors, in a Bayesian setting.

HyperBrowser, used empirical p-values, and as it does not use a CLI but a graphical web interface instead is incompatible with our reproducible research paradigm).

In contrast, in the OLOGRAM paper we prove the following:

Proposition 1. *$S(\gamma)$ follows a Negative Binomial distribution when* $\text{card}(\gamma) = 2$ *and when using a shuffling conserving region and inter-region lengths.*

In OLOGRAM-MODL (see below) we extend the proof to $n \in \mathbb{N}^+$. While we have proven that S follows a Negative Binomial laws, the parameters of said law are difficult to calculate analytically, as the region sets contain regions of different lengths and inter-region distances. This difficulty will be further increased when considering overlaps of multiple independent sets in OLOGRAM-MODL.

However, we must consider this: is a full analytical solution really needed? Since we now know that S indeed follows a law, if we can estimate accurately its mean and variance, there is no need to perform trillions of shuffles to get the required precision, since rare observations can be judged in the light of the Negative Binomial law we have just fitted. Indeed, μ and σ can be estimated fairly accurately through shuffles, which will be a much less computing intensive task, with a good precision since it is much easier[6] to estimate the moments of a distribution than to estimate its tails. This is the principle behind Monte Carlo methods, as we discuss later.

Wider applicability The mathematics of the proof presented are inspired by those of the Chinese restaurant process (M. Zhou and Carin 2015). If N is the number of intersections and the length of the intersection i is L_i nucleotides, where L follows a Log-normal law, then $S = \sum_i^N L_i$ follows a Negative Binomial law. Details of the proofs are presented in the papers.

This proof has wider implications, as it is relevant for any intersections of time intervals. I would propose that any intersection of time intervals following the same properties (no overlaps of regions within the sets[7], lengths of intervals follows a logarithmic distribution, no correlation between interval sets) can be modeled using a Negative Binomial distribution.

Indeed, we found that a few hundred shuffles (see Supplementary Material of both OLOGRAM papers) are usually sufficient to accurately estimate μ and σ. This has important implications for Monte Carlo simulations of such intersections, and can be a large time saver.

4.3.2. Monte Carlo methods

Monte Carlo experiments are a broad class of computational algorithms that rely on repeated random sampling to obtain numerical results. They are an important asset

6. By which I mean it requires much fewer samples.

7. I have an intuitite belief that the proof can be extended to drop this assumption, but I have not rigorously proven it yet.

in statistical analysis (Kroese, Brereton, Taimre, et al. 2014). They aim to compute some quantity Q by finding a random variable V with can be easily sampled for which the expectation is Q. The general principle is that a good estimation of the desired quantity Q is achieved by averaging many samples of V. The approaches vary in their execution, but they usually consist of the following steps:

— Define an input generator.
— Generate inputs randomly n times from a probability distribution.
— Perform a computation on those generated inputs.
— Collect the results.

The main use case of Monte Carlo methods is when an analytical solution would be too complicated to calculate, or when an exact solution would require too much computing power. This is the case here: our goal is perform a *relatively small* number of shuffles to accurately estimate, for each combination γ, the parameters of its underlying Negative Binomial distribution. The quantity to be estimated is the moment (first then second) of the Negative Binomial distribution of $S(\gamma)$ under (H_0). The easily-sampled random variable is simply the observed moment of $S(\gamma)$ in each individual shuffle.

Relatedly, Monte Carlo methods have been extended in the form of Markov Chain Monte Carlo (MCMC) methods where the goal is to build a Markov chain that has V as its equilibrium distribution. This is used when V is hard to sample directly, as one can sample V by recording states from the chain. Examples include the Metropolis algorithm [8], or Gibbs sampling for multivariate sampling when the multivariate probability density is not known, but the conditional distributions are. Tangentially, the use of randomness is a common metaheuristic used in approaches such as simulated annealing.

Simple example Let us consider a simple illustrative example. Assume we wish to compute an approximation of the value of π. Let M be a point of coordinates (x, y) where x and y are both drawn from a uniform continuous random distribution $\mathcal{U}(0,1)$. Let D be the disk of center $(0,0)$ and radius $r = 1$. We have $M \in D \Leftrightarrow x^2 + y^2 \leq 1$ which is easy to calculate. Since the relevant disk quarter has a surface of $\frac{\pi r^2}{4}$, we have $P(M \in D) = \frac{\pi}{4}$. As such, when creating many samples M and taking the ratio of the number of points that fall inside the disk divided by the number of samples, we get a good approximation of $\frac{\pi}{4}$ when the number of samples is large.

4.3.2.1. Computing power

When the nature of the underlying probability distribution has been analytically determined, as is the case here, Monte Carlo methods offer another advantage: pre-

8. In Metropolis, the Markov chain is generated by random walk followed by accepting or rejecting the sample x with a certain probability dependant on $f(x)$, where f is a function proportional to the density of V whose values can be calculated easily. This algorithm can only be used if a relevant f is known.

cision. The most obvious downside of empirical p-values is that their precision is limited by the number of samplings. We have observed that for longer and/or rarer combinations, p-values are often lower than 10^{-10}. To put that into perspective, the best previously available empirical precision was 10^{-5} by performing 10,000 shuffles, which took several hours on the distant server used by the tool.

How many shuffles and how much time it would take to reach the precision of 10^{-320} offered by OLOGRAM is left as an exercise to the reader [9]. Of course, the usefulness of the p-value depends on the quality of the Monte Carlo approximation, but we show that only a few hundred shuffles are usually good enough to offer a robust approximation. More details are presented in the OLOGRAM paper.

4.3.3. Intersection algorithm

To compute the values of $S(\gamma)$ in all of the shuffles in an efficient manner, I used a sweep line algorithm (Shamos and Hoey 1976) instead of a more classical interval tree. See Figure 1 of the OLOGRAM-MODL paper for more details. In a sweep line, we move from critical point to critical point (the beginning or end of a genomic interval in any set) along the genome, and remember the states of all sets (did we find a beginning or an ending at the last critical point). By contrast, in interval trees all regions of a set are inserted a large tree for query later, where each node of the tree contains a center point, intervals that overlap it, and pointed to two other nodes containing all intervals completely to the left (resp. right) of the center point.

To simplify, let us consider two sets A and B containing respectively n and m sets between which we seek to query all intersections. A sweep line algorithm, with a complexity of $O(n+m)$ is more efficient when querying the intersections of an entire set of intervals against entire other sets, as a tree-based structure would have a query complexity $O(n \log m)$. Both necessitate sorting the intervals for their creation, which has a complexity of $O(n \log n)$ [10] so this does not give an advantage to either. This difficulty is compounded with more than two sets, since the complexity of the sweep line algorithm will be a sum of linear complexities, not a product of log-linear complexities like the interval trees.

However, if querying the intersection of a single element of A against the entire set B, this observation is reversed as the complexities become respectively $O(n)$ and $O(\log n)$. This illustrates that no single algorithm is the be-all-end-all in any situation. The approach we designed calls for the computation of the intersection of entire shuffled sets and once, with those shuffles being them discarded. Thus, a sweep line is more efficient.

9. Here is a hint: with current hardware, it is much greater than the estimated time until the heat death of the universe. Better get started on that solar system-sized particle collider and create a new one. Thus, I think a very reasonable and level-headed conclusion is that by proving the nature of the underlying distribution, we have saved mankind billions of trillions of years of computing power.

10. Let $B = n + m$. The complexity is $O(B \log B)$ for the sweep line (which is still log-linear), but that disadvantage will be compensated since you will need to create two trees anyways.

4.3.4. Implementation

We just discussed the scaling of algorithms. This is all well and good, but their actual implementation on a computer brought more down-to earth considerations. If an algorithm step takes 1000x as much time for the same input size, its scaling is still linear, but in practice it will not be preferred.

While Python is a very simple language and delightful to code with, it suffers from performance problems. In this project, I have written the most performance-critical parts of OLOGRAM in C: for example, shuffling is written in C, and the intersection algorithm used to compute the overlaps in each shuffle is also in C (C++ actually). The C language however brings additional complexities such as manual memory management, strong typing, and a more complicated syntax. Thankfully, the ubiquitous NumPy array managing library stores all its arrays as C arrays allowing easy passing between the two languages, but the implementation still required clever use of Cython. This also required rigorous memory management to prevent Python and C code from simultaneously accessing the same RAM positions.

The end result is, to my mind, more robust than the sum of its parts[11]. The interface, command line interpretation and file pre-processing parts are all written in Python. This code is easy to maintain and can be shared with collaborators, as most bioinformaticians are more well-versed in Python than in C. As such, OLGORAM can be extended even without knowledge of C. The additional burden of C complexity in incurred only when absolutely necessary, reducing the likelihood of bugs. In the end, since the performance-critical parts are in C, we reach computing times that are only marginally worse than pure C, with a fraction of the maintenance complexity. Tangentially, this more efficient implementation reduces the total computing cost, and thus the ecological footprint of the algorithm.

Multiprocessing by batch was added, where the number of shuffles to be performed (and the intersections to be computed on them) will be split into several minibatches, and split among the available computing cores. However OLOGRAM, and a fortiori MODL, can cost a lost of RAM (gigabytes) and CPU time (hours). This is mostly dependant on the number of shuffles to be performed, and on the sizes of the files. To sidestep this, see the perspectives presented below.

User experience To ensure that the tool is easy to use and that the results can be iterated upon, OLOGRAM returns its moment estimations, p-values and general results as a TSV (Tabulation Separated Values) file, which is an easily parsable text file. As such, it is very easy to draw custom figures based on those results. We also propose utilities to merge different runs into a comparative heatmap, as well as easily-readable enrichment graphs. This carries over to OLGORAM-MODL.

11. "More than the sum of its parts." Where have I read this before?

4.4. Higher-order combinations and itemset mining with *OLOGRAM-MODL*

The classical OLOGRAM only considers pairwise enrichments, with combinations of order 2. Indeed, many existing methods (Bedtools Fisher, LOLA, ...) consider pairwise combinations only. Anecdotally, it is telling that in the ENCODE general paper (The ENCODE Consortium 2012) they considered only pairwise correlations (see the Figure 4 of that paper).

However, Transcriptional Regulators and epigenomic regulators do not simply work in correlated pairs, then tend to work in n-wise complexes. Some work has been done on this (see Introduction of the OLGORAM-MODL and 1.4.5) but for various reasons this was not satisfactory. This has mostly to to with byzantine complexity or inadapted modeling. Since the OLOGRAM approach corrected several biases observed in overlap computing approaches, I wondered if those insights could be extended to higher order combinations of regulators.

4.4.1. Extending OLOGRAM to higher-order combinations

A key insight was the realization that for combinations γ of order $\text{card}(\gamma) \geq 2$, $S(\gamma)$ still followed a Negative Binomial distribution. Examples can be found in the Supplementary Material of the OLGORAM-MODL paper. At first, this was only an empirical observation. But it clued me in that, indeed, our proposed OLOGRAM Negative Binomial modeling would be relevant for multiple overlaps (ie. higher-order combinations).

Eventually, I extended the proof to those combinations. The details are in the paper, but the gist of it is as follows: let I be a random variable so that $I(x, y) = 1$ if and only if the regions x and y intersect. If we accept that $I(A_i, B_j)$ is a Bernoulli random variable, then $I(A_i, B_j, C_k) = I(A_i, B_j) * I(B_j, C_k) * I(A_i, C_k)$ can be approximated also by a Bernoulli R.V. of unknown p. This lands us back in the proof used in the original OLOGRAM paper, so that $S(\gamma)$ follows a Negative Binomial. However, if p is unknown an analytical solution cannot be computed for this Negative Binomial. But the Monte Carlo method we have devised relishes such a challenge, and returns accurate estimations.

Tree-based representation Another improvement was the use of a tree-based representation of the combinations (see Supplementary Figures) along with their enrichment statistics. This is not a novelty *per se*, as this structure is commonly used in itemset mining. It improves the visibility by highlighting master regulators: combinations containing them will tend to have many enriched children combinations. This also helps identify closed itemsets by showing, for each combination γ, how much of its $S(\gamma)$ is accounted for by its parents.

4.4.2. MODL itemset mining algorithm

However, with k sets of regions, the number of potential combinations to be displayed can reach 2^k. To focus the user output on the most interesting ones, we designed an itemset mining algorithm. It ties to the biological problematic in that it is designed to find complexes of regulators, and as such find the itemsets that *best describe* the data, instead of simply the most frequent. For example, the itemsets $\gamma_1 = \{A, B\}$ and $\gamma_2 = \{C, D\}$ are sufficient to describe the three transactions $(AB, CD, ABCD)$ [12].

The details of the algorithm are presented in the paper. Here, I present some additional background information and impetus. In practice, this MODL algorithm is applied on the matrix of intersections in the true data, not in the shuffles. I would also like to point out that this is *separate* from the Negative Binomial modeling we introduce. In the tool, the MODL algorithm is only used to select the combinations for which enrichment will be calculated and displayed to the user, but does not affect the calculation of the enrichment itself.

This algorithm leverages matrix factorizations to extract itemsets, and submodular optimisation to select them. We begin by presenting the principles of these approaches.

4.4.2.1. Matrix factorizations

Let us now intuitively explain how itemsets can be found in the factors of a matrix decomposition. Consider a decomposition of an input matrix X with k latent factors:

$$\mathbf{X}_{m\times n} \approx \mathbf{U}_{m\times k} \times \mathbf{V}_{k\times n}$$

Each of the k columns of U corresponds to a row in V through the matrix product. Each of those rows of V is an itemset. Here is a practical example of this form of decomposition, with each itemset of interest highlighted in a different color:

$$\begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 1 \\ 1 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \end{pmatrix} \times \begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

A matrix decomposition of X of the form $\tilde{X} = UV$ is strong (ie. exact) if $X = \tilde{X}$. Conversely it is weak if $X \approx \tilde{X}$, for which another notation is $X = \tilde{X} + \epsilon_{m\times n}$ through the addition of an error term. Weak decompositions are of interest and will be used, since the addition of an error terms help them be resistant to noise. This is the same principle of lossy compression that was leveraged in atyPeak. There are, however, other types of matrix decomposition with different constraints. I present some common decompositions here to provide some context for the reader.

12. This is a simplified version of the problematic, but it captures the gist. See the paper for more.

The QR decomposition of the form $X = QR$, where Q is an orthogonal matrix (meaning $Q^T Q = I$) and R an upper triangular matrix, is often used to solve the linear least squares problem (least squares approximation of linear functions to data). The LU decomposition on the other hand decomposes $X = LU$, where L and U are respectively lower and upper triangular matrices, and is often used to solve systems of linear equations. A more efficient alternative to LU in certain cases is the Cholesly Decomposition of $A = LL^*$ [13].

One must also mention the singular value decomposition, or SVD. It is a decomposition of the form $\mathbf{X}_{m\times n} = \mathbf{U}_{m\times m} \times \Sigma_{m\times n} \times \mathbf{V}^*{}_{n\times n}$ where $\mathbf{U}$ and $\mathbf{V}$ are both unitary matrices and Σ is a diagonal matrix. In this decomposition, the diagonal values of Σ are the singular values of $\mathbf{X}$, with the columns of $\mathbf{U}$ and $\mathbf{V}$ being the left and right singular vectors of $\mathbf{X}$. SVD is at the core of Principal Component Analysis. To find the principal components of a data matrix $X_{n\times p}$ with n samples and p variables, one must get the eigenvectors of the covariance matrix C between the p variables. [14] While there are similarities, PCA and SVD are not designed for itemset mining but for dimensionality reduction.

Finally, *custom* matrix decompositions can be written for different objectives. To give some examples, Canonical Correlation Analysis is an analog of PCA aimed at finding pairs of correlating components between two (or more) matrices, and can be though of as a PCA applied to the covariance matrix between the features of those two matrices, as opposed to the covariance of the features of X with themselves. The Partial Least Squares regression between two data matrices X and Y inherits from both PCA and regression and seeks to decompose $X = TP^T + E$ and $Y = UQ^T + F$ into a product of matrices where the columns are components of interest so that the covariance between U and T is maximal. In the latter two cases, the optimisation objective is different so that the components obtained can give a different insight: in this case, covariance between variables in two matrices instead of merely rebuilding a single matrix.

All the decompositions presented in this part are often performed using Gaussian elimination, or various iterative algorithms designed to improve gradually and converge on the solution, often producing one component/vector of the factorization at each iteration (Gram-Schmidt, PLS, ...). The QR decomposition is usually performed using the Gram-Schmidt algorithm, by subtracting from each column vectors its own projection onto the subspace defined by the previous column vectors. Computing the SVD of a matrix X is a hard problem in general, but is usually done by first reducing X to a bidiagonal matrix B using Householder reflections, and then using the QR algorithm on B to compute it (perform a QR decomposition, multiply the factors

13. L^* is the conjugate transpose of L. The Cholesky Decomposition is only applicable when A is a Hermitian symmetric positive-definitive matrix.

14. Since $C = \frac{X^T X}{n-1}$, by replacing X with its decomposition $X = U\Sigma V^*$ in the previous equation, one gets $C = V\frac{S^2}{n-1}V^*$ meaning that columns of V are the principal directions/axes and columns of US are principal components ("scores"). Alternatively, one may instead perform an SVD decomposition of C as $C = U_C \Sigma_C V_C^*$ in which case the principal components of X are the columns of U_C.

in the reverse order, and iterate). More generally, most matrix factorizations can be reasonably well approximated using gradient descent (Ho, Van Dooren, and Blondel 2011) and some use it as a first resort, such as the Dictionary Learning presented below.

Dictionary learning In OLOGRAM-MODL, the input matrix $\mathbf{X}$ that will be factorized has one row per overlap and one column per set in the real, non-shuffled data. The specific matrix factorization used in OLOGRAM-MODL to extract itemsets, as presented in the original exemple of this part, is called dictionary learning (Mairal, Bach, Ponce, et al. 2009). This is a matrix factorization problem with sparsity that entails solving:

$$(\mathbf{U}^*, \mathbf{V}^*) = \arg\min_{\mathbf{U},\mathbf{V}} \frac{1}{2}\|\mathbf{X} - \mathbf{U}\mathbf{V}\|_2^2 + \alpha\|\mathbf{U}\|_1$$
$$\text{subject to } \|\mathbf{V}_i\|_2 = 1 \text{ for all } 0 \leq i \leq n_{\text{atoms}}$$

Note that in the previous equation only, $\mathbf{U}^*$ designates an *optimization objective* and not a conjugate transpose. A dictionary V is composed of atoms (rows of V), which are used to rebuild richer words (rows of X, combinations). The sparsity constraint α will reduce the number of words that are allowed to be used to rebuild each combination, which will in turn result in longer words. Dictionary Learning is often used in image denoising.

4.4.2.2. Submodularity and greedy algorithms

Submodular set functions [15] are functions who have the property that the incremental difference in their value upon adding new elements to the input set decreases when the size of the input set increases. Formally, a set function f is submodular if:

$$\forall X, Y \subseteq \Omega \text{ with } X \subseteq Y \text{ and every } x \in \Omega \setminus Y, \text{ we have}$$
$$f(X \cup \{x\}) - f(X) \geq f(Y \cup \{x\}) - f(Y)$$

Submodular set functions can be seen as the set function analogue of convex functions [16], and as such they share many of the same intuitions. Many problems in function approximation, game theory, and machine learning can be written as submodular functions, and use their heuristics. For instance, feature selection [17] is often submodular, and in any case it is indeed submodular for Naive Bayes classifiers (K.

15. A set function is simply a function whose input is a set.

16. Intuitively, a real-valued function is called convex if the line segment between any two points on the graph of the function lies above the graph between the two points.

17. Here, broadly defined the selection of a subset of relevant features (variables, predictors) that best approximate or are most relevant for a given result

Wei, Iyer, and Bilmes 2015). Tangentially, this will mean that the insights of the MODL algorithm that we will present can be applied to other submodular problems.

Submodular maximisation is an NP-hard problem, but greedy algorithms will provide a solution with a factor of $1 - \frac{1}{e}$, which is the best possible approximation (Feige 1998), as is explained in the OLOGRAM-MODL paper. A greedy algorithm is defined as an algorithm that makes the locally optimal choice at each iteration.

Here is an intuitive explanation: consider a hiker climbing a mountain. If the weather is clear, the summit is visible and the hiker can clearly see if they are indeed moving towards it. Now, assume a deep fog has set, so that the hiker can only see twenty meters in front of them. How to reach the summit? If our hiker uses a greedy strategy, they will simply look around and move to the most elevated position that they can see, and repeat the process until they reach a locally maximal... sorry, a local summit. With this approach, will they reach the *global* summit eventually? Yes, *provided* the mountain is concave [18] Since submodular set functions are analogous to convex functions, one can intuit how this greedy heuristic can be translated to them.

Another example of problem well-approximated by greedy algorithm is the set covering problem. Given a universe of elements $\{1, 2, ..., n\}$ and a collection S of m sets containing elements from this universe, the set cover problem is to identify the smallest (in terms of number of members) sub-selection C of S so that the union of C equals the universe. Here is an intuitive analog: how can one place 5G relays so that all of the population is covered, using the smallest possible number of relays? [19] While the set covering problem is NP-hard, a good approximation can be obtained by a greedy algorithm.

4.4.2.3. Combining factorizations and itemset mining

The essence of the MODL algorithm is this: we prepare a list of candidate itemsets (aka. words), down from all possible 2^k. The candidates are found by performing Dictionary Learning of the matrix X of true interesctions, with one line per overlap and one column per set of regions. The factorizations are performed with steadily increasing sparsity parameters [20] to get progressively longer itemsets, and repeated with different random seeds to ensure a good diversity.

Once this preselection is done, we have a set Λ of candidates itemsets. Based on a proof by Krause and Cevher 2010 we show that getting the best possible dictionary V by selecting among those candidates is a submodular problem. Hence we can solve [21] it using a greedy algorithm: starting with an empty dictionary, we iteratively add to V

18. If f is concave on a given interval, $-f$ is convex on that same interval. Minimizing a convex function and maximizing a concave one are equivalent problems.

19. Each element of the universe is one member of the population, each set of S is a set containing the inhabitants that would be covered by placing a tower at a given location.

20. In this entire chapter, α designates a sparsity controlling parameter; this follows the conventional notation. However, this α should not be confused with the learning rate in gradient descent, that is also usually noted α.

21. Or more rigorously, find the best approximation of it.

the words that best improve the reconstruction of I. This is the basic principle. Other technical considerations such as the choice of α for this reconstruction or the use of a L1 loss to discourage compromises words that contain more itemsets with a lower value are presented in the paper.

I believe that an interesting insight is to compare our approach with some other itemset mining algorithms, as presented in section 1.4.4.1 (p. 62). Indeed, the KRIMP algorithm similarly selects itemsets based on how well each itemset helps rebuild the larger set of all itemsets, much like we select itemsets that best rebuild the original data. Furthermore, closed itemset mining is vulnerable to noise, which is a concern here, and we use a weak matrix decomposition to perform approximate itemset mining to counter that problem; doing so is seen as a promising avenue in the presence of noise.

4.4.3. Conclusion and biological interest

Complementarity of the statistical and ML approaches In conclusion, the OLOGRAM-MODL approach shows an example of mutual cooperation between our statistical Negative Binomial model and the Machine Learning algorithm used to select itemsets. These two parts are run independently, but complement each other. The itemset mining algorithm, in turn, is only used to restrict the number of combinations to be evaluated and, more importantly, displayed. In any case, the enrichment of any combination of regions γ is quantified through the Negative Binomial model we propose. This has the considerable advantage of being very straightforward. For the average bioinformatician and biologist, the p-value returned by the model has an immediate significance, as opposed to more arcane coefficients in a linear regression or other selection approach, which can only be appreciated when compared against each other.

Biological interest This approach allows one to find relevant enriched combinations of regulators that might previously have been neglected. In the OLOGRAM-MODL paper, the illustrative biological example we used is that of FOXA1 in the MCF7 breast cancer cell line. As is standard, most of the literature on FOXA1 focused on pairwise interactions with other Transcriptional Regulators. For instance, it is known to act as a pioneer factor to the regulator ERα (aka. ESR1, Ross-Innes, Stark, Teschendorff, et al. 2012), and is known to be downstream target of the regulator GATA3 in breast cells (Kouros-Mehr, Slorach, Sternlicht, et al. 2006).

We show that FOXA1 is instead part of a more complex regulatory network. Instead of speaking of FOXA1 only as a pioneer to ESR1, it would be more correct to say it is part of a regulatory complex to which FOXA1 also belongs, which can include EP300 in certain cases. We also show that FOXA1 is associated to active enhancers, a point that is to my knowledge seldom acknowledged in the literature. I hope that this example can demonstrate that considering n-wise combinations can help highlight *regulatory complexes* and perhaps functional associations with other genomic elements, beyond a simple pairwise analysis.

OLOGRAM and OLGORAM-MODL were also used as part of a collaboration with Nori Sadouni to determine the enrichment of combinations of Transcription Factor Binding Sites in candidate silencers sites in mice. In this project, ChIP-Seq data was unavailable, so we had to find another ersatz to estimate the Transcription Factor Binding Sites. We settled on the use of JASPAR motifs, using only those with a score of at least 500 to reduce the amount of false positives, and expanding them to cover the entire candidate silencer in which they were present [22].

4.4.4. Limitations

We discuss several limitations to this approach in the OLOGRAM and OLOGRAM-MODL paper.

General limits The most apparent limitation of the OLOGRAM approach concerns the higher-order combinations. The longer a combination is (meaning, the more constituent sets it has), the higher its enrichment will usually be regardless. This is not an error: under (H_0), it is perfectly logical that it would be less frequent for five independent sets of regions to be open at the same position than it would be for only two. However, this makes comparison between combinations of different orders more complicated. As such, I would recommend mostly focusing on combinations of shorter and/or of the same order.

Furthermore, for very rare and/or very high order combinations, only a few hundred shuffles may be insufficient to encounter them on the genome. Indeed, let us consider three sets covering each 1% of the genome. A quick back-of-the-envelope approximation shows that the likelihood that those tree sets will all cover a given nucleotide is $(10^{-2})^3 = 10^{-6}$, one in a million odds. Which mean we would expect to encounter this combination only for around one shuffle when performing a million of them, while the usual procedure calls only fore hundreds of shuffles. Hence, I recommend restricting the shuffling only to a sub-section of the genome that is of interest. For example, as is done in the papers, shuffling TFBS not across the entire genome but only across open chromatin sites, as estimated through DNAse I Hypersensitivity Sites. This increases the relative coverage of the sets and results in higher odds, that are also more biologically relevant.

When not using the MODL itemset mining algorithm, there are up to 2^k possible combinations, where k is the number of sets. In practice however, the true maximal number of combinations displayed is simply the number that were indeed encountered in the data, which is much lower than 2^k but can still easily amount to thousands. Some form of selection of the combinations, be it with MODL or with a custom selection, is mandated. This is made easier by returning a TSV file which can be easily parsed.

22. Otherwise, there would we no overlap between the TFBS, as two proteins cannot physically occupy the same space. This *slopping* imitates the wider peaks found in ChIP-seq data and allows us to consider close TFs as overlapping.

Also, processing very large files and/or too many of them has a severe computing cost. In our tests presented in the papers, on a laptop, the time scale is of the order of minutes to hours to process a dozen files (sets) containing several tens of thousands of regions. Storing all the shuffles in memory can consume several gigabytes of RAM during the processing (this can be alleviated by some parallelization). This number of files and regions constitutes a very usual scale for genomic assays. As such, the time cost remains in a range that I believe is reasonable for most use cases it is likely to encounter. This is of course faster when using a supercomputer, as was the case for some applications, but developing a tool that could still reasonably be run on a common computer was an important goal of mine. However, dealing with truly enormous files containing millions of regions, or using dozens of sets, can drive the time and memory cost upwards.

MODL itself The MODL itemset mining algorithm has several further limitations. The key one lies in the choice of the number of words to be returned. Indeed, the number of words queried during the dictionary learning steps depends on it. Using a too large number of words will not be informative, while using a too low number of words may result in words representing *potential correlation groups*, meaning the matrix factorization may learn the word $\begin{pmatrix}1 & 1 & 1\end{pmatrix}$ to represent both $\begin{pmatrix}1 & 1 & 1\end{pmatrix}$ and $\begin{pmatrix}1 & 1 & 1\end{pmatrix}$. This is partially alleviated by the subsequent selection performed in step 2 by the greedy algorithm, which considers the best candidates among those available and can tolerate a handful or irrelevant words in the candidates.

Some other choices of parameters, such as the sparsity controlling parameter to be used in Dictionary Learning ($\alpha_D L$) and the one used during the second step of submodular selection for the candidate reconstructions (α_R) can also impact the found words. Indeed, I had observed that a poor choice of α (either too low or too high) could result in convergence errors, making the results unusable, or in selected words that were not closed itemsets and focused on improving the rebuilding of the most frequent combinations rather than consider the other combinations.

Relatedly, to prevent a focus on the most frequent combinations in general, it was necessary to implement a procedure I called *smothering*. The MODL algorithm works on a smothered version of the input matrix of intersections X, where:

Definition 8. *Let the abundance of a row x in a matrix X be the number of rows in X which are exactly equal to it, noted $a_X(x)$. Then the smothered version of the matrix X is the matrix $\psi(X)$. For each unique row x of X, $a_{\psi(X)}(x) = \sqrt{\frac{a_X(x)}{\nu}}$, where ν is the highest of either* $\min(a_X)$ *or the abundance threshold τ. Row order is unimportant.*

As a result, the most frequent combinations are de-emphasized during the itemset mining search.

To conclude, I view the MODL algorithm as a separate part of the approach, needing more development and more rigorous evaluation of the impact of its different parameters. Although it is, of course, fully functional at present. This "work-in-progress"

status made it even more important to ensure that OLOGRAM could be run on higher-order combinations even without MODL, and that MODL was independent from the statistical framework itself.

4.4.5. Perspectives

In this section, I would like to discuss some perspectives and potential applications for the OLOGRAM-MODL approach. Several of them were planned for the papers' submissions but had to be cut due to the depressing fact that there are only seven days in a week. However, the implementation of many of these applications is either very straightforward or already prepared in the code.

4.4.5.1. Applicability to closely related problems

The perspectives presented in this subsection are merely a matter of applicability: OLOGRAM-MODL already fully supports them.

Distance between the features Currently, the OLOGRAM approach only registers overlaps between intervals in different sets. We do not consider closeness, meaning whether elements are found closer to each other than would be expected by chance. However, a workaround can be applied by *slopping* the regions in the intervals; this means extending the intervals in 5' and 3' by a given length. For example, if there is a significant enrichment found when extending the regions in the sets A and B by 2000 base pairs, it would suggest that the genomic features described by the sets A and B indeed tend to be closer than 2kbp, even if they do not overlap. Relatedly, the statistical enrichments could be compared for different slop values to find the optimal distance.

Regions of a different nature and multi-omics The examples presented in the papers concern only Transcriptional Regulator Binding Sites, as estimated through ChIP-Seq. However, as has been emphasized *ad nauseam* in section 1.4.1, this approach can be used with any data that can be represented as a set of intervals. Indeed, this is true for the genomic assays we presented in section 1.2 and many more that were not presented, such as DNA methylation status assays.

I believe it would be very appropriate to use OLOGRAM-MODL to assess the functional enrichment between regions of a different nature, perhaps even as part of of a multi-omics approach by assessing the statistical significance of overlaps between, say, a histone mark of interest with Transcriptional Regulators of interest.

Another possiblity that I find most intriguing would be to use certain genomic regions as proxies for objects of interest. For instance, the set of promoters for the overexpressed genes in cancer patients for a given condition could be a set, and OLOGRAM-MODL would quantify the enrichment between those promoters and various epigenomic marks of interest. To refine the analysis, the enrichment for each

combination γ could even be compared to the enrichment of γ observed for all genes. This would help determine the regulatory processes involved in the condition.

Contacts between genomic elements To integrate contacts between genomic regions and study the structure of the genome, it is possible to use a set the list of genomic intervals with which a region of interest is in contact. For example, let us assume an analysis with 2000 genomic regions of interest. Now, we consider a BED file containing the positions with which the genomic region n°1429 was found in contact with the genome using a HiC experiment.

It is possible to use OLOGRAM-MODL to evaluate the statistical enrichment of the contacts of region n°1429 with other types of elements in the genome, and even combinations of contacts for several regions [23].

Extension of MODL to other problems The MODL itemset selection algorithm can be applied to any submodular problem of which, as previously discussed, there are many that are interesting. One such perspective that I find promising would be to use it to select itemsets that are the best predictors of a condition of interest, as opposed to merely the ones that best rebuild the original data. Several algorithms for such a supervised selection have been proposed (see introduction of OLOGRAM-MODL) and I believe my approach could help, assuming the underlying problem can be framed as a submodular selection.

In order to allow such uses, I made the MODL algorithm accessible through a Python API separately from the OLOGRAM-MODL tool itself, and custom error functions can be specified even as of today, making this a possible near term application.

Parallelization This is more of an implementation problem. We have implemented demonstrations as Snakemake pipelines and data parallelization is very easy for different files. OLOGRAM is parallelized by threads so you can run one minibatch by thread. I recently developed a plugin that permits some further parallelization, by merging different runs so that each run works as a batch of batches.

4.4.5.2. Extensions of the approach

The perspectives presented here would require minor adjustements, some of which have been partially prepared.

Lebesgue integration of signal values This conversion would allow OLOGRAM-MODL to work on timeseries as a signal with values in $\mathbb{R}^+$, as opposed to the simple binary signal of "present" or "absent".

23. Although technically possible, this should not be used for all 2000 regions at once, but for a more reasonable amount.

To do so, we need to allow intervals to overlap within a set. This has been partially prepared in the code already, this imply entails keeping negative inter-region distances. For example, the distance between the intervals $[100;200]_{chr1}$ and $[150;250]_{chr1}$ is the distance between the end of the beginning of the second one and the end of the first one $150-200=-50$. Then we can approximate the signal through vertical signal binning, in a process analogous to a Lebesgue integration. For example, if binning with a resolution of 20, a signal of 100 would be represented by 5 stacked intervals.

Having several intervals open at any given time for the set would be represented as a value higher than 1 in the intersection matrix X. The sweep line algorithm is already designed to count the number of currently open regions at any given critical point, and not simply to register either "open" or "not open".

This perspective is one of the reasons why I used Dictionary Learning[24], which works using real-values matrices, as opposed to binary matrix factorizations.

Time based A more distant perspective would be to integrate a time-based component in the approach by concatenating several lines into a single line. For example, consider three sets A, B and C. If the combination $\{A,B\}$ occurs at the position $t-1$ and the combination $\{B,C\}$ occurs at the position t, the representative vector at the position t would be $x_t = \begin{pmatrix}1 & 1 & 0 & 0 & 1 & 1\end{pmatrix}$, with the first three columns giving the status of the sets at $t-1$ and the next three at t. This could be extended to $t-2$, etc.

I think this would be interesting as it would hearken back to the original roots of Dictionary Learning as an image denoising tool, which worked by learning the usual successions of pixels in an image over short distances on the X axis (say, 5 pixels).

Non-independent sets OLOGRAM-MODL's frameworks also supports the future implementation of custom shuffles, where a shuffling random seed could be shared between sets. This, in turn, could be interpreted by the shuffling function to perform shuffles with additional weight to certain regions, and would have the end result of representing a correlation between those two sets.

This could also be used to pass any message to the shuffling function to modify the shuffle being performed. Integrating such correlations between sets would make an analytical solution even more elusive and justify the need for a Monte Carlo approach even further. However, it would likely require an adaptation of our mathematical proof. Since we already assume sums of *dependent* Bernouilli variables $I_{a,b}$ to represent intersections between the regions a and b as the regions do not overlap within the sets, I believe it would be possible to integrate an additional dependence while retaining a Negative Binomial distribution, but this needs to be rigorously proven.

24. If interpreting the words found by Dictionary Learning, one must be careful as they would only show the proportions between features, not absolute values. This is easily fixable by adding a control column in X with a value of always 1, and normalize the words so they have a value of 1 in that column.

4.5. Modelisation of Cap-STARR-Seq data

The use of a Negative Binomial model was also extended to other problematics. In the Appendices (section A, p. 238), we present a more work-in-progress project, for silencer Cap-STARR-Seq data. This section is mostly based on work by Dominic Van Essen and Nori Sadouni. I present there some insights that I contributed to the modeling of Cap-STARR-Seq data. This is a very annex part of my thesis project, and readers uninterested in the particulars of Cap-STARR-Seq should feel free to skip it.

4.6. Articles

— F. Lopez, G. Charbonnier, Y. Kermezli, et al. “Explore, edit and leverage genomic annotations using Python GTF toolkit”. In: *Bioinformatics* (Mar. 12, 2019). DOI: 10.1093/bioinformatics/btz116. URL: https://academic.oup.com/bioinformatics/advance-article/doi/10.1093/bioinformatics/btz116/5320559

— Q. Ferré, G. Charbonnier, N. Sadouni, et al. “OLOGRAM: determining significance of total overlap length between genomic regions sets”. In: *Bioinformatics* 36.6 (Mar. 1, 2020), pp. 1920–1922. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/btz810. URL: https://academic.oup.com/bioinformatics/article/36/6/1920/5613178 (visited on 08/27/2020)

In the text, the formulation "the OLOGRAM MODL paper" refers to the included paper "Monte Carlo based mining of enriched n-wise combinations of genomic features with dictionary learning". This paper is, as of writing, under review at the *Bioinformatics* journal.

Bioinformatics
doi.10.1093/bioinformatics/xxxxxx
Advance Access Publication Date: Day Month Year
Application Note

Genome analysis

Explore, edit and leverage genomic annotations using Python GTF toolkit

F. Lopez [1,2], G. Charbonnier [1], Y. Kermezli [1,3], M. Belhocine [4], Q. Ferré [1], N. Zweig [5], M. Aribi [3], A. Gonzalez [1], S. Spicuglia [1,6], D. Puthier [1,*]

[1] Aix Marseille Univ, INSERM, UMR U1090, TAGC, Marseille, France , [2] Aix Marseille Univ, INSERM, TAGC UMR U1090, BCF-C platform, Marseille, France, [3] Tlemcen University, The Laboratory of Applied Molecular Biology and Immunology, Algeria, [4] Molecular Biology and Genetics Laboratory, Dubai, United Arab Emirates, [5] Aix Marseille Univ, [6] Equipe Labellisée LIGUE contre le Cancer.

*To whom correspondence should be addressed.

Associate Editor: XXXXXXX

Received on XXXXX; revised on XXXXX; accepted on XXXXX

Abstract

Motivation: While Python has become very popular in bioinformatics, a limited number of libraries exist for fast manipulation of gene coordinates in Ensembl GTF format.
Results: We have developed the GTF toolkit Python package (pygtftk), which aims at providing easy and powerful manipulation of gene coordinates in GTF format. For optimal performances, the core engine of pygtftk is a C dynamic library (libgtftk) while the Python API provides usability and readability for developing scripts. Based on this Python package, we have developed the gtftk command line interface that contains 57 sub-commands (v0.9.10) to ease handling of GTF files. These commands may be used to (*i*) perform basic tasks (*e.g.* selections, insertions, updates or deletions of features/keys), (*ii*) select genes/transcripts based on various criteria (*e.g.* size, exon number, TSS location, intron length, GO terms) or (*iii*) carry out more advanced operations such as coverage analyses of genomic features using bigWig files to create faceted read-coverage diagrams. In conclusion, the pygtftk package greatly simplifies the annotation of GTF files with external information while providing advance tools to perform gene analyses.
Availability: pygtftk and gtftk have been tested on Linux and MacOSX and are available from https://github.com/dputhier/pygtftk under the MIT license. The libgtftk dynamic library written in C is available from https://github.com/dputhier/libgtftk
Contact: denis.puthier@univ-amu.fr

1 Introduction

Several formats exist to store genomic features. The standard BED format stores basic information (chromosome, start, end, name, score and strand) related to generic genomic features (BED6) or composite genomic features (BED12). The GTF/GFF2 format (thereafter referred as GTF) can describe more exhaustively defined genomic features (genes, transcripts, exons...) by taking advantage of the 'attributes' column which contains a set of keys/values to store various kinds of annotations. Some composition relationships are implicitly declared in the GTF file making it possible to describe, for instance, the exons of the transcripts corresponding to a gene. This relationship is more explicit in the GFF3 format that can be viewed as a directed acyclic graph with nodes corresponding to features (gene, transcript, exon...) and edges corresponding to part-of relationships. Only few libraries are specifically dedicated to GTFs and most of them propose very focused tasks. The GenomeTools suite is a collection of bioinformatic tools based on the libgenometools C library that handle GTF and GFF3 formats (Gremme *et al.*, 2013). However, this library extends well beyond these annotation formats and the developing framework may appear rather complicated for naive developers as it requires deep knowledge of C programming language. Regarding R/Bioconductor, the rtracklayer provides fast access to the GTF/GFF by providing the user with a GRanges object (Lawrence *et al.*, 2009).

While the Python language has gained lot of popularity among bioinformaticians, only a handful of tools are available for manipulating GTF files. The gffutils package can parse and store GTF/GFF files into SQLite databases. The creation of a subsequent hierarchical models of genomic features while highly useful can be relatively time consuming. We developed the pygtftk package with the objective to provide a fast and readable way to load and manipulate GTF files within Python scripts. This package comes with the gtftk command line interface (CLI) that provide various operations to write workflows based on GTF files.


```
from pygtftk.gtf_interface import  GTF
from pygtftk.utils import get_example_file

# Create a GTF instance
cod_pot = get_example_file('mini_real', 'tab')[0]
gn_info = get_example_file('mini_real', 'genome')[0]
gtf = GTF(get_example_file('mini_real', 'gtf.gz'))

# Get a BedTool object containing the TSSs
# of the selected transcripts
tss = gtf.select_by_key('gene_biotype', 'lincRNA'
).select_by_transcript_size(min=200
).select_by_number_of_exons(min=2
).add_attr_from_file(feat='transcript',key='transcript_id',
                             has_header=True, new_key='coding_pot',
                             inputfile=cod_pot
).eval_numeric('coding_pot < 0.2', na_omit='.,?'
).get_tss(name=['transcript_id', 'gene_name', 'gene_id']
).slop(s=True, l=1000,r=1000, g=gn_info)
```

Fig. 1. Use case for the pygtftk package. These few lines of codes are used to extract the promoter region ([-1000, 1000] around the TSS) of LincRNAs, with the conditions that the transcripts have size greater than 200nt, at least two exons and a coding potential (assessed by CPAT and joined from an external file) below 0.2. (Wang et al., 2013)

2 Implementation

2.1 The core libgtftk C library

The core of the package is written in C and exposed through a dynamic library called libgtftk. The GTF format is represented without hierarchical relationships to maximize performances. More complex operations are carried out by the libgtftk Python client.

2.2 The pygtftk Python package

The GTF class of pygtftk comes with a large number of methods. Most of these methods return a new GTF object so that they can be chained intuitively. This object can also produce two additional objects from the gtftk library including: a TAB object (representation of a matrix) and a FASTA object (representation of a FASTA file). The GTF object is integrated within the scientific Python ecosystem and can produce *pybedtools.BedTool* objects, *Bio.SeqRecord* generators or a *pandas.DataFrame* (Quinlan, 2014; Cock *et al.*, 2009; McKinney, 2010). A typical use case is proposed in Figure 1 where the transcription start site (TSS) coordinates of lincRNAs are extracted with the conditions that *(i)* the transcript size is above 200nt, *(ii)* the number of exons is greater than 2 *(iii)* and the coding potential (imported from a separated file) is lower than 0.2. The TSSs are then obtained using the *get_tss()* method returning a *pybedtools.BedTool* object that can be used to extend coordinates by 1000 nucleotides in the 5' and 3' directions. Regarding performances, the human genome annotation in GTF format from Ensembl release 92 ($\sim 2.7.10^6$ lines) is loaded in about 30 seconds while the creation of a hierarchical model using gffutils takes about 11 minutes (performed on Intel(R) Xeon(R) CPU E5-2640 v3, 2.60GHz). In addition, the search engine is also highly optimized since it takes 0.6 seconds to select all lincRNAs from the human genome.

2.3 The gtftk command-line interface

The pygtftk package provides a gtftk CLI with 57 subcommands. These subcommands can be used to: *(i)* download GTF files, *(ii)* edit them, *(iii)* mine the GTF files in various ways (*e.g.* select transcripts by genomic/exonic/intronic size, number of exons, associated GO term...), *(iv)* annotate the GTF files (e.g. flagging divergent/convergent/overlapping transcripts...), *(v)* convert them to other formats or *(vi)* perform epigenomic analyses by producing faceted coverage diagrams through the plotnine Python package (*i.e.* the recently developed Python port of ggplot2).

3 Conclusion

The pygtftk package and the associated gtftk CLI provides a new way to easily handle gene coordinates with Python. They are regularly updated and users familiar with Python and/or command-line programs should quickly get comfortable and productive with (py)gtftk. As the GTF/GFF format is now also used for storing regulatory features and variants, this paves the way for future developments of (py)gtftk that could be an interesting framework for the integration of heterogeneous genomic data (Zerbino *et al.*, 2018; Reese *et al.*, 2010).

Acknowledgements

We thank Jacques van Helden for helpful discussion.

Funding

G.C. was supported by a fellowship from the "Fondation pour la Recherche Médicale" (FRM). S.S. and D.P. were supported by recurrent funding from INSERM and Aix Marseille Univ and by the Foundation for Cancer Research ARC (ARC PJA 20151203149) and A*MIDEX (ANR-11-IDEX-0001-02), Plan Cancer 2015 (C15076AS) and Ligue contre le Cancer Equipe Labellisée. Y.K., was supported, by the Franco-Algerian partenariat Hubert Curien (PHC) Tassili (15MDU935).

References

Cock, P. J. *et al.* (2009). Biopython: freely available Python tools for computational molecular biology and bioinformatics. *Bioinformatics*, **25**(11), 1422–1423.

Gremme, G. *et al.* (2013). GenomeTools: a comprehensive software library for efficient processing of structured genome annotations. *IEEE/ACM Trans Comput Biol Bioinform*, **10**(3), 645–656.

Lawrence, M. *et al.* (2009). rtracklayer: an R package for interfacing with genome browsers. *Bioinformatics*, **25**(14), 1841–1842.

McKinney, W. (2010). Data structures for statistical computing in python. In S. van der Walt and J. Millman, editors, *Proceedings of the 9th Python in Science Conference*, pages 51 – 56.

Quinlan, A. R. (2014). BEDTools: The Swiss-Army Tool for Genome Feature Analysis. *Curr Protoc Bioinformatics*, **47**, 1–34.

Reese, M. G. *et al.* (2010). A standard variation file format for human genome sequences. *Genome Biol.*, **11**(8), R88.

Wang, L. *et al.* (2013). CPAT: Coding-Potential Assessment Tool using an alignment-free logistic regression model. *Nucleic Acids Res.*, **41**(6), e74.

Zerbino, D. R. *et al.* (2018). Ensembl 2018. *Nucleic Acids Res.*, **46**(D1), D754–D761.

Bioinformatics
doi.10.1093/bioinformatics/xxxxxx
Advance Access Publication Date: Day Month Year
Application Note

OXFORD

Genome analysis

OLOGRAM : Determining significance of total overlap length between genomic regions sets

Q. Ferré [1,2,3,†], G. Charbonnier [1,3,†], N. Sadouni [1,3], F. Lopez [1,3], Y. Kermezli [1,3,4], S. Spicuglia [1,3], C. Capponi [2], B. Ghattas [5], D. Puthier [1,3,*]

[1]Aix Marseille Univ, INSERM, UMR U1090, TAGC, Marseille, France, [2]Aix Marseille Univ, CNRS, UMR 7020, LIS, Qarma, Marseille, France, [3]Equipe Labellisée LIGUE contre le Cancer, [4]Tlemcen University, The Laboratory of Applied Molecular Biology and Immunology, Algeria, [5]Aix Marseille Univ, CNRS, UMR 7373, IMM, Marseille, France.

*To whom correspondence should be addressed. †These authors contributed equally.

Associate Editor: XXXXXXX

Received on XXXXX; revised on XXXXX; accepted on XXXXX

Abstract

Motivation: Various bioinformatics analyses provide sets of genomic coordinates of interest. Whether two such sets possess a functional relation is a frequent question. This is often determined by interpreting the statistical significance of their overlaps. However, only few existing methods consider the lengths of the overlap, and they do not provide a resolutive p-value.
Results: Here, we introduce *OLOGRAM*, which performs overlap statistics between sets of genomic regions described in BEDs or GTF. It uses Monte Carlo simulation, taking into account both the distributions of region and inter-region lengths, to fit a negative binomial model of the total overlap length. Exclusion of user-defined genomic areas during the shuffling is supported.
Availability: This tool is available through the command line interface of the *pygtftk* toolkit. It has been tested on Linux and OSX and is available on Bioconda and from https://github.com/dputhier/pygtftk under the GNU GPL license.
Contact: denis.puthier@univ-amu.fr
Supplementary information: Supplementary data are available at *Bioinformatics* online.

1 Introduction

Current genomic analysis methods can localize a variety of sets of genomic regions, such as epigenomic features, resulting in a BED file giving their coordinates. To determine whether two such sets have a functional relationship, a typical approach is to look for significant co-localization by assessing the statistical significance of the amount of overlap between them (Haiminen *et al.*, 2008).

A comprehensive review of such methods is available through the *Coloc-stats* web interface (Simovski *et al.*, 2018), showing the biggest difference between them to be their null model. Many, such as *GREAT* (McLean *et al.*, 2010) or *CEAS* (Ji *et al.*, 2006) use a binomial test considering only the intersections of the peak centers with the query regions, while *BEDTOOLS fisher* (Quinlan and Hall, 2010) uses the number of intersecting "bins" (whose size depends on the input regions) to compute a hypergeometric test.

Generating an empirical null distribution by random shuffling of the regions within the sets is another possibility. For example, *pybedtools* incorporates a wrapper for this (Dale *et al.*, 2011) which was also used to tackle the N-fold overlap problem (Aszódi, 2012). For a more realistic null model, conservation of inter-segment length during the shuffling was first proposed by the *Genomic HyperBrowser* (Sandve *et al.*, 2010). However, the p-value they provide is only empirical and limited in its resolution by shuffling depth, itself limited by computation time.

Here we propose a new method, implemented in a tool named *OLOGRAM (OverLap Of Genomic Regions Analysis using Monte Carlo)*, to conveniently assess the significance of overlaps by fitting a Negative Binomial model on overlap statistics of interest via a Monte Carlo method.

2 Methods

2.1 Permutation and intersection computation

Let A and B be two sets of genomic regions with no overlaps within A nor B. For each subset $E_{A,k}$ (resp. $E_{B,k}$) of A (resp. B) only for chromosome k, let $L(E_{A,k})$ and $I(E_{A,k})$ be respectively the lists of regions' sizes and inter-regions distances (from end to start).

A shuffle is generated by performing independent random permutations of $L(E_{A,k})$ and $I(E_{A,k})$ for all chromosomes separately, and separately for A and B. This method differs from the classical *BEDTOOLS shuffle* which sets regions at random positions. The Genome HyperBrowser showed the relevance of this idea.

Our approach can also exclude regions from the shuffle by shuffling across a shorter, concatenated "sub-genome" generated by removing the excluded regions from both sets. This allows to compute enrichment relative to the genome minus excluded regions. For example, one can remove low mappability regions, or consider only accessible (i.e. DNAse I HyperSensitive) regions.

The tool then computes the regions' intersections between the i^{th} shuffle of A and the i^{th} of B, for all shuffles. This is done in RAM with a custom sweep-line (Shamos and Hoey, 1976) algorithm of $O(n)$ complexity to avoid disk I/O overhead. As intersections are only computed once per shuffle, the use of other algorithms such as Interval Trees with $O(n \log(n))$ complexity is not justified.

2.2 Discussion of statistical modeling

The null hypothesis (H_0) is that the regions of A are located independently of B. As such, we do not expect them to overlap more than expected by chance, if the regions were independently randomly placed on the genome.

Here, we propose a new statistical framework to model this problem. Under (H_0), for all regions A_i of A and B_j of B, consider the Bernoulli random variables $I_{i,j} = \mathbb{1}_{A_i \cap B_j \neq \varnothing}$.

They have very small probabilities $p_{i,j}$ (region sizes are typically small relative to chromosome size), that differ (each region has a different length, hence different intersection probability), and are dependent (the regions do not overlap).

Let N be the number of intersections and S the total number of overlapping nucleotides. Then $N = \sum_{i,j} I_{i,j}$ is a sum of dependant Bernoulli r.v. and can be modeled with a beta-binomial (Yu and Zelterman, 2008), itself modeled with a Negative Binomial. Unlike with *BEDTOOLS shuffle*, the dependency of the $I_{i,j}$ makes Poisson modeling unadapted.

Then consider $S = \sum_{i,j} \Lambda_{i,j}$ where $\Lambda_{i,j}$ is the length of the intersection between A_i and B_j. This sum has N nonzero terms, making it a Compound Negative Binomial. Furthermore, empirically $\Lambda_{i,j}$ will often follow a logarithmic distribution, so S can be approximated via a negative binomial (Omair *et al.*, 2018).

The assumptions taken here are confirmed in practice by a fitting test. Consequently, we reckon our model is plausible with N and S following negative binomial distributions of under (H_0) unknown parameters, approximated via this Monte Carlo approach. As such, we use them as test statistics: the p-value associated to their value in the true data is used to accept or reject the alternative hypothesis (H_1) that the regions of the query tend to overlap the reference.

3 Implementation

Our method is implemented as a plugin to *pygtftk* (Lopez *et al.*, 2019) and can be passed a GTF/BED stream or file (examples in documentation and Supplementary Data). Most of the code is written in Python 3, with performance-critical operations written in C++ and/or Cython (Behnel *et al.*, 2011). To preserve RAM, the total number of shuffles to be computed is divided into batches.

The tool will compute the overlap between the supplied BED region file and *(i)* any desired GTF feature, or *(ii)* features derived from GTF file attributes (e.g "gene_biotype"), or *(iii)* additional regions supplied as BEDs. It will output overlap statistics and the associated p-values.

The computing cost scales with the total number of lines in the reference and query files. A typical pairwise enrichment analysis of 10k regions against 10k takes 62 seconds on an 2,5 GHz Intel Core i7 processor. 200k against 200k takes 11 minutes.

3.1 Results

Suppl. Table 1 presents the applicability conditions and functionalities of various tools and approaches including *GREAT*, *CEAS*, *Bedtools Fisher*, *Genomic HyperBrowser* and *LOLA* (Sheffield and Bock, 2016).

An example of *OLOGRAM* output is available in *Suppl. Fig. 1*. We showcase interactions with *pygtftk* in *Suppl. Fig. 2*, and the importance of considering both S and N in *Suppl. Fig. 3*.

Using biological and artificial testing data, we found both S and N indeed follow a negative binomial distribution; this is shown in particular in *Suppl. Fig. 4* with the example of S on artificial data. A small total number of shuffles results in a noisy distribution, but whose two first moments (expectation, variance) remain similar than with a larger number of shuffles, making them sufficient to estimate the underlying distributions. We believe 200 shuffles (default parameter) to be an acceptable compromise between computing cost and precision of evaluation in most cases.

Fitting a distribution (as opposed to an empirical p-value) allows for better assessment of extreme overlaps presumably not encountered while shuffling. To confirm the goodness of fit, a fitting quality is given as $1 - V$ where V is Cramér's V score (Cramér, 1946) for the contingency table of observed vs. expected histogram bins. It works best when the individual probability of intersection is not too small, meaning the query and reference regions are not too small and/or scarce compared to each other.

We compare our tool to other existing approaches in *Suppl. Table 2*, showing that OLOGRAM can provide meaningful insights by being resolutive at low p-values. Discussion of those results can be found in *Suppl. Note 1*. The full code to reproduce the analyses presented is available at : https://github.com/dputhier/ologram_supp_mat, showcasing Snakemake integration.

4 Conclusion

We have implemented a method which allows to consider the information found in the number of overlapping base pairs, with a shuffling paradigm that conserves inter-region length, used to fit a negative binomial model. New features are being developed, including support for multiple overlaps between $n \geq 2$ sets.

Funding

Q.F, G.C., N.S., S.S. and D.P. were supported by recurrent funding from INSERM and Aix Marseille Univ and specific grants from A*MIDEX (A-M-AAP-EI-17-63-170228-17.32-SPICUGLIA-HLS), Institut National du Cancer (PLBIO018-031 INCA_12619) and Ligue contre le Cancer (Equipe Labellisée). Y.K. was supported by the Franco-Algerian partenariat Hubert Curien (PHC) Tassili (15MDU935)

References

Aszódi, A. (2012). MULTOVL: fast multiple overlaps of genomic regions. *Bioinformatics*, **28**(24), 3318–3319.

Behnel, S. *et al.* (2011). Cython: The best of both worlds. *Computing in Science Engineering*, **13**(2), 31 –39.

Cramér, H. (1946). *Mathematical Methods of Statistics*. Princeton University Press.

Dale, R. K. *et al.* (2011). Pybedtools: a flexible python library for manipulating genomic datasets and annotations. *Bioinformatics*.

Haiminen, N. *et al.* (2008). Determining significance of pairwise co-occurrences of events in bursty sequences. *BMC bioinformatics*, **9**, 336.

Ji, X. *et al.* (2006). CEAS: cis-regulatory element annotation system. *Nucleic Acids Research*, **34**, W551–W554.

Lopez, F. *et al.* (2019). Explore, edit and leverage genomic annotations using python GTF toolkit. *Bioinformatics*.

McLean, C. Y. *et al.* (2010). GREAT improves functional interpretation of cis-regulatory regions. *Nature Biotechnology*, **28**(5), 495–501.

Omair, M. A. *et al.* (2018). A bivariate model based on compound negative binomial distribution. *Revista Colombiana de Estadística*, **41**(1), 87–108.

Quinlan, A. R. and Hall, I. M. (2010). BEDTools: a flexible suite of utilities for comparing genomic features. *Bioinformatics*, **26**(6), 841–842.

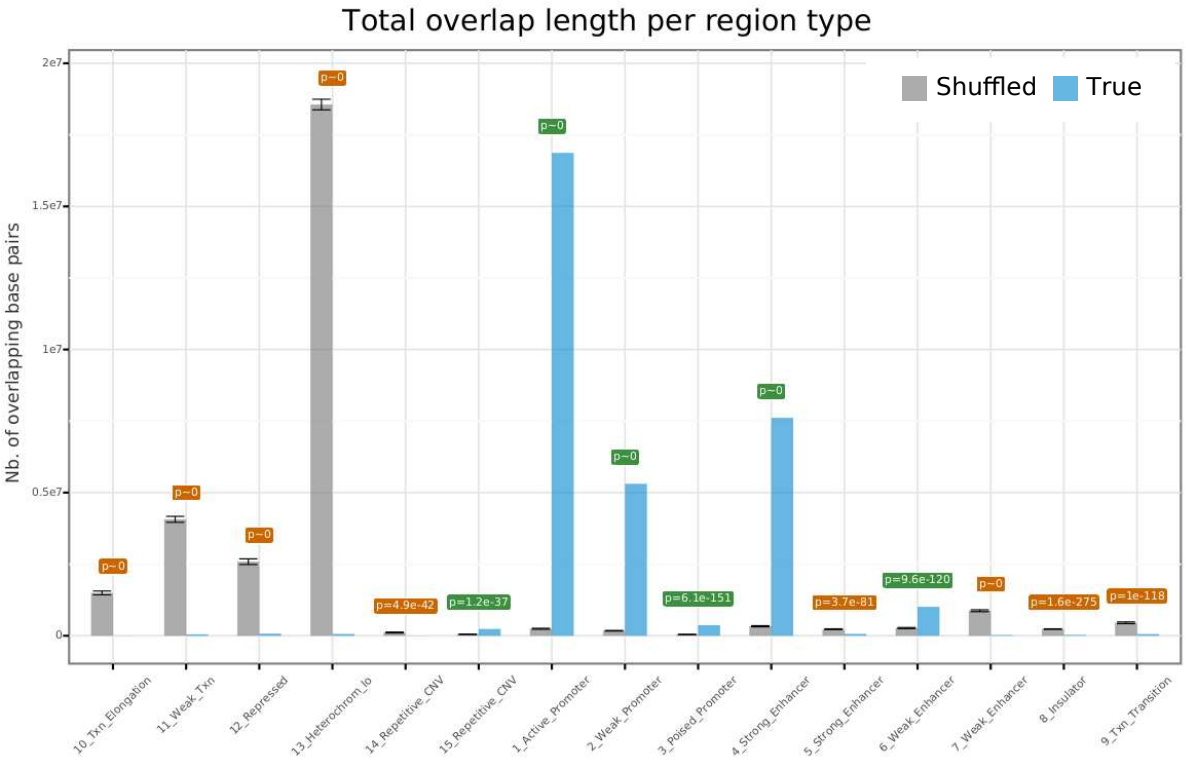
Sandve, G. K. *et al.* (2010). The genomic HyperBrowser: inferential genomics at the sequence level. **11**(12), R121.

Shamos, M. I. and Hoey, D. (1976). Geometric intersection problems. In *17th Annual Symposium on Foundations of Computer Science (sfcs 1976)*, pages 208–215.

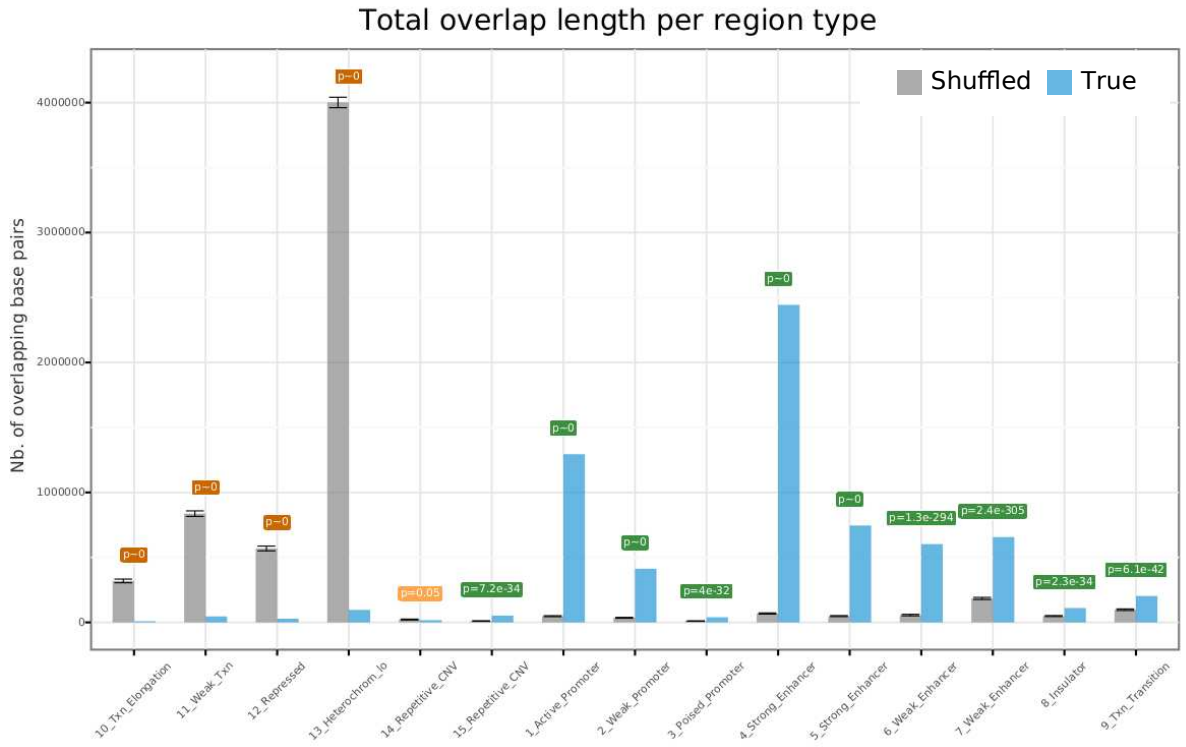
Sheffield, N. C. and Bock, C. (2016). LOLA: enrichment analysis for genomic region sets and regulatory elements in r and bioconductor. **32**(4), 587–589.

Simovski, B. *et al.* (2018). Coloc-stats: a unified web interface to perform colocalization analysis of genomic features. *Nucleic Acids Research*, **46**, W186–W193.

Yu, C. and Zelterman, D. (2008). Sums of exchangeable bernoulli random variables for family and litter frequency data. *Computational Statistics & Data Analysis*, **52**(3), 1636–1649.



A



B

Supplementary Figure 1

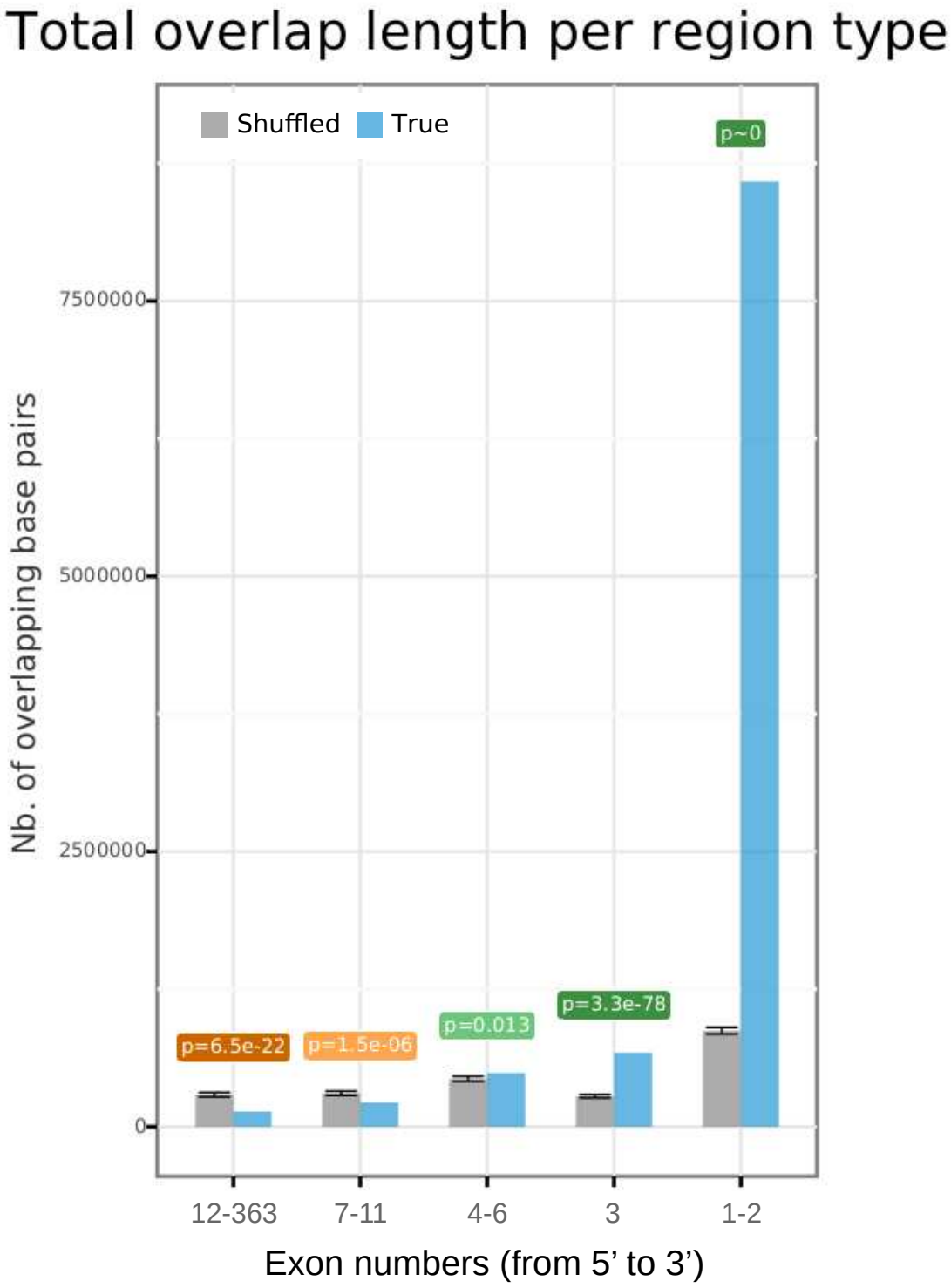
Example of OLOGRAM results, calculating the significance of intersections between :

A – H3K4me3 vs ChromHMM states

B – EP300 vs ChromHMM states

As expected, EP300 is mostly enriched in enhancer-associated states, and H3K4me3 in promoter-associated ones.

The EP300 peaks and H3K4me3 peaks come from ENCODE datasets, respectively ENCFF433PKW and ENCFF616DLO, in the K562 cell line. The ChromHMM states used are available as the wgEncodeEH000790 dataset, lifted over from hg19 to hg38.

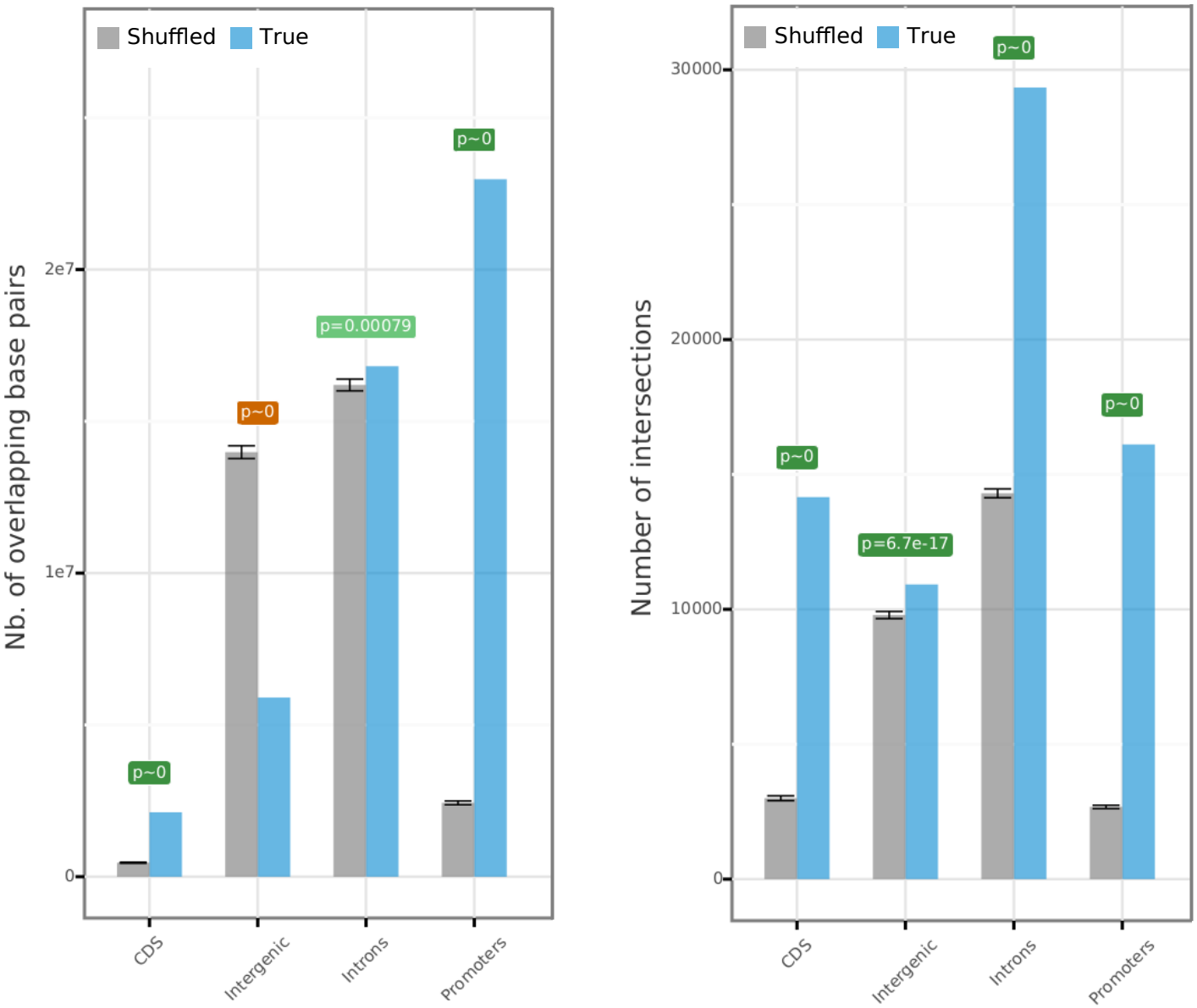


Supplementary Figure 2

Example of analysis result using OLOGRAM and providing as input a GTF treated by *pygtftk.* Here, exons have been numbered for each gene from 5' to 3'. Results are ordered from most depleted group to the most enriched.

We calculate the significance of intersections between H3K4me3 peaks and GTF-defined numbered exons. The peaks are much more present in the first exons, likely due to the broadness of H3K4me3 peaks.

This uses the same datasets as in Supplementary Figure 1. The GTF used is the Ensembl human GTF (hg38, release 92).

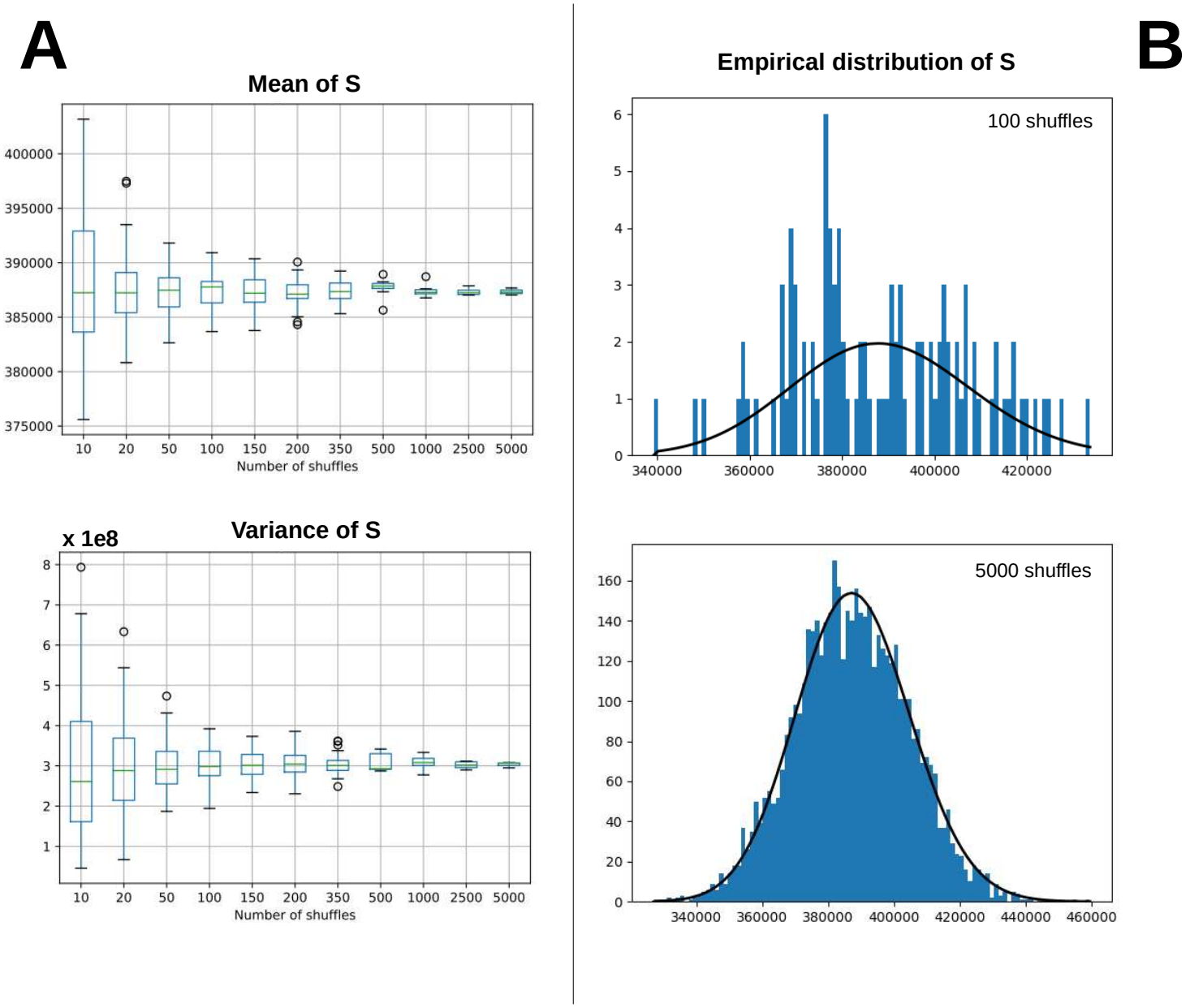


Supplementary Figure 3

Further example of OLOGRAM result. Computing the significance of intersections between H3K4me3 peaks and the regions defined in the hg38 Ensembl GTF. The H3K4me3 peaks used are the same as in Suppl. Fig. 1, and the GTF the same as in Suppl. Fig. 2.

We compare the results for S (total number of overlapping base pairs) and N (number of intersections) for a subset of GTF elements. For example, the peaks appear to be significantly enriched in introns based on N, but for S that it is not the case ; and vice-versa for intergenic regions.

Hence S is an important statistic to consider : in this particular example it may mean that the overlaps of peaks and introns are frequent but short. Note that here, an "intersection" means having at least one nucleotide in common.



Supplementary Figure 4

Evolution of OLOGRAM estimation precision for the S statistic (number of overlapping base pairs) under (H0) based on the number of shuffles. For this figure, both query and reference datasets are artificial data (10 000 regions of length 1000 in an artificial genome).

(A) For the S statistic, evolution of empirical mean and variance for different tries as a function of the number of shuffles in each try.

(B) Example of empirical distribution of S with corresponding Negative Binomial distribution of same mean and variance for 100 and 5000 shuffles.

The precision of the estimation of the mean and variance increases with the number of shuffles. We also see that the S statistic tends towards a Negative Binomial distribution when the number of shuffles is large enough, and that the moments of this distribution (expectancy, variance) can still be estimated with acceptable precision using relatively few shuffles.

		OLOGRAM	Genomic HyperBrowser (4)	GREAT	CEAS	Bedtools fisher	LOLA
Statistical test	Statistical test used	Neg. Binomial	MCFDR (5)	Binomial	Binomial	Fisher's exact test	Fisher's exact test
	Null model conserves intersegment length	Yes	Yes	No	No	No	No
	Statistical test cares about region size	Yes	Yes	No	No	Yes	Yes
	Statistics of interest: number of intersections between query and reference (N)	Yes	Yes	Yes	Yes	No	No
	Statistics of interest: number of intersections with genomic bins or universe regions	No	No	No	No	Yes	Yes (2)
	Statistics of interest: sum of overlapping base pairs between query and reference (S)	Yes	Yes	No	No	No	No
	Test is recommended for broad regions	Yes	Yes	No	No	Yes	Yes
	Test is recommended for short regions	Yes	Yes	Yes	Yes	Yes	Yes
Total number of regions in query and reference	Approach designed for few query regions (tens-hundreds)	No (1)	No	Yes	Yes	Yes	yes
	Approach designed for numerous query regions (tens to hundreds of thousands)	Yes	Yes	Yes	Yes	Yes	yes
	Approach designed for very numerous query regions (millions)	No (1)	No	Yes	Yes	Yes	yes
Reference	Support for any organism annotation (user-supplied)	Yes	No	No	Yes	Yes	yes
	Independent from online database update	Yes	No	No	Yes	Yes	Yes
	Can interpret GTF keys/values (original or custom) as annotations	Yes	No	No	No	No	No
	Native support for functional annotation databases (e.g. GO-term, MSIGDB)	No	No	Yes	No	No	No
Usage	Very fast computations	No	No	Yes	Yes	Yes	Yes
	Conda installation (versioning, reproductibility)	Yes	No	No	Yes	Yes	Yes
	Web interface	No	Yes	Yes	No	No	No
	Command Line Interface (workflow intergration)	Yes	No	No	Yes	Yes	No
	Graphical representation of the results (diagrams)	Yes	No	Yes	Yes	No	Yes
Query customization	CLI command chaining/piping to customize query regions (e.g add novel keys to a GTF)	Yes	Yes	No	No	Yes	Yes
	Support for user defined background (region exclusion)	Yes	Yes	Yes	No	No	Yes
	Easy integration of several functional analyses into a single report	Yes (3)	No	No	No	No	No
Community	Full code available in Github (can be forked)	Yes	Yes	No	No	Yes	Yes

Supplementary Table 1

Comparison of features and assumptions of various enrichment analysis tools.

Software versions: OLOGRAM (pygtftk, v01.9.7), GREAT (accessed 10 july 2019), CEAS (1.0.2), Bedtools fisher (2.28.0), HyperGenome Browser (v2.0b5), LOLA (1.14.0).

The OLOGRAM statistic used is S (number of overlapping base pairs).

NB: HypergenomeBrowser contains several tools for statistical analysis of overlaps. Here we are comparing to HYPERBROWSER > analyse genomic track > Hypothesis testing > overlap > Null model : preserve segment lengths and intersegment-gaps; randomize positions (T1 & T2) > MCFDR sampling depth: Fixed 10 000 samples"

(1) This can be done, but will be time-consuming. Few regions would need lots of shuffles for a clean NB fitting, and lots of regions will be long to shuffle.
(2) Fixed size bins or genomic regions (region universe) must be fully defined a priori by the user.
(3) This functionnality is available in the 'merge_ologram_stats' plugin of gtftk. See corresponding CLI and documentation.
(4) Comparison is done with the tool "Statistical analysis of tracks > hypothesis testing > overlap " with null model "preserve segment, segment length and intersegment gaps and MCFDR set to fixed for best precision.
(5) Monte Carlo (MC) and False Discovery Rate (FDR). The resulting p-value is based on the empirical distribution of the test statistic in the shuffles, without fitting.

	Promoters (alt. Hyp 'more')	Introns (alt. Hyp 'more')	Exon 1 to 2 (alt. Hyp 'more')	Exon 3 (alt. Hyp 'more')	Exon 4 to 6 (alt. Hyp 'more')	Exon 7 to 11 (alt. Hyp 'less')	Exon 11 to 363 (alt. Hyp 'less')	Random regions (2) (alt. Hyp 'more')	Representative time required (1)
Genomic HyperBrowser (Moderate resolution of global p-value)	0.003984	0.003984	0.003984	0.003984	0.003984	0.003984	0.003984	0.25	~10 min
Genomic HyperBrowser (Moderate resolution of global and local p-value)	0.003984	0.003984	0.003984	0.003984	0.01105	0.003984	0.003984	0.3	~10 min
Genomic HyperBrowser (Fixed 10 000 samples)	9.999e-05	9.999e-05	9.999e-05	9.999e-05	0.0112	9.999e-05	9.999e-05	0.29	~5 h
Bedtools Fisher	0	0	0	0	4.67e-104	0.09	1.3751e-32	0.97	~1 sec
Binomial test on number of intersections with H3K4me3 midpoints (5)	0	1 (6)	0	8.31e-47	6.52e-13	0.24	2.76e-4	0.36	~1 sec
Binomial test on number of intersections with H3K4me3 (5)	0	0	0	0	0	1.71e-135	1.25e-44	1.56E-50	~1 sec
OLOGRAM	1e-320	0.00079	1e-320	3.3e-78	0.0013	1.5e-6	6.5e-22	0.26	~2 min*

Supplementary Table 2

Summary of p-values obtained when testing overlaps of H3K4me3 on various genomic regions using enrichment analysis tools.

Promoters are defined here has merged regions -1kb and+1kb around the TSS. Introns correspond to all merged genic regions with no exonic overlap. Query file was H3K4me3 sites from the ENCFF616DLO ENCODE experiment and all reference regions were extracted from Ensembl GTF (hg38, release 92).

The OLOGRAM statistic used here is S (number of overlapping base pairs).

(1) Except for Genomic HyperBrowser all tests were performed on a MacBook Pro 2,5 GHz Intel Core i7 on a single core.
(2) 20 000 random regions of size 1000.
(3) 20 000 random regions of size 1.
(4) The computation of p-values was performed using a decicated script available through github.
(5) Here the binomial test report a depletion with pvalue equal to 1.45e-11.

All code is available in the following github repository: https://github.com/dputhier/ologram_supp_mat

From what we know about H3K4me3, this epigenetic mark is highly enriched in promoters and is known to also overlap the first exon, as peaks can be quite broad. This table reports the significance of enrichment/depletion of H3K4me3 overlaps observed using representative tools in various genomic regions. OLOGRAM reports the expected results with a very highly significant enrichment in promoters and first/second exons (p<1e320 being the minimum value), progressively depelting with further exons.

Supplementary Note 1

In Supplementary Table 1, we tried to clarify the pros and cons of various tools including OLOGRAM, GREAT, CEAS, Bedtools Fisher, Genomic HyperBrowser and LOLA. This table shows that the tools and results are not easily comparable, as their conditions of applicability and underlying assumptions are different. We tried anyways to offer some insight by performing such a comparison with a set of representative datasets in Supplementary Table 2.

We used the well-known H3K4me3 epigenetic marks with results previously obtained in K562 cell line (ENCODE project ID ENCFF616DLO) and checked its overlaps with several genic elements including promoters, exons (number 1 to 2, 3, 4 to 6, 7 to 11, 12 to 363) and introns.

From what we know about H3K4me3, this epigenetic mark is highly enriched in promoters and is also known to overlap the first exon. Depending on whether the H3K4me3 signal is broad and on the size of the most 5' exonic and intronic regions, overlaps may also be encountered in more 3' exons/introns relative to the TSS. However, as we are moving away from the promoter in 3' direction the enrichment in H3K4me3, overlaps are expected to decrease and ultimately become a depletion as the H3K4me3 (concentrated in promoters) should be less frequent in these regions than expected by chance.

We report the significance of enrichment/depletion of H3K4me3 overlaps observed using representative tools in various genomic regions. CEAS was not included, as it does not currently offer an annotation for hg38. GREAT was not included as it works by design on defined regulatory regions. However both CEAS and GREAT may be considered as solutions that use internally the binomial test for computing their p-values. As a comparison, we thus computed p-values from the binomial tests both using the number of intersecting peaks or their corresponding midpoints as statistic.

OLOGRAM reported the expected results, with a very highly significant enrichment in promoters and first/second exons ($p<1e320$ being the lower limit), a very significant enrichment in third exons, a moderate enrichment in Exon 4 and 6 while, starting from exons 7 to 11, H3K4me3 become increasingly depleted.

Interestingly, the same trends (enrichment then depletion) were reported by HyperGenomic Browser that uses the same kind of null model as OLOGRAM (Monte-Carlo simulation preserving both segment and intersegment length), but with empirical p-value computation instead of model fitting. However, when used with moderate resolution the p-value was floored to 0.003984, while it was floored to 9.999e-05 (1/10001) when run with the parameter 'Fixed 10 000 samples'. We assume that this “0.003984” value was due to the fact that hyperBrowser uses 250 permutations by default ($0.003984 \approx 1/251$). Moreover the time needed for computation was about 5 hours for a typical run with 'Fixed 10 000 samples', while OLOGRAM took about 2 minutes to return a far more resolutive p-value.

Bedtools reported very strong enrichment in promoters, introns, exons 1 to 2 and exon 3 with p-values equal to zero. Similarly, a very highly significant enrichment was observed in exons 4 to 6. Exons 7 to 11 were reported not to be significantly depleted while exons 11 to 363 were reported to be significantly depleted. This result is completely in accordance with the known limitations of Bedtools Fisher. This aspect is covered in details in the Bedtools Fisher

documentation (https://bedtools.readthedocs.io/en/latest/content/tools/fisher.html) that highlight a strong tendency of Bedtools Fisher to produce lower p-values as compared to Monte-Carlo methods at the cost of potentially producing false positives. As such the Bedtools authors recommend validating low p-values from fisher using simulation. In this regard, Bedtools Fisher should be more considered as a fast screening solution producing results to be double-checked with more precise solutions such as OLOGRAM.

A binomial test computed from the number of overlapping H3K4me3 midpoints reported a very high enrichment in promoters and exons 1 to 2, together with a high enrichment in exon 3 and exons exons 4 to 6. While no depletion was observed in exon 7 to 11 a moderate depletion was observed in exon 11 to 363. Introns were found to be depleted in midpoints of H3K4me3 in contrast to OLOGRAM that reports an enrichment through a very highly significant N value (see supplementary figure 3) and a significant value of S. This underlines that focusing on peak centers using a binomial test or full peaks overlaps through S or N statistics, as proposed by OLOGRAM, may lead to seeming discrepancies. However, these results are easy to reconcile and here the results may be interpreted as: 'While H3K4me3 peaks centers are depleted in introns, short overlaps of H3K4me3 with introns are more frequently observed than expected by chance'.

Finally, a binomial test taking into account overlaps counts with full peaks leads to false positive as underlined by very highly significant p-values observed even with random regions.

We think that this benchmark underlines that OLOGRAM results are very meaningful and complementary to approaches relying on binomial tests. It also clearly underlines the limits of using Monte-Carlo approaches without model fitting.

All the code required for reproducing the results from supplementary figures 1, 2, 3 and supplementary table 1 and 2 is available as a Snakefile workflow in a dedicated GitHub repository: https://github.com/dputhier/ologram_supp_mat.

Monte Carlo based mining of enriched *n*-wise combinations of genomic features with dictionary learning

Q. Ferré[1,2], C. Capponi[2], and D. Puthier[1]

[1]**Aix Marseille Univ, INSERM, UMR U1090, TAGC, Marseille, France**
[2]**Aix Marseille Univ, CNRS, UMR 7020, LIS, Qarma, Marseille, France**

ABSTRACT

Most epigenetic marks, such as Transcriptional Regulators or histone marks, are biological objects known to work together in *n*-wise complexes. A suitable way to infer such functional association between them is to study the overlaps of the corresponding genomic regions. However, the problem of the statistical significance of *n*-wise overlaps of genomic features is seldom tackled, which prevent rigorous studies of *n*-wise interactions.
We introduce *OLOGRAM-MODL*, which considers overlaps between $n \geq 2$ sets of genomic regions, and computes their statistical mutual enrichment by Monte Carlo fitting of a Negative Binomial distribution, resulting in more resolutive p-values. An optional machine learning method is proposed to find complexes of interest, using a new itemset mining algorithm based on dictionary learning which is resistant to noise inherent to biological assays. The overall approach is implemented through an easy-to-use CLI interface for workflow integration, and a visual tree-based representation of the results suited for explicability. The viability of the method is experimentally studied using both artificial and biological data.
This tool is available through the command line interface of the *pygtftk* toolkit on Bioconda and from `https://github.com/dputhier/pygtftk`.
Contact: denis.puthier@univ-amu.fr, Cecile.Capponi@lis-lab.fr

Keywords: genomic regions, combinations, overlap, machine learning, statistical modeling, itemset mining, Monte Carlo

1 INTRODUCTION

Modern genomic analysis methods can localize many different types of genomic features, such as histone modifications, Transcriptional Regulator binding sites, or gene promoters. As such, a fundamental question arises: do those *sets* of features have a functional association? A typical approach is to represent such features as regions, or intervals (hence, as BED files [1]) and look for significant co-localization through the statistical significance of the amount of overlap between them, against (H_0) of overlapping no more than by chance. Indeed, co-localization is often associated to functional association in genomic elements (Biggar and Crabtree, 2001).

Pairwise overlaps between two sets can be analyzed with methods such as *GeometriCorr*, *BEDTOOLS fisher* (Quinlan and Hall, 2010), *GREAT*, *Genomic HyperBrowser* (Sandve et al., 2010), mostly available in the *coloc-stats* interface (Simovski et al., 2018). Those methods are usually based on shuffles or on a statistical model. Challenges in such approaches have been summarized in a recent review (Kanduri et al., 2019). Recently, Ferré et al., 2020 proposed another type of method involving Monte Carlo fitting of a Negative Binomial distribution while keeping inter-region distances, proven to be more resolutive than previous approaches. However, considering only pairwise overlaps cannot reveal higher order associations between a query interval set and multiple reference sets simultaneously. Indeed, most chromatin components such as Transcriptional Regulators or histones are known to work in combinations and form complexes (Lambert et al., 2018) when binding to the genome. As such, a method is required in order to rigorously evaluate those combinations. Pairwise overlaps are sometimes used to build association

[1]See `https://genome.ucsc.edu/FAQ/FAQformat.html#format1`

networks (Meckbach et al., 2015) but this can be misleading, as an association of a regulator A with B and of B with C does not necessarily mean A and C will be found in the same complex in real conditions.

However, the problem of the significativity of multiple overlaps is rarely tackled. Some existing approaches include *MULTOVL* (Aszódi, 2012) which uses empirical p-values determined from shuffling the region sets to determine the statistical enrichment of higher-order associations. Furthermore, simply evaluating the enrichment of all n-wise combinations of k sets returns up to 2^k possibilities, which can be hard to parse. To filter those, other current approaches such as *TFCoop* (Vandel et al., 2018) look for combinations of factors that best explain another, but use linear regressions which does not show the diversity of existing complexes, instead giving a weight to each set.

Itemset mining, which groups many methods aimed at identifying patterns between sets (Luna et al., 2019, *i.e.* when an element of set A is present, sets B and C are often present as well) has also been used for to identify interesting combinations of genomic regions (Teng et al., 2014). For instance, *GINOM* selects n-wise itemsets that best explain the query region set (Bryner et al., 2017). A more distant parallel can also be drawn to *ChromHMM* (Ernst and Kellis, 2012) which however divides the genome in mutually exclusive states without hierarchizing combinations. Although itemset mining is mostly performed with tree based algorithms (Chee et al., 2019) such as APRIORI (Agrawal and Srikant, 1994), some advances are made with non-negative matrix factorization, including inferring TF (Transcription Factors) combinations (Giannopoulou and Elemento, 2013), and with dictionary learning (Mansha et al., 2018).

Another reason to use itemset mining to identify combinations of interest is the presence of noise. For example in ChIP-seq, which is a technique used to locate binding sites of proteins on the genome, there are known difficulties resulting in false positive peaks, either for biological or technical reasons (Marinov et al., 2014). This may complicate analysis leading to spurious results. Some methods seek to correct the noise, sometimes also leveraging combinations between sets (Koh et al., 2017). In particular, matrix factorization methods are quite effective although costful on such noisy data (Mairal et al., 2009).

However, using itemset mining to find combinations of interest based on a criterion and assessing their enrichment are two different approaches, which are worthwhile to now be combined. This paper proposes a method named **OLOGRAM-MODL** to leverage both, by calculating the significance of mined combinations of overlaps of interest 2. Before discussion and conclusion, section 3 analyses the approach both on artificial and biological data through several types of experiments.

2 MATERIALS AND METHODS

As an extension of OLOGRAM (Ferré et al., 2020), OLOGRAM-MODL[2] can now process overlaps between $n \geq 2$ sets and compute statistically relevant p-values for each combination. The optional *MODL* algorithm is proposed to find interesting combinations using dictionary learning.

Definition 1. *Let A_i be a genomic region, that is a position interval on the genome (*eg. $A_i[100_1;200_1] =$ *"chromosome 1, base pairs 100 to 200"). Then, the set $A = \{A_1, A_2, ...\}$ is defined as a finite set of individual genomic regions.*

Definition 2. *A combination $\gamma = \{A + B + C\}$ is defined whenever genomic regions from A, B and C embed a common genomic position. Combinations can be defined on any $n \geq 2$ sets.*

Definition 3. *For a given combination γ, $S(\gamma)$ is the total number of base pairs on which this combination is observed.*

2.1 OLOGRAM enrichment analysis

For each combination γ, the objective is to determine whether it is observed in the real data at a higher frequency than it would be under (H_0) of no association between its constituent sets.

Definition 4. *Let γ be a combination. Its enrichment is $m(\gamma) = \log_2(\frac{S_{obs}(\gamma)}{S_{exp}(\gamma)})$ where $S_{obs}(\gamma)$ is the S statistic in real data, and $S_{exp}(\gamma)$ is the expected value of $S(\gamma)$ under (H_0).*

OLOGRAM's original principle is to determine the statistical significance of the overlap between two region sets by shuffling them independently many times, while conserving region and inter-region lengths. Exclusion by concatenation for restricting the shuffling to certain regions of the genome is possible. The

[2] *OverLap Of Genomic Regions Analysis using Monte Carlo - Multiple Overlap combinations with Dictionary Learning*

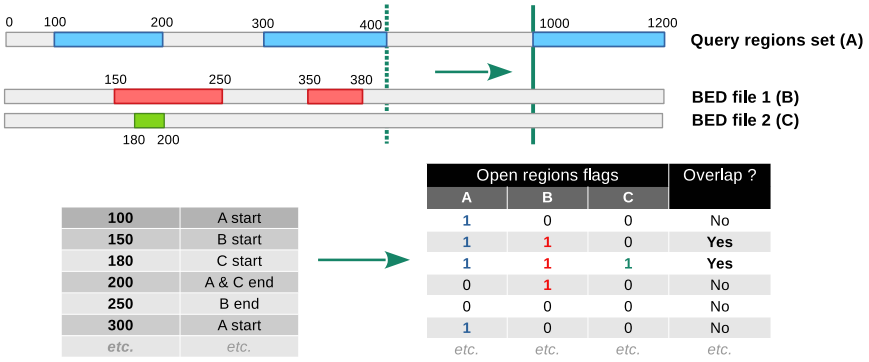


Figure 1. Multiple overlap algorithm implemented in OLOGRAM-MODL, belonging to the sweep line family. It takes as input $n \geq 2$ sets from BED files.
It registers critical points (regions' beginnings and ends) and remembers how many such points of each type have been previously encountered. For each observed overlap, the algorithm returns a vector giving the number of regions from each set that are open at this position. The output is the intersection matrix X consists of all such vectors with nonzero sum, with one line per intersection and one column per region set. Due to the partitioning of intersections, the total number of base pairs will be the statistic of interest.

key contribution was fitting a Negative Binomial model to pairwise S instead of using empirical p-values. As such, OLOGRAM has already been shown to be much more resolutive in terms of p-value compared to state-of-the-art tools. OLOGRAM-MODL generalizes it to overlaps between potentially *more than two* sets of genomic regions.

First, the real overlaps are computed and stored in a relevant matrix (section 2.1.1) from which candidate combinations can, optionally, be easily extracted. Then overlaps are also computed on shuffles (section 2.1.2) and used to statistically model the enrichment of the combinations (section 2.1.4).

2.1.1 Multiple intersection algorithm

An algorithm based on the sweep line principle (Shamos and Hoey, 1976) has been designed (Figure 1), which takes as inputs many sets of regions and returns a matrix representation of their overlaps. Although this algorithm has a time complexity of $O(N \log N)$ where N is the total number of regions in all sets at initialization[3], querying overlaps has a complexity of $O(\sum n_i)$ where n_i is the number of regions in the i^{th} set. It contrasts with interval trees (used in *BEDTOOLS fisher*) whose complexity is $O(\log n)$ per queried region, simplified to $O(n_2 \log n_1)$ for the full overlaps between just two sets.

2.1.2 Computing the combination enrichment

For each combination γ, the number of times and base pairs in which is it encountered in the shuffles is used to fit a Negative Binomial model, and give the p-value for the number of actual observed occurrences happening by chance. By default, the approach computes the enrichment of all combinations observed in the real data. To select among those, the MODL algorithm (see section 2.2) is useful. The user can also provide a custom selection. Only combinations containing the specified query region are considered.

2.1.3 Parent combinations and inexact counting

Definition 5. *A combination γ_1 may include all the sets of a combination γ_2, plus some others: γ_2 is the parent and γ_1 is the child of the relationship, denoted by $\gamma_2 \preceq \gamma_1$.*

Definition 6. *Unlike in an exact counting, in a transitive (i.e. inexact) counting of a combination γ, any observation of the children of γ is counted as an unduplicated observation of γ.*

For example, $A+B$ is a parent of $A+B+C$ or $A+B+D$, but not of $B+C$. In a transitive counting, $A+B$ represents all combinations of type $A+B+\Delta$ where Δ is any set of regions other than A and B, and is labeled as such. Counting is transitive by default, as this allows easier study of cases where multiple regulators can have a combined effect and not be mutually exclusive. An exact counting will instead

[3] It requires sorting the critical points through a merge sort (Merrett, 1983)

show cases when, for example, $A+B+C$ is the only existing complex and the $A+B$ combination is not enriched.

2.1.4 Statistical model discussion

Consider the regions sets A, B, C of a combination $A+B+C$. Under (H_0) of no association between the sets, consider the Bernoulli random variables (r.v.) $I_{A_i,B_j,C_k} = \mathbb{1}_{A_i \cap B_j \cap C_k \neq \varnothing}$.

Proposition 1. *For any combination γ, if (H_0) is true then $S(\gamma)$ can be modeled with a Negative Binomial distribution.*

Proof sketch. Consider the regions sets A, B, C of a combination $A+B+C$. Under (H_0), consider the Bernoulli r.v. $I_{A_i,B_j,C_k} = \mathbb{1}_{A_i \cap B_j \cap C_k \neq \varnothing}$. They can be broken as a product of pairwise $I_{A_i,B_j} * I_{A_i,C_k} * I_{B_j,C_j}$. Those are dependant Bernoulli r.v. for two reasons. First, the locations of the regions are permuted in the shuffles, so if A_i and B_j overlap in a shuffle the likelihood of A_i also overlapping with a different region B_k of the set B is greatly reduced, since the regions are merged. Second, let us now consider several sets so if A_i overlaps B_j and B_j overlaps C_k, it is likely than A_i also overlaps C_k.

Let us express this in terms of conditional probabilities: $P(I_{A_i,B_j,C_k}) = P(I_{A_i,B_j} = 1) * P(I_{A_i,C_k} = 1 | I_{A_i,B_j} = 1) * P(I_{B_j,C_j} = 1 | I_{A_i,B_j} = I_{A_i,C_k} = 1)$. If one approximates each term as the result of another Bernoulli variable of unknown but fixed probability p, one can approximate their products I_{A_i,B_j,C_k} themselves as dependant Bernoulli r.v. of unknown p. While calculating the p themselves requires the expression of the correlations between the variables, they can be instead estimated via a Monte Carlo approach. Indeed, Ferré et al., 2020 shows that dependent Bernoulli r.v. can be modeled with a Negative Binomial distribution, which is also true for $S(\gamma) = \sum I * \Lambda_I$ where Λ_I is the length of each intersection, if Λ is assumed to follow a log-normal distribution.

□

The distributions are fitted by the method of moments. In practice, this proposed modeling is confirmed by a fitting deal on the shuffled data (cf. section 3.1 for an example). In most cases, 100-200 shuffles is enough for a correct fit (Ferré et al., 2020), but if the shuffling is too restricted by either stringent exclusions or the use of small regions then those assumptions may no longer hold. Using a statistical model instead of empirical p-values is crucial for combinations containing a large number of sets, for which the likelihood of observing high values of $S(\gamma)$ in the shuffles will be low.

2.2 MODL itemset mining algorithm

The optional MODL *(Multiple Overlap Dictionary Learning)* algorithm for itemset mining is introduced (Algorithm 1). In OLOGRAM-MODL, it can be used to pre-select combinations of sets which are of interest, hence fairly reducing afterwards the total number of enrichment computations and making the results easier to interpret. Users who wish to study all combinations encountered in the real data can skip this section.

MODL takes as input a transactions matrix with one column per set and one line per observation. In step 1, MODL performs various reconstructions with dictionary learning with various sparsity constraints to get a set of candidate atoms. In step 2, a greedy algorithm builds the final selection by getting the best encoding candidates using the maximization of a local function with regularization. In the following, let k be the number of sets in the matrix $X \in \mathbb{R}^{m \times k}$ of m observations, and q the queried final number of itemsets is a parameter of MODL.

2.2.1 Dictionary learning for biological combination extraction

In the OLOGRAM-MODL approach, the input matrix of MODL is the matrix of overlap flags provided by the algorithm in Figure 1, with one row per overlap and one column per set in the real, non-shuffled data. However, any matrix matching this format can be used.

Figure 2 indicates the principle of dictionary learning (Mairal et al., 2009) as used by MODL, which is a factorization matrix problem with sparsity that entails solving:

$$(U^*, V^*) = \underset{U,V}{\arg\min} \frac{1}{2} \|X - UV\|_2^2 + \alpha \|U\|_1$$
$$\text{subject to } \|V_i\|_2 = 1 \text{ for all } 0 \leq i \leq n_{\text{atoms}}$$

Algorithm 1: Multiple Overlap Dictionary Learning (*MODL*) algorithm for combination mining

Data: $X \in \mathbb{N}^{m \times k}$ the matrix of m overlap flags with k sets, and q the queried number of atoms

// Pre-processing in section 2.2.2

$X \leftarrow \psi(X)$, $\Lambda \leftarrow \emptyset$

$\alpha \leftarrow \frac{1}{k}$, $i \leftarrow 0$

// Step 1 presented in section 2.2.3

while $\sum U \neq 0$ **do**

  $U, V \leftarrow$ `DictionaryLearning` $(X, \alpha, n_{\text{atoms}} = 2 * q)$ with LASSO-Coordinate Descent, LARS if fails to converge.

  foreach $v \in V$ **do**

    Binarize v

    $\Lambda[v] \leftarrow \Lambda[v] + U[v]$

  end

  $i \leftarrow i + 1$

  $\alpha \leftarrow \alpha + \frac{i}{k}$

end

Keep highest $3 * q$ atoms of Λ sorted by total usage. Remove those longer than desired.

// Step 2 presented in section 2.2.4

$V_1 \leftarrow \emptyset$, $\Lambda_1 = \Lambda$

foreach $t \in [1..T]$ **do**

  $\Delta \leftarrow \emptyset$

  foreach $\lambda \in \Lambda_t$ **do**

    $S_{\lambda,t} \leftarrow V_t \cup \{\lambda\}$

    $U_{\lambda,t} \leftarrow$ `SparseEncode` $(X, S_{\lambda,t}, \alpha = \frac{1}{k})$ with LASSO-LARS

    $\Delta(\lambda) \leftarrow \|X - U_{\lambda,t} S_{\lambda,t}\|_1 + \frac{1}{k} \sum U_{\lambda,t}$

  end

  // Get best candidate at this iteration

  $\lambda_t^* = \arg\min_{\lambda \in \Lambda_t} \Delta(\lambda)$

  $V_{t+1} \leftarrow V_t \cup \{\lambda_t^*\}$

  $\Lambda_{t+1} \leftarrow \Lambda_t \setminus \{\lambda_t^*\}$

end

Result: Learned dictionary of interesting combinations V_T

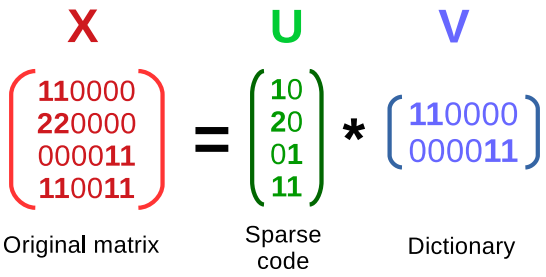


Figure 2. Principle of itemset mining via dictionary learning. The goal of dictionary learning is to learn U and V from X under certain constraints, minimizing the reconstruction error.
It can be clearly seen how the atoms (rows) of the learned dictionary V can be mined for frequent itemsets in the data, giving sets that are often present together.

It shows that relevant itemsets can be extracted from the atoms of the dictionary V. Here, an itemset corresponds to a combination as defined above. As a matrix factorization based algorithm, MODL is less vulnerable to noise than usual tree-based approaches. As such, the learned atoms can be buildings blocks referring to parts of a complex, like in the third line of figure 2 instead of minor variations of the combinations (cf. section 3.1).

A dictionary V is composed of atoms (rows of V), which are used to rebuild richer words (rows of X, combinations). Here, atoms represent biologically relevant sub-complexes. Adding redundant atoms (*i.e.* (11) if (01) and (10) are already present) if they improve the rebuilding can be warranted to represent the entire complex.

Definition 7. *The usage of an atom V_j in rebuilding a given word $\hat{X}_i = U_i V$ is $U_{i,j}$. Its total usage is $\sum_i^m U_{i,j}$.*

2.2.2 Pre-processing

Since MODL's goal is to best reconstruct the input matrix, its cost scales with matrix size, and it emphasizes combinations found in the most frequent observations. To mitigate this, a compressed version of the input matrix X is processed instead, called a smothered matrix.

Definition 8. *The abundance of a row x in a matrix X is the number of rows in X which are exactly equal to it, noted $a_X(x)$.*

Definition 9. *The smothered version of the matrix X is the matrix $\psi(X)$. For each unique row x of X, $a_{\psi(X)}(x) = \sqrt{\frac{a_X(x)}{\nu}}$, where ν is the highest of either $\min(a_X)$ or the abundance threshold τ. Row order is unimportant.*

After smothering, X is reduced to one elementary repetition of itself to save computing time. The default abundance threshold is $\tau = 1\text{e}{-4}$, combinations rarer than this are ignored. The use of the square root of the abundances gives more weight to the rarest combinations, instead of simply focusing on an even better reconstruction of frequent combinations.

2.2.3 Library creation through sparse dictionary learning

After the previous pre-processing, the first step of MODL itself is to compute a library of candidate atoms. This is done by performing several successive factorizations on $\psi(X)$ as explained in section 2.2.1. The reconstructions are repeated with different sparsity constraints α to get candidate atoms of various lengths. At each iteration, α is increased by i/k where i is the iteration number. Longer atoms are learned because a higher α allows less atoms to be used. As too low values can hinder convergence, α begins at $1/k$. Using increasing steps avoids lingering at high α, where results can be redundant. This step stops once α is so high that the total usage of all atoms is zero.

The reconstructions are repeated on a 3-fold cross-sampling (rotating 2/3 of data) to increase result variety, as random initialization can result in different reconstructions. Coordinate descent with LASSO is used for the fitting with 200 iterations, where negative atoms are disallowed to allow later interpretation. The more widespread Least-angle regression (LARS) fitting algorithm is known to select wrong features

when the features are correlated (Efron et al., 2004) and as such is is only used as a fallback if LASSO fails to converge.

A higher number of atoms in the learned dictionary would result in more precise reconstructions, but lessen the need to learn itemsets instead of individual components or simply unique rows (*i.e. words*). Conversely, fewer atoms will result in compromises. Although this is the point of the approach and grants resistance to noise (both false positive and negatives), too much compromise can result in learning potential correlation groups such as using an atom $A+B+C$ to reconstruct the words $\{A\}$, $\{B+C\}$ and $\{C\}$. As a trade-off between having variety and the effects just mentioned, the number of atoms in the learned dictionary is by default $2q$.

After each factorization, each atom v is binarized and saved along with its total usage. In binarization, as $\sum v^2 = 1$ for each atom v, the cutoff for whether a feature is considered to be used or not in a atom is $v_i^2 > 1/n^2$. Finally, once all reconstructions are done, a library Λ of candidate atoms is obtained. In order to save time, only the top $3 * q$ atoms ordered by their summed usage across all reconstructions are kept before passing Λ to the next step. This will also discard leftover atoms with low usage. An optional filtering by atom length is possible.

2.2.4 Greedy algorithm for combination selection

Now, the final q combinations constituting the final dictionary V_T will be selected by iteratively adding the atoms maximizing the fidelity f of the rebuilding:

$$V_T = \arg\max_S f(S) \text{ , where } f(S) = -\|X - US\|_1 + \alpha \sum U$$

At each iteration, the best atom λ^* of the library Λ is greedily added to the dictionary V_t, which is initially empty. For that purpose, at each step t, a two-stages optimization process is performed which first computes all the sparse approximations for all remaining candidates $U_{\lambda,t}$ of X using the current dictionary $S = V_t \cup \{\lambda\}$, and which then chooses the λ^* that minimizes the difference d_1 between X and its approximation $U_{\lambda,t}S_t$, where:

$$d_1(X, \tilde{X}) = \|X - \tilde{X}\|_1 + \alpha \sum U$$

Unlike in step 1 where U^*, V^* were optimized conjointly, here the sparse coder will find U^* for a given V_t. The sparsity controller α tends to emphasize the longest atoms: this impact is reduced by projecting each $\lambda \in S$ on the surface of the 1-unit ball ($\|\lambda\|_2 = 1$). To ensure no two atoms have the exact same dot product with a given word, a small jitter of $\frac{\sqrt{i}}{10^4}$ is added to each value of the i-th atom. S is sorted in lexicographic order.

As the set of atoms usually has some degree of degeneracy (similar atoms), the sparse coder used is LASSO-LARS. Coordinate Descent does not handle it well, but LARS tends to drop correlated regressors, which is a strength here. In any case, the process does not compare the usage of each atom, only the quality of the reconstruction.

The sparsity controlling parameter α on both the coder's LASSO and the d_1 error is nonzero, in order to encourage adding (11) to the dictionary even if (01) and (10) are already present, as that would bring an improvement by using only one atom. Manually computing the Manhattan error in the evaluation instead of Euclidian penalizes rebuilding both (01) and (10) as $(\frac{1}{2}\frac{1}{2})$.

A relatively high $\alpha = \frac{1}{\sqrt{k}}$ (capped at 0.5) is used by default, but can be changed. This helps convergence and emphasizes using as few words as possible to get closed itemsets, instead of focusing on improving the rebuilding of frequent combinations.

In case of a tie, the first atom is selected. This is a greedy algorithm, in that it makes the locally optimal choice at each iteration. The raised solution is optimal if the optimization problem is the maximization of a submodular function (diminishing returns when adding new elements). A set function f is submodular if:

$$\forall X, Y \subseteq \Omega \text{ with } X \subseteq Y \text{ and every } x \in \Omega \setminus Y, \text{ we have}$$
$$f(X \cup \{x\}) - f(X) \geq f(Y \cup \{x\}) - f(Y)$$

Then the use of a greedy algorithm to maximize f will produce a solution within a factor of $1 - 1/e$, which is the best possible approximation (Feige, 1998).

Proposition 2. *The problem of finding* $S^* = \arg\max_S f(S)$ *admits a good submodular approximation.*

Proof sketch. Consider the gain made by adding λ so a set S, and consider S' a set of atoms such that $S \subseteq S'$. Assuming the sparse encoder optimizes correctly, the encoding cannot be made worse by adding more atoms to choose from in the dictionary: if useless, they will simply be ignored. So $f(S) \leq f(S')$, and f is monotonous.

Furthermore, adding redundant atoms does not improve the reconstruction further, resulting in diminishing returns: for each row (*i.e.* word) $x \in X$ to be rebuilt, the solution found is a linear combination of the atoms in the dictionary. Adding a new atom λ to this dictionary will only improve the reconstruction depending on the coefficient given to the new atom, which is turn depends on how much error the previous dictionary made on this word x that the new atom was brought to correct. Indeed, the improvement that a candidate λ can bring is bounded by the difference in the projections of x on the subspaces defined by V_t and by $V_t \cup \lambda$ (Krause and Cevher, 2010). This is even more true due to α which penalizes the use of too many atoms to rebuild x. □

2.3 Implementation and availability of the method

As an update of *OLOGRAM*, the code is written in Python 3, with some performance-critical tasks in Cython and C++. To preserve RAM, the shuffles are divided into batches. Optimizations result in major speedups. This allows working on larger cases, totalling hundreds of thousands of regions with enough patience. However in such cases the RAM cost will be high and scale with the number of shuffles, since OLOGRAM remember all intersections to compute the statistics. To alleviate this, separate runs can be merged by treating each as a superbatch. The demonstration example takes about an hour on an i7-7820HQ.

For the MODL subroutines of dictionary learning and sparse coding, the *Scikit-Learn* implementation is used (Pedregosa et al., 2011). OLOGRAM-MODL is accessible through the command line interface of *pygtftk* (Lopez et al., 2019) which is available on Bioconda, and at `https://github.com/dputhier/pygtftk/` along with the documentation containing more information on the approach. The integration with the *pygtftk* suite of tools allows easy use in bioinformatics pipelines, and easier extension.

The tool will output one set of statistics per combination of sets of interest. An *ologram_modl_treeify* plugin creates visual representations of the results of a multiple overlap analysis, used to generate figure 3. The resulting tsv file can be manually filtered before producing the representation. The MODL algorithm can also be used as a standalone combination mining algorithm through the API by importing the *pygtftk.stats.intersect.dict_learning.Modl* class.

2.4 Data

Three different types of data are used in this study, using both artificial data with known ground truth and real biological data. Full data is available in Supplementary Material repository `https://github.com/qferre/ologram-modl_supp_mat`.

1. **Noisy matrices**: an artificial overlap matrix whose unique rows are representation of $A+B$, $A+B+C+D$ or $E+F$. This equates to row vectors of respectively $(110000), (111100), (000011)$. A NOT is then applied to each element (turning a 0 into 1, and 1 into 0) with a probability p_N to represent noise.

2. **Artificial BEDs of regions**: a query set of 1,000 artificial genomic regions of length 200,000 has been generated and compares against (a) a third of the query, (b) a copy of said third, (c) a different third of the query that does not overlap with the first, and (d) a negative control of other random peaks.

3. **Real data**: selected binding regions from ReMap 2018 data (Chèneby et al., 2018) for the transcription factors FOXA1, BRD4, EP300, ESR1, GATA3, JUN, MAX, MED1 and *MYC* in the *hg38* human genome assembly for the *MCF7* breast cancer cell line.

3 RESULTS

The complete workflow, with Supplementary Data, is available as a Snakemake and can be used as a starting point for a different analysis. It is available at `https://github.com/qferre/ologram-modl_supp_mat`.

The first goal of the experiments below is to validate the contribution of this paper on both enrichment evaluation and itemset mining, using artificial data for which the ground truth is known and the results can be compared to. The second important issue is to ensure that the algorithm is not only able to deal with true biological data but also is a fair way to discover relevant complex of genomics regions and get novel insights.

3.1 Artificial data

3.1.1 OLOGRAM itself

The BEDs of artificial regions are used in this section, with an inexact counting. OLOGRAM correctly identifies the associations between sets: as a general rule, sets that have strong overlap with each other are seen as enriched, and vice-versa. Notably, the query was found enriched with its subsets but not with the negative control, and the combination of non-overlapping thirds is seen as depleted. Detailed results are presented in Suppl. Fig. 1.

As longer combinations will be more enriched for statistical reasons, enrichment should be compared between combinations of similar order. Similarly, if a set C is depleted with the query A but always present with B, and the combination $A+B$ is enriched, the combination $A+B+C$ will be enriched: what matters is what is the difference in enrichment from adding C. For the studied combinations, the S statistic (number of overlapping base pairs) indeed follows a Negative Binomial distribution, which confirms the assumptions of the statistical model. Full histograms are presented in Suppl. Fig. 2.

3.1.2 MODL and comparison with apriori

MODL is compared to the *apriori* algorithm using the artificial overlap matrices. *apriori* remains a reference in itemset mining and it still being worked on today (Raj et al., 2020). Both algorithms' usefulness in identifying the underlying combinations used when generating the data (see section 2.4) is compared. The criterion is the ranking given to each correct combination. Ranking in MODL is determined by the step at which the combination was selected, and the ranking in apriori is determined through ordering all found rules by support.

The top three combinations found by MODL are indeed the three complexes defined when generating the data: $A+B$, $A+B+C+D$ and $E+F$. However, their ranking is lower for apriori, because if the rule $A+B+C$ is true the rules $A+B$ and $A+B+C$ are equally true. While this particular pitfall can be avoided by using a closed itemset miner eliminating redundant rules, this is vulnerable to noise and approximate itemset miners, such as MODL purports to do, are proposed as a solution (Chen et al., 2009). The two approaches have different goals, with MODL mining for complexes and apriori for association rules. In biology, the first goal would be preferred to identify full regulatory complexes. These findings hold with and without using noise in the matrices. With 12% noise, MODL still returns the correct combinations. However, smothering increases the sensibility to noise by emphasizing rarer combinations: its use is a compromise between denoising and not ignoring the rarest combinations. Detailed results are presented in Suppl. Fig 3.

Furthermore, when compared to apriori and by extension other itemset miners, MODL has a bias towards the most abundant combinations in the data, instead of those with highest support. It also tends to return longer combinations, some of which are too broad potential correlation groups (cf. section 2.2.3) such as (110011) here. This loss of granularity is a known necessary drawback of Approximate Itemset Mining approaches (Chen et al., 2009). As MODL is designed to mine for complexes, the candidates that were not selected will often represent over-fitted combinations that may be due to noise patterns, instead of representing shorter association rules. The normalization of the atoms by their squared sum helps correct this problem (cf. section 2.2.4).

MODL admittedly has a high computation cost due to the large number of factorizations performed: it scales in $O(k)$ with the number of sets and $O(q^2)$ in the number of queried combinations (atoms), but in most use cases the time cost remains reasonable, a few minutes at most. The abundance threshold τ (cf. section 2.2.2) helps reducing the time cost. The use of τ, along with the number of desired combinations q, is analogous to the minimum support used in apriori and other itemset miners to limit the exponential

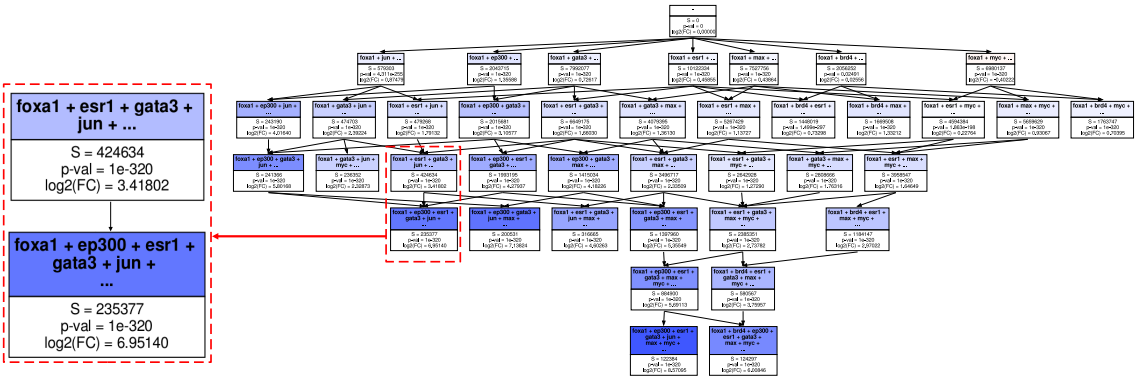


Figure 3. Example of the structure of the output graph that pictures combination enrichment for FOXA1 in MCF7, with a zoom on a relevant part. For each combination, the number of base pairs over which it is found in real data (S) is presented, followed by the $\log_2$ of the fold change and corresponding p-value according to a Negative Binomial model. Color gradient is based on the fold change. The combinations displayed here were manually selected, and the full tree of all combinations and the selection by MODL are available in Supplementary Data.

complexity of the problem. Details of MODL scaling are presented in Suppl. Fig. 4.

When comparing the scaling of the elementary operation of MODL (one instance of dictionary learning) with apriori, dictionary learning scales linearily with the number of sets and the number of queried atoms, while apriori scales exponentially with the number of sets and as such exponentially with lower minimum support. Time costs are of the same order of magnitude. These insights can be extended to other algorithms such as FP-Growth and ECLAT which have similar time costs (Garg and Kumar, 2013). Details in Suppl. Fig. 5.

3.2 Transcription Factors combinations in MCF7

The combinations of Transcriptional Regulators (TRs) associated with the transcriptional activator FOXA1 are studied here in the MCF7 breast cancer cell line. FOXA1 is a transcriptional activator, known to interact with chromatin as a pioneer factor. In many types of breast cancer, such as MCF7, it is also known to act as a pioneer factor to the regulator ERα (aka. ESR1, Ross-Innes et al., 2012). Conversely, it is a downstram target of the regulator GATA3 in breast cells (Kouros-Mehr et al., 2006). As the regions considered (TR binding sites) cover a small proportion of the genome, to ensure the longer combinations still have a statistically reasonable chance of occurring under (H_0) the shuffling is restricted to a subgenome of interest. This subgenome is made of estimated pseudo-Cis-Regulatory-Modules, defined as the merged regions for all considered TRs.

A manual illustrative selection of combinations is presented in Figure 3 as a directed acyclic graph. The expected correlators of ESR1 and GATA3 are indeed found enriched, confirming the relevance of the approach. MAX and MYC are also found more enriched together than separately, as could be expected since they are known to be associated (Laskowski and Knoepfler, 2013). The graph representation highlights the importance of EP300, and JUN to a lesser extent: ESR and GATA3 without either only have moderate enrichment, and same-length combinations containing ESR1 and GATA3 but without either EP300 or JUN have lower fold changes. This suggests that they are all an important part of a FOXA1 regulatory complex. Indeed, when looking at the total number of basepairs (S), FOXA1 + EP300 + ESR1 + GATA3 covers 2M base pairs out of the 6M of FOXA1 + ESR1 + GATA3. Conversely, this means that of all the 2M base pairs on which EP300 intersects with GATA3, almost all are with ESR1 and GATA3. Furthermore, FOXA1 + ESR1 + GATA3 + MAX covers 3.5M base pairs out of the 4M of FOXA1+GATA3+MAX. The full tree is presented in Suppl. Fig 6.

MODL (with $k = 8$ and $q = 20$) selected relevant shorter combinations before resorting to noisy patterns, although they are not the most enriched. The full selection is presented Suppl. Fig. 7. EP300 and JUN are less prominent in the selection as they are rarer than other TRs, but indeed found as part of their regulatory complexes (large proportion of their total S). Conversely, the frequent MAX and MYC are more represented. It tends to select closed itemsets (for instance, EP300 was not selected alone) but

this is admittedly not always true. The selection shown in Figure 3 was based in half on the selection by MODL, with additions selected to illustrate our point.

This highlights the regulatory complexes ESR1 and others regulators are a part of. Surprisingly, rather than saying ESR1 is associated to FOXA1, it would be more correct to say it is part of a regulatory complex to which FOXA1 also belongs. EP300 appears necessary in some cases. Considering only pairwise overlaps would have blinded us to that fact. This suggests that FOXA1 is associated to enhancer regions activated by histone acetylation, from the presence of EP300 (Ogryzko et al., 1996) and BRD4 (Lee et al., 2017). High activity of those enhancers is further confirmed by the presence of known transcriptional activators, such as MYC and MAX and especially ESR1. This deserves further study as such associations are, to our knowledge, not explored in the literature. This illustrates how OLOGRAM-MODL can be applied meaningfully to certain biological problems.

4 DISCUSSION

The *OLOGRAM-MODL* approach consists of two steps. First, mining for itemsets of open regions in the matrix of true overlaps. This is optional, and used to reduce the number of combinations to be studied. The second step is to compute the enrichment of all relevant combinations with the OLOGRAM approach, by determining whether this combination occurs in the real data across more base pairs that would be expected by chance.

By using inexact combinations, it is possible to emphasize necessary elements in regulatory clusters. This is shown in Results (section 3) and emphasized by the graphical representation, showing which additions to any combination actually increase its enrichment. Necessary regulators introduced by child combinations will account for most of the $S(\gamma)$ of the parents (downward closure). The algorithm is expected to be useful in studying Cis-Regulatory Elements as n-wise clusters of regulators, and moving away from only considering pairwise associations.

However, with a large number of sets ($k \geq 5$), or with regions covering a small proportion of the genome such as Transcriptional Regulator binding sites, longer combinations will have small expected overlaps. It is recommend to restrict the shuffling to a smaller sub-genome of interest, for example only to enhancers or promoters, or to the merged regions of all candidate sets (maybe identified by running a first pairwise analysis). Even so, combinations with more sets are often more enriched since all sets are supposed independent. Relatedly, it is recommended to ignore combinations found on such a small number of basepairs that they are unlikely to be biologically significant, even if highly enriched.

The MODL itemset mining algorithm can be used to focus on elementary combinations of interest. It leverages matrix factorization techniques for their robustness to noise, which is a widespread problem in biological assays (cf. Introduction). This entails learning compromise combinations, but this is not always desirable and the queried number of sets should be kept close to the actual expected number. Note that no matter which combinations are identified by MODL, the enrichment results do not change. While MODL is time consuming due to performing numerous factorizations, its time cost remains reasonable in most use cases. MODL can be used to pre-select combinations, using its results as a starting point to filter combinations in a larger analysis based on the biological problematic (eg. all combinations containing the particular regulator that you are studying, or the most frequent ones).

The integration with the *pygtftk* toolkit facilitates complex queries in bioinformatic worfklows. As a command line tool whose dependencies can be handled by *conda*, it is convenient to run on clusters with reproducible workflow managers such as Snakemake. This is also true of the plugins that produce the graphical representations. Thanks to the use of a Monte Carlo approach and to C++ optimization, OLOGRAM-MODL can even be run on a laptop, as demonstrated.

4.1 Perspectives

OLOGRAM-MODL is applicable to any problem that can be reduced to quantifying the significance of overlaps between n sets of position intervals. Besides epigenetic marks binding sites, associations between sets of regions such as "promoters of housekeeping genes" or "Binding sites for the regulator X in the experimental condition Y" can also be integrated. As such, this approach can also be extended to multi-omics problems. Since the overlaps are considered in terms of S (overlapping base pairs), it is also possible to do a proximity analysis by extending the regions by different values and comparing the significance of enrichment of each (*ologram_merge_stats*). This is necessary when working on

small TF motifs instead of ChIP-Seq. Our approach would be useful to find common occupancies of chromatin-binding proteins (Partridge et al., 2020).

The MODL algorithm can be applied to any submodular problem, as the API supports custom error functions, and customization of parameters such as smothering and α. For example, since variable selection in Naive Bayes classifiers is indeed submodular (Wei et al., 2015), MODL could select combinations that help such a classifier predict active enhancers. MODL is a first contribution open to further research. In order to solve its poor scaling with the number of queried atoms, it would be possible in step 2 to not test all candidates but only the children of the already present nodes assuming that if an atom A is better than an atom B, then its best parent is better than the best parent of B, proceeding until an optimum is reached.

It would be interesting to extend OLOGRAM-MODL to intra-set overlaps, which could be used to model a signal through quasi-Lebesgue integration by converting it into blocks of reads into overlapping regions. Concatenating flags on successive lines could be a way to include temporality. The implementation includes notes to facilitate such improvements, and others such as integrating custom shuffles for the user, or remembering regions IDs when shuffling. Indeed, as for *pygtftk* itself, OLOGRAM-MODL was designed to be evolutive and collaborative.

5 CONCLUSION

The major contribution of this work is the design and implementation of an algorithm that both mines and evaluates the enrichment for combinations of more than two sets of genomic regions, which is relevant because genomic regulators tend to work as complexes. *OLOGRAM-MODL* was designed to leverage itemset mining together with a statistical model analysis, and get the strengths of both. A novel optional itemset mining algorithm designed to denoise and mine for clusters, not simply association rules, is proposed. Then, a statistical framework evaluates the enrichment of the combinations using a Negative Binomial model, which is more resolutive than empirical p-values. It returns a parsable graphical representation which helps the identification of master regulators, by supporting inexact combinations. Finally, the approach is validated on artificial data, and shown to be useful in identifying previously neglected regulators associated to FOXA1. This method is implemented as an easy-to-use tool for the scientific community in the *pygtftk* suite, which makes it easy to use in bioinformatic pipelines.

ACKNOWLEDGEMENTS

The authors wish to thank Guillaume Charbonnier, Marina Kreme and Nori Sadouni for helpful discussion, beta-testing and feedback.

FUNDING

Q.F, C.C and D.P. were supported by recurrent funding from INSERM and Aix Marseille Univ.

REFERENCES

Agrawal, R. and Srikant, R. (1994). Fast Algorithms for Mining Association Rules in Large Databases. In *Proceedings of the 20th International Conference on Very Large Data Bases*, VLDB '94, pages 487–499, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Aszódi, A. (2012). MULTOVL: fast multiple overlaps of genomic regions. *Bioinformatics*, 28(24):3318–3319.

Biggar, S. R. and Crabtree, G. R. (2001). Cell signaling can direct either binary or graded transcriptional responses. 20(12):3167–3176.

Bryner, D., Criscione, S., Leith, A., Huynh, Q., Huffer, F., and Neretti, N. (2017). GINOM: A statistical framework for assessing interval overlap of multiple genomic features. *PLOS Computational Biology*, 13(6):e1005586.

Chee, C.-H., Jaafar, J., Aziz, I. A., Hasan, M. H., and Yeoh, W. (2019). Algorithms for frequent itemset mining: a literature review. *Artificial Intelligence Review*, 52(4):2603–2621.

Chen, J., Zhou, B., Wang, X., Ding, Y., and Chen, L. (2009). Mining noise-tolerant frequent closed itemsets in very large database. 92:1523–1533.

Chèneby, J., Gheorghe, M., Artufel, M., Mathelier, A., and Ballester, B. (2018). ReMap 2018: an updated atlas of regulatory regions from an integrative analysis of DNA-binding ChIP-seq experiments. *Nucleic Acids Research*, 46(D1):D267–D275.

Efron, B., Hastie, T., Johnstone, I., and Tibshirani, R. (2004). Least angle regression. *Annals of Statistics*, 32(2):407–499.

Ernst, J. and Kellis, M. (2012). ChromHMM: automating chromatin state discovery and characterization. *Nature methods*, 9(3):215–216.

Feige, U. (1998). A Threshold of ln n for Approximating Set Cover. *Journal of the Acm*, 45:314–318.

Ferré, Q., Charbonnier, G., Sadouni, N., Lopez, F., Kermezli, Y., Spicuglia, S., Capponi, C., Ghattas, B., and Puthier, D. (2020). OLOGRAM: determining significance of total overlap length between genomic regions sets. *Bioinformatics*, 36(6):1920–1922.

Garg, K. and Kumar, D. (2013). Comparing the Performance of Frequent Pattern Mining Algorithms. *International Journal of Computer Applications*, 69(25):21–28.

Giannopoulou, E. G. and Elemento, O. (2013). Inferring chromatin-bound protein complexes from genome-wide binding assays. *Genome Research*, 23(8):1295–1306.

Kanduri, C., Bock, C., Gundersen, S., Hovig, E., and Sandve, G. K. (2019). Colocalization analyses of genomic elements: approaches, recommendations and challenges. 35(9):1615–1624.

Koh, P. W., Pierson, E., and Kundaje, A. (2017). Denoising genome-wide histone chip-seq with convolutional neural networks. *Bioinformatics*, 33(14):i225.

Kouros-Mehr, H., Slorach, E. M., Sternlicht, M. D., and Werb, Z. (2006). GATA-3 Maintains the Differentiation of the Luminal Cell Fate in the Mammary Gland. *Cell*, 127(5):1041–1055.

Krause, A. and Cevher, V. (2010). Submodular dictionary selection for sparse representation. In *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ICML'10, pages 567–574, Madison, WI, USA. Omnipress.

Lambert, S. A., Jolma, A., Campitelli, L. F., Das, P. K., Yin, Y., Albu, M., Chen, X., Taipale, J., Hughes, T. R., and Weirauch, M. T. (2018). The Human Transcription Factors. *Cell*, 172(4):650–665.

Laskowski, A. I. and Knoepfler, P. S. (2013). Myc binds the pluripotency factor Utf1 through the basic-helix-loop-helix leucine zipper domain. *Biochemical and Biophysical Research Communications*, 435(4):551–556.

Lee, J.-E., Park, Y.-K., Park, S., Jang, Y., Waring, N., Dey, A., Ozato, K., Lai, B., Peng, W., and Ge, K. (2017). Brd4 binds to active enhancers to control cell identity gene induction in adipogenesis and myogenesis. *Nature Communications*, 8(1):2217.

Lopez, F., Charbonnier, G., Kermezli, Y., Belhocine, M., Ferré, Q., Zweig, N., Aribi, M., Gonzalez, A., Spicuglia, S., and Puthier, D. (2019). Explore, edit and leverage genomic annotations using python GTF toolkit. *Bioinformatics*.

Luna, J. M., Fournier-Viger, P., and Ventura, S. (2019). Frequent itemset mining: A 25 years review. *WIREs Data Mining and Knowledge Discovery*, 9(6):e1329.

Mairal, J., Bach, F., Ponce, J., and Sapiro, G. (2009). Online dictionary learning for sparse coding. In *Proceedings of the 26th Annual International Conference on Machine Learning - ICML '09*, pages 1–8, Montreal, Quebec, Canada. ACM Press.

Mansha, S., Lam, H. T., Yin, H., Kamiran, F., and Ali, M. (2018). Layered convolutional dictionary learning for sparse coding itemsets. *World Wide Web*.

Marinov, G. K., Kundaje, A., Park, P. J., and Wold, B. J. (2014). Large-Scale Quality Analysis of Published ChIP-seq Data. *G3: Genes, Genomes, Genetics*, 4(2):209–223. tex.pmid: 24347632 tex.publisher: G3: Genes, Genomes, Genetics.

Meckbach, C., Tacke, R., Hua, X., Waack, S., Wingender, E., and Gültas, M. (2015). PC-TraFF: identification of potentially collaborating transcription factors using pointwise mutual information. *BMC Bioinformatics*, 16.

Merrett, T. H. (1983). Why sort-merge gives the best implementation of the natural join. *SIGMOD Rec.*, 13(2):39–51.

Ogryzko, V. V., Schiltz, R. L., Russanova, V., Howard, B. H., and Nakatani, Y. (1996). The Transcriptional Coactivators p300 and CBP Are Histone Acetyltransferases. *Cell*, 87(5):953–959.

Partridge, E. C., Chhetri, S. B., Prokop, J. W., Ramaker, R. C., Jansen, C. S., Goh, S.-T., Mackiewicz, M., Newberry, K. M., Brandsmeier, L. A., Meadows, S. K., Messer, C. L., Hardigan, A. A., Coppola, C. J., Dean, E. C., Jiang, S., Savic, D., Mortazavi, A., Wold, B. J., Myers, R. M., and Mendenhall, E. M.

(2020). Occupancy maps of 208 chromatin-associated proteins in one human cell type. 583(7818):720–728.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Quinlan, A. R. and Hall, I. M. (2010). BEDTools: a flexible suite of utilities for comparing genomic features. *Bioinformatics*, 26(6):841–842.

Raj, S., Ramesh, D., Sreenu, M., and Sethi, K. K. (2020). EAFIM: efficient apriori-based frequent itemset mining algorithm on Spark for big transactional data. *Knowledge and Information Systems*, 62(9):3565–3583.

Ross-Innes, C. S., Stark, R., Teschendorff, A. E., Holmes, K. A., Ali, H. R., Dunning, M. J., Brown, G. D., Gojis, O., Ellis, I. O., Green, A. R., Ali, S., Chin, S.-F., Palmieri, C., Caldas, C., and Carroll, J. S. (2012). Differential oestrogen receptor binding is associated with clinical outcome in breast cancer. *Nature*, 481(7381):389–393.

Sandve, G. K., Gundersen, S., Rydbeck, H., Glad, I. K., Holden, L., Holden, M., Liestøl, K., Clancy, T., Ferkingstad, E., Johansen, M., Nygaard, V., Tøstesen, E., Frigessi, A., and Hovig, E. (2010). The Genomic HyperBrowser: inferential genomics at the sequence level. *Genome Biology*, 11(12):R121.

Shamos, M. I. and Hoey, D. (1976). Geometric intersection problems. In *17th Annual Symposium on Foundations of Computer Science (sfcs 1976)*, pages 208–215.

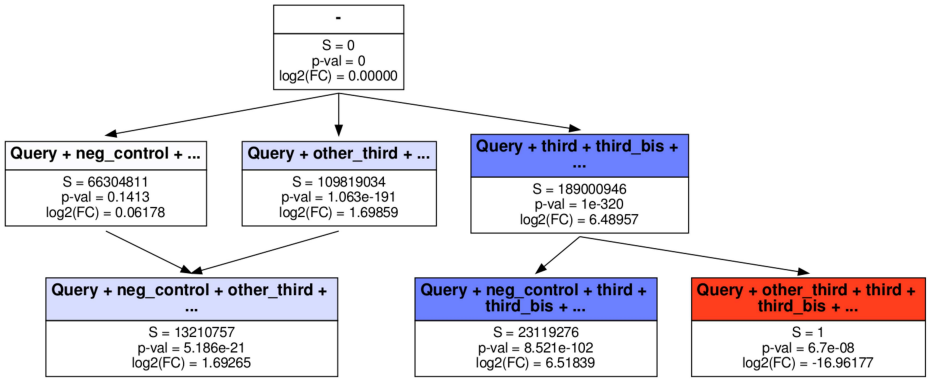
Simovski, B., Kanduri, C., Gundersen, S., Titov, D., Domanska, D., Bock, C., Bossini-Castillo, L., Chikina, M., Favorov, A., Layer, R. M., Mironov, A. A., Quinlan, A. R., Sheffield, N. C., Trynka, G., and Sandve, G. K. (2018). Coloc-stats: a unified web interface to perform colocalization analysis of genomic features. *Nucleic Acids Research*, 46(W1):W186–W193.

Teng, L., He, B., Gao, P., Gao, L., and Tan, K. (2014). Discover context-specific combinatorial transcription factor interactions by integrating diverse ChIP-Seq data sets. *Nucleic Acids Research*, 42(4):e24–e24.

Vandel, J., Cassan, O., Lebre, S., Lecellier, C.-H., and Brehelin, L. (2018). Probing transcription factor combinatorics in different promoter classes and in enhancers. *bioRxiv*, page 197418.

Wei, K., Iyer, R., and Bilmes, J. (2015). Submodularity in Data Subset Selection and Active Learning. In *International Conference on Machine Learning*, pages 1954–1963. PMLR.

A

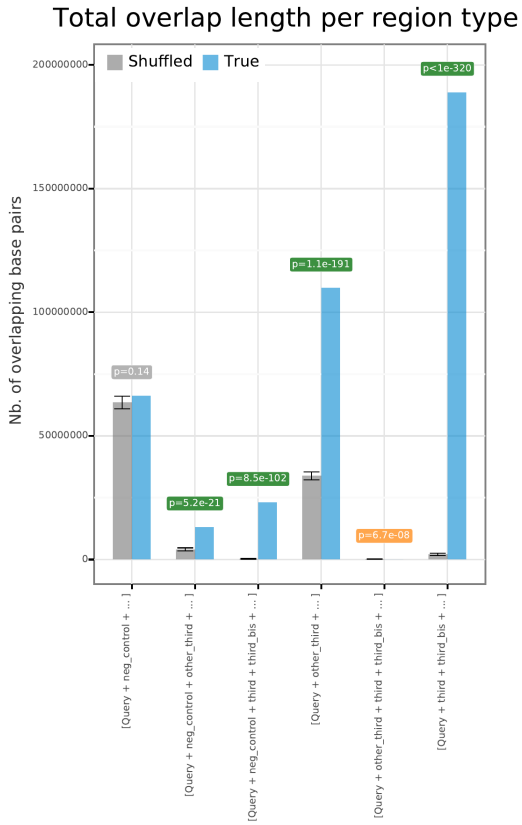


Supplementary Figure 1A: OLOGRAM run on multiple artificial data sets. This figure presents a graph-based representation giving the S statistic (number of overlapping base pairs), corresponding Negative Binomial p-value and fold change for each combinations.

The query was a set of 1,000 artificial regions of length 200,000 in hg38, compared against :
- “*third*”: a third of the aforementioned set
- “*third_bis*”: a copy of “*third*”
- “*other_third*” : a different third of the data, which does not overlap with “*third*” except on one base pair.
- “*neg_control*”: a negative control of random peaks of the same size as the original set.

OLOGRAM correctly retrieves the associations between sets. The query is shown to be enriched with its subsets, as presumably 100% overlap is more than one would expect by chance. Similarly, the first subset ("third") significantly overlaps its copy "third_bis". The two different subsets (“third” and “other_third”) never overlap in the true data, which constitutes a depletion and is marked as such. The negative control, made of random peaks, indeed not overlap the query more than randomly expected.

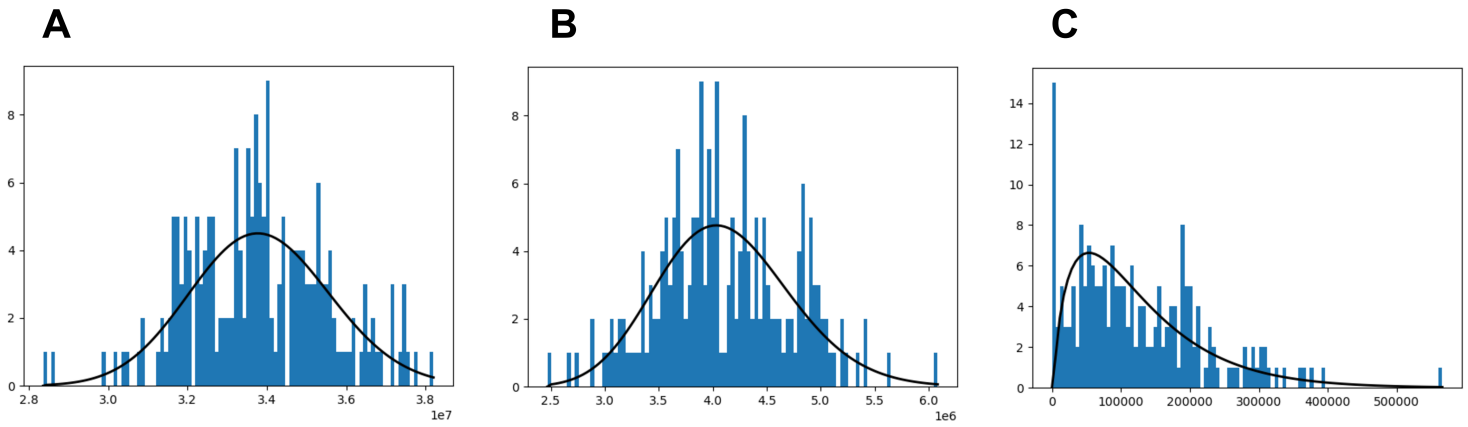
B



Supplementary Figure 1B : Alternative representation of the results in Suppl. Fig. 1A. The histograms give the number of overlapping base pairs (S) for each combination in the true data (blue) and expected, and in the shuffles (grey). Error bars indicate standard deviation. The p-value is indicated in the labels.

The command line used to run this example was:
`gtftk ologram -z -c hg38 -p query.bed --more-bed {params.peaks} --more-bed-multiple-overlap`
where {params.peaks} is a whitespace separated list of BED files.

The combinations are inexact, meaning that an overlap of [Query + neg_control + third] will still count as [Query + neg_control + …].



Supplementary Figure 2 : Distribution of the number of overlapping base pairs (S) in the shufffles, for artificial data of Supplementary Figure 1. Shuffles are made under the null hypothesis that all regions are independant. Negative binomial distributions of same esperance and variance as the data are overlaid. We used 160 shuffles.

A is [Query + other_third + …]
B is [Query + other_third + neg_control + …]
C is [Query + other_third + third + third_bis + …]

We can see that even with multiple overlaps, the S statistic still follow Negative Binomial distributions. Longer combinations (such as C here) will be more rarely observed. This supports fitting a Negative Binomial model, as otherwise determining the significance of larger fold changes would require many shuffles.

MODL

Combination						Status (with score at step 1 for the candidates)
A	B	C	D	E	F	
1	**1**	**1**	**1**			Selected at step 1 (-759)
				1	**1**	Selected at step 2 (-799)
1	**1**					Selected at step 3 (-761)
1	1		1			Candidate (-765)
1	1				1	Candidate (-805)
1	1	1	1		1	Candidate (-809)
1	1	1	1	1		Candidate (-810)
1	1			1	1	Candidate (-827)
1		1		1	1	Candidate (-839)

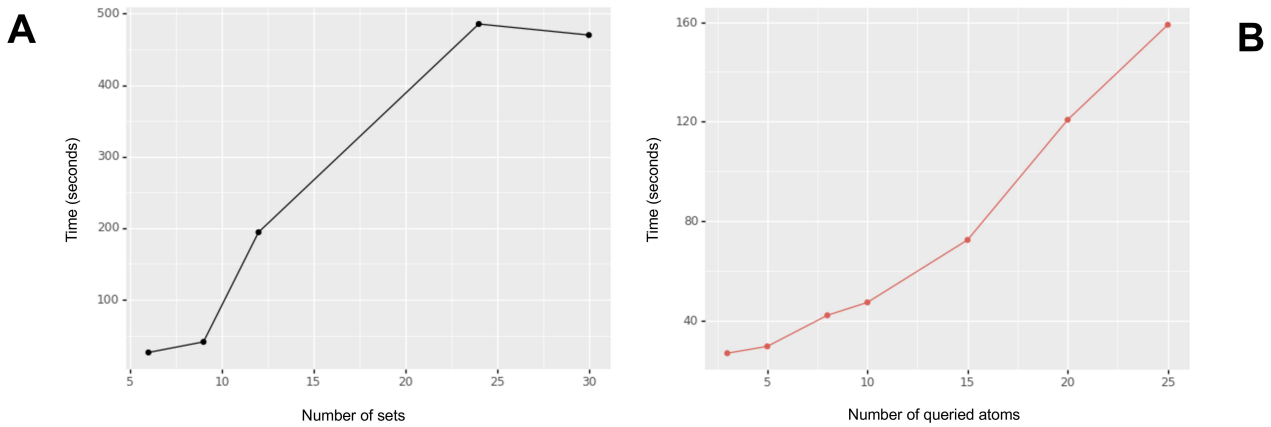
Apriori

Combination						Rank	Support
A	B	C	D	E	F		
1	**1**					1	0.5447
	1	1				2	0.3036
1		1				3	0.2975
1			1			4	0.2942
				1	**1**	5	0.2740
1	1	1				6	**0.2739**
1	1		1			7	**0.2685**
...						...	...
1	**1**	**1**	**1**			11	0.2160

Supplementary Figure 3 : Comparison of itemsets found by MODL and apriori when run on an artificial matrix. The itemsets used when generating the matrix are represented in green. Uniform noise of 0.12 was also applied.

For MODL, we queried 3 atoms, as this corresponds to the number of complexes used when generating the data. The first three atoms returned by MODL correspond to the itemsets defined when generating the data, with the remaining candidates corresponding to noise patterns or compromises.

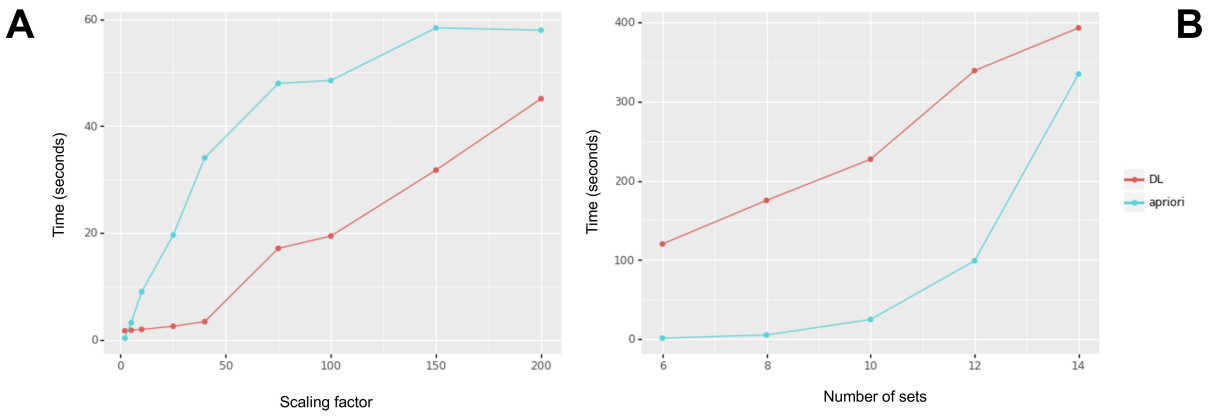
When ordering by support in apriori however, the rank of the true itemsets is much lower.



Supplementary Figure 4 : Time benchmark of the MODL algorithm. Time cost in seconds. Done on a matrix with 20,000 rows, 50% added noise with the same groups as the usual artificial matrices (AB, ABCD, EF).

(A) Scaling with the number of sets (columns) in the matrix. 8 queried atoms.
(B) Scaling with the final number of queried atoms. 8 sets in the matrix.

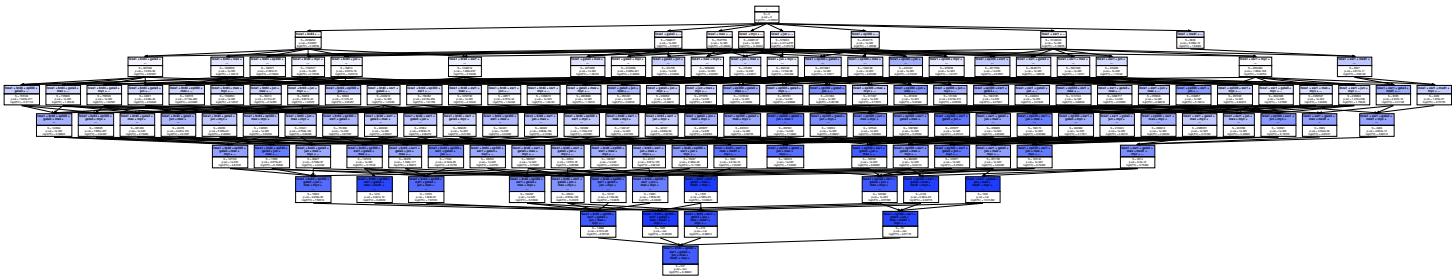
MODL scales mostly linearly with the number of sets, but semi-quadratically with the number of queried atoms. Late convergence due to a high alpha is the bulk of that cost for larger matrices. Total time remains on the order of minutes in most cases. Times given on an i7-7820HQ with 8 threads. The scaling with the number of lines, not shown here, is linear. MODL time cost stops increasing when the number of sets is so high that 2**k is much greater than the number of queried atoms, but results are unusable in such cases.



Supplementary Figure 5 : Scaling of dictionary learning (ie. DL, elementary operation of MODL) vs apriori. Single thread. Time given in seconds on an i7-7820HQ. Apriori here is a Python implementation, and DL is the Scikit-Learn Python implementation.

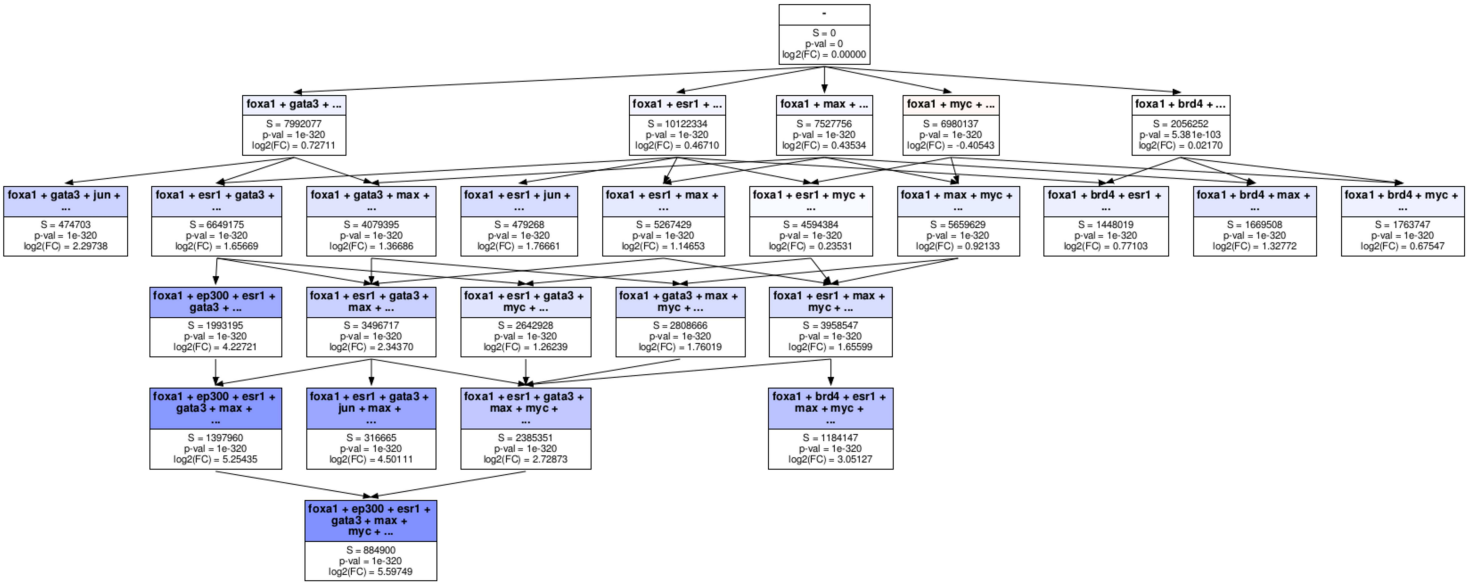
(A) Comparing the two main parameters. A scaling factor of F means the DL was instructed to learn 2*F atoms, and apriori had a minimum support of 1/F. The matrix used had 25,000 flags and 13 sets. Alpha for DL was 0.5. (B) Scaling with the number of sets (columns) and a matrix of 100,000 flags. Min support for apriori was 0.01, alpha for DL was 0.

In both cases, DL scales linearly and apriori scales exponentially. Parameters, such as DL's alpha, were chosen to have comparable running times, but a lower alpha makes DL slower and apriori scales exponentially when the minimum support decreases. DL cost increases faster once early convergence is no longer possible. If the number of sets k is so high that 2**k is much greater than the number of queried atoms, time cost will stop increasing. Apriori's time cost stop increasing once the minimum support is so low that all itemsets are found. For the number of lines, both apriori and DL scale linearily, not shown here. These insights can also be extended to others algorithms such as FP-growth which scales quadratically with k. Indeed, in most use cases, apriori's time cost is within the same order of magnitude as ECLAT and FP-growth.



Supplementary Figure 6 : Full tree representation of enriched combinations for FOXA1 in MCF7. No filtering, manual or MODL, was applied.

In the MCF7 breast cancer cell line, we study enrichment of combinations of transcription factors for FOXA1. We restrict the shuffling to pseudo-CRM made by merging of all query peaks, meaning all binding sites for all TF studied. We see the expected GATA3 and ESR1, but this representation highlights that adding JUN and EP300 increases enrichment, suggesting they are part of the complex.



Supplementary Figure 7 : Same tree as Suppl. Fig. 6 but restricted to combinations selected by MODL. MODL selects atoms that best help rebuild the matrix of original overlaps. Rarer but more enriched factors such as EP300, are found with their usual correlators and not alone.

5. Discussion

The central philosophy of the work presented in this thesis was based on the realization that genomic cis-regulation is effected by *combinations* of genomic cis-regulators. However, as we demonstrated throughout this manuscript, there was (and still remains) much to be done to leverage this fact to obtain useful insights into biological problems.

In this context, I sought to use those combinations as a *resource*. When sources (TRs, datasets) are correlating and forming a combination, this gives us an insight in the genomic function effected at the position where those source are found, since in biology colocalization is often a marker of functional association. It also increases the confidence that the observation of a regulator is correct, if its collaborators are also found at the same position, which is relevant for anomaly detection. The correct identification of those combinations is also an important itemset mining problem.

This is especially important now that we have entered the "big data" era. These two insights about combinations of regulators can also be extended to designate combinations of *datasets* to perform integrative analysis, using the multiplicity of datasets, and more generally data views, to strengthen the confidence of the observations. Indeed, the cyclopean amounts of data available vary in quality and may suffer from many different sources of error, which is compounded by the large amounts of data used. This is at the heart of the multi-testing corrections implemented in statistics. However, as the amounts of data continue to grow, supervised verification is still a tall order, as the experimental noise sources are very difficult to correct and annotated supervised data is rarely available. Even so, it would require a tedious error-by-error approach.

5.1. Summary of contributions

The contributions of this thesis are twofold. While my early work presented in chapter 2 dealt with supervised data, I eventually switched focus to the problem of leveraging combinations without supervision to help decide which sources of combinations thereof are relevant.

Unsupervised anomaly detection in genomic catalogues Using the combinations as a resource to perform curation is important and is indeed the gist of anomaly detection. Since the availability of many independent, and in some cases redundant, ChIP-Seq experiments is the entire impetus behind the ReMap project, this made collaboration a natural fit.

We showed in chapter 3 that combinations can be leveraged to curate databases of genomic region assays in an unsupervised manner. The general principle behind this anomaly detection approach is that confidence in an observation is strengthened when sources correlated with it are present. I used a specifically designed multi-view convolutional autoencoder to perform a "Goldilocks" compression. Here, the model is tasked to learn sources (TR, datasets) as part of a groups of correlating sources, and not alone. We identify peaks which have fewer known collaborators present in their vicinity than what would be average for their sources.

This exemplifies the need for cross-disciplinary collaboration. The project was born out of discussion with J. Chèneby who realized that her database of Cis-Regulatory Elements (ReMap) contained more than thrice as many inferred CREs as the human genome is estimated to contain. We further noticed that existing methods were rather rudimentary: for example the Irreducible Discover Rate used by ENCODE amounts to a simple pairwise comparison at the scale of an entire dataset. Due to the many potential sources of error, it became necessary to design an approach that would correct them indiscriminately.

Convolutional filters were used because their dedicated purpose is to learn combinations of elements. However, convolutional networks are usually designed for images. Bending them into working on a more general tensor representation was an interesting challenge. Furthermore, the notion of a "Goldilocks" compression is counter intuitive, since compression methods are usually designed to rebuild with maximum fidelity. Here, we wanted just enough fidelity to learn groups, but not the noise. This led to another challenge: the development of methods to permit interpretation the results and select the information budget of the model. In the end, I developed approaches to evaluate auto-encoders based on their respect of existing correlations. I also proposed a new normalization method based on correcting for the average cardinality of the aforementioned correlation groups. It can be applied to any black box model, and is useful to interpret autoencoders when performing anomaly detection.

Our cleaned data improves Cis-Regulatory Element detection. As a result, it is now possible to assign confidence to ChIP-seq datasets in an unsupervised manner. This data also helps identify true Cis Regulatory Elements, by focusing on the ones where regulator complexes are present and complete.

Combination mining and statistical enrichment When it comes to the combinations γ themselves, The enrichment of combinations of regulators (and more generally whether two or n sets of genomic regions intersect more than by chance) is one of the most fundamental problem in bioinformatics.

We use a Monte Carlo based method to fit a novel Negative Binomial model on the number of base pairs on which a given combinations of elements is observed. This returns p-values that are orders of magnitude more precise compared to existing approaches, and is even valid for multiple combinations (ie. order $n \geq 2$). It is my belief that a rigorous model for this problem was long overdue. Another benefit is that it is comparatively simple, and as such immediately understandable for biologists and

more generally for people without a deep mathematical background.

I tried to get the best of two different worlds for OLOGRAM-MODL. After proposing this Negative Binomial model, I asked myself how to extract the most meaningful combinations out of the possible 2^k, for k sets, so as not to overwhelm the user. Matrix factorization methods are more rarely used in itemset mining compared to tree-based methods (ie. apriori). I though this was a shame: since biological assays are known to present significant noise, these methods with their known resistance to noise became a natural fit, dictionary learning in particular. Finally, this approach was combined with a custom greedy algorithm to identify interesting combinations of regulators, based on which itemsets best rebuild the original data in a noise resistant manner, by proving that the combination selection problem is submodular.

5.1.1. What have the combinations ever done for us ?

To summarize, for this paradigm of **combinations as a resource**, we show they can be leveraged to perform anomaly detection in bioinformatical data (atyPeak) in chapter 3. We also show they can be mined with algorithms tailored to find complexes (MODL) and their enrichment precisely quantified (Monte Carlo) in chapter 4. In this, we tie back into anomaly detection and itemset mining.

Work impact Combinations of regulators are seldom studied, even in the recent literature. I believe providing an easy-to-use and easy-to-understand statistical model (OLOGRAM) will help rekindle that. Furthermore, the statistical models and algorithms proposed in my research can be reused on any data structured as a list of genomic intervals, perhaps even to study correlations in non-biological data. For atyPeak, I hope it can sensibilize people to the problem of database curation, and more specifically unsupervised curation, since the lack of curated references is a widespread problem in many databases. Again, this need not be restricted to biological data. Furthermore, the normalization methods I developed should be of interest to anyone using black box model to perform unsupervised anomaly detection, but also itemset mining.

In summary, the contributions of this thesis can be approached from different angles, depending on your domains of interest:

— For biologists, we provide data assorted of a new confidence metric (atyPeak). This permits identification of anomalous binding events [1] and facilitates the identification of Cis-Regulatory Elements of interest. This data can also be used to robustly identify the complexes of regulatory elements (OLOGRAM) since we harped on about how crucial they are.

— We raise new questions on the usage of alternative promoters for ATP2C1, and on the regulatory complexes of which FOXA1 is a part of.

1. As discussed, those may be errors but also rare real events or simply marks of missing data. It is impossible to tell which without supervision.

— For bioinformaticians, we provide a tool that can be used to clean databases of genomic elements, and perform a rigorous statistical study of the enrichment of combinations of regulators.
— For mathematicians and machine learning specialists, we provide new tools and algorithms to find itemsets and select them, as well as study their enrichment. We further provide tools for anomaly detection using autoencoders, and the interpretation of such models. Those insights can be extended beyond a biological context.

Perspectives and generalization During my thesis, my goal was to leverage such combinations through the use of machine learning methods, which are very effective at learning regularities in the data: in other words, learning combinations. I elected to represent the regions where regulators bind as lists of intervals, converted into matrix and tensor representations. As a result, the approaches I developed during my thesis are generalizable to any data represented as lists of intervals, as long as it matches the format given in section 1.4. Those approaches are contributions to the broader domains of anomaly detection and itemset mining, but also statistical modeling and DNN interpretation in general.

5.2. Methodological notes

I would like to conclude this manuscript on a more philosophical and, dare I say, epistemological note, by offering a perspective over what I learned as a scientist and a critique of certain current misconceptions.

It is my belief that even the most outwardly simple problems (for example, the statistical enrichment of combinations) can have very interesting research questions behind them. This idea of tackling problems that are ostensibly simple, but in reality far from solved, using algorithms tailored to the problems themselves is at the heart of my research philosophy. Indeed, during my PhD, I designed and taught a machine learning class to Masters students. My goal was to impart them this philosophy, by teaching them enough about the mathematical foundations of the methods so that they would understand why and how to apply them to biological problems. I would also recommend the use of simple models in front-end combined with, perhaps, a more complex back-end, for ease of interpretation.

Furthermore, the current *zeitgeist*[2] seems to be rooted in the consensus that Machine Learning methods will soon solve every problem forever. And that we will soon live in an utopia with self-driving robot butlers, cyborg kittens for everyone and gold-plated sapient toothbrushes. I would offer a different perspective. I do not believe Machine Learning to be a silver bullet: the models must be designed and calibrated for their specific applications, and while very interesting data analysis can be done, the models cannot, for example, magically generate clean data when there was none

2. Literally, "spirit of the times", in the same vein as Jung's "collective unconscious".

to be trained on in the first place. Indeed, the atyPeak project took me the better part of two years to perform what is on paper a fairly straight forwards anomaly detection task. During my thesis, I also worked on metrics and model evaluation to help tailor the model to the biological problematic (Q-score for atyPeak, custom loss for MODL). I believe it is important to use such scores to ensure the Machine Learning model remains close to the reality it is ostensibly modeling.

I also find that the field also suffers from inflated expectations. I cannot count the number of times somebody asked me out-of-the-blue how they could apply Deep Learning to their – yet unspecified – data. My answer was always the same: "what is the problematic you want to tackle?". When Deep Learning is warranted, like atyPeak, it was a fascinating dive. When it is not? Well, I believe approaches should only be as complex as they are need to be if they are to remain interpretable. None of this should be taken as a criticism of the field itself! I am an ardent supporter of the integration of Machine Learning in bioinformatics and beyond. I am simply weary, and wary, of the hype.

What I enjoyed the most during my thesis was the opportunity to bridge the gap across two disciplines which, at first glance, may seem to be alien to one another. I noticed that the fields of machine learning and applied statistics had many solutions to offer to biological problems, and working at the interface of those fields was an exciting opportunity to discover and apply new methods. Working on the *pygtftk* toolset led me to discover proper development practices.

This work helped keep me grounded, by constantly applying the developed models to real biological data, which pointed to several biological scenarios that required the model to be adjusted (such as non-exclusive correlation groups and large imbalances in dataset frequencies). I believe it is important to always keep in mind that reality is noisy and does not always follow our nice, simple models.

As parting words, I would like to say that I am very grateful for the confidence placed in me by my advisors during the completion of my thesis. They trusted me when I suggested avenues of research that I thought would be interesting, and allowed me to pursue them.

Bibliography

[Aba+16] Martin Abadi, Paul Barham, Jianmin Chen, et al. “TensorFlow: A system for large-scale machine learning”. In: *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)*. 2016, pp. 265–283. URL: https://www.usenix.org/system/files/conference/osdi16/osdi16-abadi.pdf (visited on 12/28/2020) (cit. on p. 60).

[AIS93] Rakesh Agrawal, Tomasz Imieliński, and Arun Swami. “Mining association rules between sets of items in large databases”. In: *Proceedings of the 1993 ACM SIGMOD international conference on Management of data*. SIGMOD ’93. New York, NY, USA: Association for Computing Machinery, June 1, 1993, pp. 207–216. ISBN: 978-0-89791-592-2. DOI: 10.1145/170035.170072. URL: https://doi.org/10.1145/170035.170072 (visited on 12/01/2020) (cit. on p. 61).

[Agr+96] Rakesh Agrawal, Heikki Mannila, Ramakrishnan Srikant, et al. “Fast discovery of association rules”. In: *Advances in knowledge discovery and data mining*. USA: American Association for Artificial Intelligence, Feb. 1, 1996, pp. 307–328. ISBN: 978-0-262-56097-9. (Visited on 12/17/2020) (cit. on p. 63).

[Air+11] Daniel Aird, Michael G. Ross, Wei-Sheng Chen, et al. “Analyzing and minimizing PCR amplification bias in Illumina sequencing libraries”. In: *Genome Biology* 12.2 (2011), R18. ISSN: 1474-760X. DOI: 10.1186/gb-2011-12-2-r18 (cit. on p. 47).

[ATK15] Leman Akoglu, Hanghang Tong, and Danai Koutra. “Graph based anomaly detection and description: a survey”. In: *Data Mining and Knowledge Discovery* 29.3 (May 1, 2015), pp. 626–688. ISSN: 1573-756X. DOI: 10.1007/s10618-014-0365-y. URL: https://doi.org/10.1007/s10618-014-0365-y (visited on 12/17/2020) (cit. on p. 60).

[Ala+15] T. Alasalmi, H. Koskimäki, J. Suutala, et al. “Classification Uncertainty of Multiple Imputed Data”. In: *2015 IEEE Symposium Series on Computational Intelligence*. 2015 IEEE Symposium Series on Computational Intelligence. Dec. 2015, pp. 151–158. DOI: 10.1109/SSCI.2015.32 (cit. on p. 97).

[Amb+20] Giovanna Ambrosini, Ilya Vorontsov, Dmitry Penzar, et al. “Insights gained from a comprehensive all-against-all transcription factor binding motif benchmarking study”. In: *Genome Biology* 21 (Dec. 1, 2020). DOI: 10.1186/s13059-020-01996-3 (cit. on p. 21).

[AKB19] Haley M. Amemiya, Anshul Kundaje, and Alan P. Boyle. “The ENCODE Blacklist: Identification of Problematic Regions of the Genome”. In: *Scientific Reports* 9.1 (June 27, 2019), p. 9354. ISSN: 2045-2322. DOI: 10.1038/s41598-019-45839-z. URL: https://www.nature.com/articles/s41598-019-45839-z (visited on 11/25/2020) (cit. on p. 45).

[Asz12] András Aszódi. “MULTOVL: fast multiple overlaps of genomic regions”. In: *Bioinformatics* 28.24 (Dec. 1, 2012), pp. 3318–3319. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/bts607. URL: https://academic.oup.com/bioinformatics/article/28/24/3318/245868 (visited on 11/03/2018) (cit. on p. 64).

[BKC16] Min Gyun Bae, Jeong Yeon Kim, and Jung Kyoon Choi. “Frequent hypermethylation of orphan CpG islands with enhancer activity in cancer”. In: *BMC Medical Genomics* 9.1 (Aug. 12, 2016), p. 38. ISSN: 1755-8794. DOI: 10.1186/s12920-016-0198-1. URL: https://doi.org/10.1186/s12920-016-0198-1 (visited on 11/11/2020) (cit. on p. 19).

[Bak16] Monya Baker. “1,500 scientists lift the lid on reproducibility”. In: *Nature* 533.7604 (2016), pp. 452–454. ISSN: 1476-4687. DOI: 10.1038/533452a (cit. on p. 50).

[BK11] Andrew J. Bannister and Tony Kouzarides. “Regulation of chromatin by histone modifications”. In: *Cell Research* 21.3 (Mar. 2011), pp. 381–395. ISSN: 1748-7838. DOI: 10.1038/cr.2011.22. URL: https://www.nature.com/articles/cr201122 (visited on 11/03/2020) (cit. on p. 17).

[Bar+13] Anaïs F. Bardet, Jonas Steinmann, Sangeeta Bafna, et al. “Identification of transcription factor binding sites from ChIP-seq data at high resolution”. en. In: *Bioinformatics* 29.21 (Nov. 2013), pp. 2705–2713. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/btt470. URL: https://academic.oup.com/bioinformatics/article/29/21/2705/195767 (visited on 09/18/2020) (cit. on p. 35).

[Bar+07] Artem Barski, Suresh Cuddapah, Kairong Cui, et al. “High-Resolution Profiling of Histone Methylations in the Human Genome”. In: *Cell* 129.4 (May 18, 2007), pp. 823–837. ISSN: 0092-8674, 1097-4172. DOI: 10.1016/j.cell.2007.05.009. URL: https://www.cell.com/cell/abstract/S0092-8674(07)00600-9 (visited on 11/11/2020) (cit. on p. 28).

[Bar+17] Anna Bartlett, Ronan C. O'Malley, Shao-shan Carol Huang, et al. "Mapping genome-wide transcription factor binding sites using DAP-seq". In: *Nature protocols* 12.8 (Aug. 2017), pp. 1659–1672. ISSN: 1754-2189. DOI: 10.1038/nprot.2017.055. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5576341/ (visited on 11/25/2020) (cit. on p. 47).

[BS14] Tuncay Baubec and Dirk Schübeler. "Genomic patterns and context specific interpretation of DNA methylation". In: *Current Opinion in Genetics & Development*. Genome architecture and expression 25 (Apr. 1, 2014), pp. 85–92. ISSN: 0959-437X. DOI: 10.1016/j.gde.2013.11.015. URL: http://www.sciencedirect.com/science/article/pii/S0959437X13001792 (visited on 11/04/2020) (cit. on p. 19).

[BP20] Jonathan A. Beagan and Jennifer E. Phillips-Cremins. "On the existence and functionality of topologically associating domains". en. In: *Nature Genetics* 52.1 (Jan. 2020), pp. 8–16. ISSN: 1546-1718. DOI: 10.1038/s41588-019-0561-1. URL: https://www.nature.com/articles/s41588-019-0561-1 (visited on 09/18/2020) (cit. on p. 26).

[BCV14] Yoshua Bengio, Aaron Courville, and Pascal Vincent. "Representation Learning: A Review and New Perspectives". In: *arXiv:1206.5538 [cs]* (Apr. 23, 2014). arXiv: 1206.5538. URL: http://arxiv.org/abs/1206.5538 (visited on 01/16/2021) (cit. on p. 89).

[Ben+18] Mary Lauren Benton, Sai Charan Talipineni, Dennis Kostka, et al. "Genome-wide Enhancer Maps Differ Significantly in Genomic Distribution, Evolution, and Function". In: *bioRxiv* (Apr. 30, 2018), p. 176610. DOI: 10.1101/176610. URL: https://www.biorxiv.org/content/10.1101/176610v3 (visited on 01/04/2021) (cit. on p. 67).

[Bes+18] J. Besser, H. A. Carleton, P. Gerner-Smidt, et al. "Next-generation sequencing technologies and their application to the study and control of bacterial infections". In: *Clinical Microbiology and Infection: The Official Publication of the European Society of Clinical Microbiology and Infectious Diseases* 24.4 (Apr. 2018), pp. 335–341. ISSN: 1469-0691. DOI: 10.1016/j.cmi.2017.10.013 (cit. on p. 32).

[Bi+21] Xuan Bi, Xiwei Tang, Yubai Yuan, et al. "Tensors in Statistics". In: *Annual Review of Statistics and Its Application* 8.1 (2021), null. DOI: 10.1146/annurev-statistics-042720-020816. URL: https://doi.org/10.1146/annurev-statistics-042720-020816 (visited on 12/01/2020) (cit. on p. 56).

[BC01] Stephen R. Biggar and Gerald R. Crabtree. "Cell signaling can direct either binary or graded transcriptional responses". In: *The EMBO Journal* 20.12 (June 15, 2001), pp. 3167–3176. ISSN: 0261-4189. DOI: 10.1093/emboj/20.12.3167. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC150188/ (visited on 10/13/2020) (cit. on p. 29).

[BH12] Clemens Bönisch and Sandra B. Hake. “Histone H2A variants in nucleosomes and chromatin: more or less stable?” In: *Nucleic Acids Research* 40.21 (Nov. 2012), pp. 10719–10741. ISSN: 0305-1048. DOI: 10.1093/nar/gks865. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3510494/ (visited on 11/04/2020) (cit. on p. 18).

[Bou+18] Guillaume Bourque, Kathleen H. Burns, Mary Gehring, et al. “Ten things you should know about transposable elements”. In: *Genome Biology* 19.1 (Nov. 19, 2018), p. 199. ISSN: 1474-760X. DOI: 10.1186/s13059-018-1577-z. URL: https://doi.org/10.1186/s13059-018-1577-z (visited on 11/11/2020) (cit. on p. 27).

[Bry+17] Darshan Bryner, Stephen Criscione, Andrew Leith, et al. “GINOM: A statistical framework for assessing interval overlap of multiple genomic features”. In: *PLOS Computational Biology* 13.6 (June 15, 2017), e1005586. ISSN: 1553-7358. DOI: 10.1371/journal.pcbi.1005586. URL: https://journals.plos.org/ploscompbiol/article?id=10.1371/journal.pcbi.1005586 (visited on 08/10/2020) (cit. on p. 64).

[Cao+16] Bokai Cao, Hucheng Zhou, Guoqiang Li, et al. “Multi-view Machines”. In: *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining*. WSDM ’16. New York, NY, USA: Association for Computing Machinery, Feb. 8, 2016, pp. 427–436. ISBN: 978-1-4503-3716-8. DOI: 10.1145/2835776.2835777. URL: https://doi.org/10.1145/2835776.2835777 (visited on 01/16/2021) (cit. on p. 43).

[Car+06] Piero Carninci, Albin Sandelin, Boris Lenhard, et al. “Genome-wide analysis of mammalian promoter architecture and evolution”. In: *Nature Genetics* 38.6 (June 2006), pp. 626–635. ISSN: 1061-4036. DOI: 10.1038/ng1789 (cit. on p. 72).

[Caw+04] Simon Cawley, Stefan Bekiranov, Huck H. Ng, et al. “Unbiased mapping of transcription factor binding sites along human chromosomes 21 and 22 points to widespread regulation of noncoding RNAs”. In: *Cell* 116.4 (Feb. 20, 2004), pp. 499–509. ISSN: 0092-8674. DOI: 10.1016/s0092-8674(04)00127-8 (cit. on p. 47).

[Cay+15] Christelle Cayrou, Benoit Ballester, Isabelle Peiffer, et al. “The chromatin environment shapes DNA replication origin organization and defines origin classes”. In: *Genome Research* 25.12 (Dec. 1, 2015), pp. 1873–1885. ISSN: 1088-9051, 1549-5469. DOI: 10.1101/gr.192799.115. URL: http://genome.cshlp.org/content/25/12/1873 (visited on 11/11/2020) (cit. on p. 26).

[CBK09] Varun Chandola, Arindam Banerjee, and Vipin Kumar. “Anomaly Detection: A Survey”. In: *ACM Comput. Surv.* 41.3 (July 2009), 15:1–15:58. ISSN: 0360-0300. DOI: 10.1145/1541880.1541882. URL: http://doi.acm.org/10.1145/1541880.1541882 (visited on 11/01/2019) (cit. on p. 60).

[Che+09] Junbo Chen, Bo Zhou, Xinyu Wang, et al. “Mining Noise-Tolerant Frequent Closed Itemsets in Very Large Database”. In: *IEICE Transactions on Information and Systems* 92 (2009), pp. 1523–1533. DOI: 10.1587/transinf.E92.D.1523. URL: http://adsabs.harvard.edu/abs/2009IEITI..92.1523C (visited on 10/13/2020) (cit. on p. 63).

[Chè+20] Jeanne Chèneby, Zacharie Ménétrier, Martin Mestdagh, et al. “ReMap 2020: a database of regulatory regions from an integrative analysis of Human and Arabidopsis DNA-binding sequencing experiments”. In: *Nucleic Acids Research* 48 (D1 Jan. 8, 2020), pp. D180–D188. ISSN: 0305-1048. DOI: 10.1093/nar/gkz945. URL: https://academic.oup.com/nar/article/48/D1/D180/5608991 (visited on 07/20/2020) (cit. on p. 42).

[Che+06] Jieun Cheong, Yoichi Yamada, Riu Yamashita, et al. “Diverse DNA Methylation Statuses at Alternative Promoters of Human Genes in Various Tissues”. In: *DNA Research* 13.4 (Jan. 1, 2006), pp. 155–167. ISSN: 1340-2838. DOI: 10.1093/dnares/dsl008. URL: https://academic.oup.com/dnaresearch/article/13/4/155/348476 (visited on 11/11/2020) (cit. on p. 24).

[Chi+18] Jeffrey Chiang, Wai Lim Ku, Kairong Cui, et al. “The use of alternative promoters in T cell development”. In: *The Journal of Immunology* 200.1 (May 1, 2018), pp. 165.20–165.20. ISSN: 0022-1767, 1550-6606. URL: http://www.jimmunol.org/content/200/1_Supplement/165.20 (visited on 07/19/2018) (cit. on p. 71).

[CAP18] Justin G. Chitpin, Aseel Awdeh, and Theodore J. Perkins. “RECAP reveals the true statistical significance of ChIP-seq peak calls”. In: *bioRxiv* (Oct. 4, 2018). DOI: 10.1101/260687. URL: http://biorxiv.org/lookup/doi/10.1101/260687 (visited on 11/01/2019) (cit. on p. 45).

[CL09] Vivek S. Chopra and Mike Levine. “Combinatorial patterning mechanisms in the Drosophila embryo”. In: *Briefings in Functional Genomics* 8.4 (July 1, 2009), pp. 243–249. ISSN: 2041-2649. DOI: 10.1093/bfgp/elp026. URL: https://academic.oup.com/bfg/article/8/4/243/297941 (visited on 11/11/2020) (cit. on p. 29).

[Cia+16] Samantha Cialfi, Loredana Le Pera, Carlo De Blasio, et al. “The loss of ATP2C1 impairs the DNA damage response and induces altered skin homeostasis: Consequences for epidermal biology in Hailey-Hailey disease”. In: *Scientific Reports* 6 (Aug. 16, 2016), p. 31567. ISSN: 2045-2322. DOI: 10.1038/srep31567 (cit. on p. 77).

[Clé+18] Levin Clément, Dynomant Emeric, Gonzalez Bruno J, et al. “A data-supported history of bioinformatics tools”. In: *arXiv:1807.06808 [cs]* (July 18, 2018). arXiv: 1807.06808. URL: http://arxiv.org/abs/1807.06808 (visited on 11/16/2020) (cit. on p. 40).

[Coh88] J Cohen. *Statistical power analysis for the behavioral sciences*. 2nd. Hillsdale, New Jersey: Lawrence Erlbaum Associates, 1988 (cit. on p. 71).

[Cre+10] Menno P. Creyghton, Albert W. Cheng, G. Grant Welstead, et al. "Histone H3K27ac separates active from poised enhancers and predicts developmental state". In: *Proceedings of the National Academy of Sciences of the United States of America* 107.50 (Dec. 14, 2010), pp. 21931–21936. ISSN: 1091-6490. DOI: 10.1073/pnas.1016071107 (cit. on p. 18).

[DG12] Maria D. Chikina and Olga G. Troyanskaya. "An effective statistical evaluation of ChIPseq dataset similarity". In: *Bioinformatics* 28.5 (Mar. 1, 2012), pp. 607–613. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/bts009. URL: https://academic.oup.com/bioinformatics/article/28/5/607/247792 (visited on 11/01/2019) (cit. on p. 43).

[DSM13] Robert Daber, Shrey Sukhadia, and Jennifer J. D. Morrissette. "Understanding the limitations of next generation sequencing informatics, an approach to clinical pipeline validation using artificial data sets". In: *Cancer Genetics* 206.12 (Dec. 1, 2013), pp. 441–448. ISSN: 2210-7762. DOI: 10.1016/j.cancergen.2013.11.005. URL: http://www.sciencedirect.com/science/article/pii/S2210776213001609 (visited on 04/30/2020) (cit. on p. 100).

[Dao+17] Lan T. M. Dao, Ariel O. Galindo-Albarrán, Jaime A. Castro-Mondragon, et al. "Genome-wide characterization of mammalian promoters with distal enhancer functions". In: *Nature Genetics* 49.7 (July 2017), pp. 1073–1081. ISSN: 1546-1718. DOI: 10.1038/ng.3884 (cit. on pp. 26, 68).

[DGG16] Andrea De Mauro, Marco Greco, and Michele Grimaldi. "A formal definition of Big Data based on its essential features". In: *Library Review* 65.3 (Jan. 1, 2016), pp. 122–135. ISSN: 0024-2535. DOI: 10.1108/LR-06-2015-0061. URL: https://doi.org/10.1108/LR-06-2015-0061 (visited on 11/12/2020) (cit. on p. 39).

[De +10] Francesca De Santa, Iros Barozzi, Flore Mietton, et al. "A large fraction of extragenic RNA pol II transcription sites overlap enhancers". In: *PLoS biology* 8.5 (May 11, 2010), e1000384. ISSN: 1545-7885. DOI: 10.1371/journal.pbio.1000384 (cit. on p. 26).

[DS20] Monica Della Rosa and Mikhail Spivakov. "Silencers in the spotlight". In: *Nature Genetics* 52.3 (Mar. 2020), pp. 244–245. ISSN: 1546-1718. DOI: 10.1038/s41588-020-0583-8. URL: https://www.nature.com/articles/s41588-020-0583-8 (visited on 11/11/2020) (cit. on p. 26).

[Dem+18] Deniz Demircioğlu, Martin Kindermans, Tannistha Nandi, et al. "A Pan-Cancer Transcriptome Analysis Reveals Pervasive Regulation through Tumor-Associated Alternative Promoters". In: *bioRxiv* (May 2, 2018), p. 176487. DOI: 10.1101/176487. URL: https://www.biorxiv.org/

content/early/2018/05/02/176487 (visited on 07/12/2018) (cit. on p. 24).

[DKJ20] Devanshi Dhall, Ravinder Kaur, and Mamta Juneja. "Machine Learning: A Review of the Algorithms and Its Applications". In: *Proceedings of ICRIC 2019*. Ed. by Pradeep Kumar Singh, Arpan Kumar Kar, Yashwant Singh, et al. Lecture Notes in Electrical Engineering. Cham: Springer International Publishing, 2020, pp. 47–63. ISBN: 978-3-030-29407-6. DOI: 10.1007/978-3-030-29407-6_5 (cit. on p. 59).

[Dul10] Catherine Dulac. "Brain function and chromatin plasticity". en. In: *Nature* 465.7299 (June 2010), pp. 728–735. ISSN: 1476-4687. DOI: 10.1038/nature09231. URL: https://www.nature.com/articles/nature09231 (visited on 09/18/2020) (cit. on p. 17).

[DAB09] Cathérine Dupont, D. Randall Armant, and Carol A. Brenner. "Epigenetics: Definition, Mechanisms and Clinical Perspective". In: *Seminars in reproductive medicine* 27.5 (Sept. 2009), pp. 351–357. ISSN: 1526-8004. DOI: 10.1055/s-0029-1237423. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2791696/ (visited on 10/28/2020) (cit. on p. 15).

[Era+19] Gökcen Eraslan, Žiga Avsec, Julien Gagneur, et al. "Deep learning: new computational modelling techniques for genomics". In: *Nature Reviews Genetics* 20.7 (July 2019), pp. 389–403. ISSN: 1471-0064. DOI: 10.1038/s41576-019-0122-6. URL: https://www.nature.com/articles/s41576-019-0122-6 (visited on 04/30/2020) (cit. on p. 88).

[EK12] Jason Ernst and Manolis Kellis. "ChromHMM: automating chromatin state discovery and characterization". In: *Nature methods* 9.3 (Feb. 28, 2012), pp. 215–216. ISSN: 1548-7091. DOI: 10.1038/nmeth.1906. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3577932/ (visited on 04/30/2018) (cit. on p. 28).

[Fan19] Jianwen Fang. "Tightly integrated genomic and epigenomic data mining using tensor decomposition". In: *Bioinformatics* 35.1 (Jan. 1, 2019), pp. 112–118. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/bty513. URL: https://doi.org/10.1093/bioinformatics/bty513 (visited on 12/17/2020) (cit. on p. 56).

[FAN+14a] FANTOM Consortium and the RIKEN PMI and CLST (DGT), Alistair R. R. Forrest, Hideya Kawaji, et al. "A promoter-level mammalian expression atlas". In: *Nature* 507.7493 (Mar. 27, 2014), pp. 462–470. ISSN: 1476-4687. DOI: 10.1038/nature13182 (cit. on p. 24).

[FAN+14b] FANTOM Consortium and the RIKEN PMI and CLST (DGT), Alistair R. R. Forrest, Hideya Kawaji, et al. "A promoter-level mammalian expression atlas". In: *Nature* 507.7493 (Mar. 27, 2014), pp. 462–470. ISSN: 1476-4687. DOI: 10.1038/nature13182 (cit. on p. 42).

[Fei98] Uriel Feige. “A Threshold of ln n for Approximating Set Cover”. In: *Journal of the Acm* 45 (1998), pp. 314–318 (cit. on p. 168).

[Fer+20] Q. Ferré, G. Charbonnier, N. Sadouni, et al. “OLOGRAM: determining significance of total overlap length between genomic regions sets”. In: *Bioinformatics* 36.6 (Mar. 1, 2020), pp. 1920–1922. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/btz810. URL: https://academic.oup.com/bioinformatics/article/36/6/1920/5613178 (visited on 08/27/2020) (cit. on p. 175).

[For+20] Oriol Fornes, Jaime A. Castro-Mondragon, Aziz Khan, et al. “JASPAR 2020: update of the open-access database of transcription factor binding profiles”. In: *Nucleic Acids Research* 48 (D1 Jan. 8, 2020), pp. D87–D92. ISSN: 0305-1048. DOI: 10.1093/nar/gkz1001. URL: https://academic.oup.com/nar/article/48/D1/D87/5614568 (visited on 11/24/2020) (cit. on p. 42).

[Fre+17] Romain Frelat, Martin Lindegren, Tim Spaanheden Denker, et al. “Community ecology in 3D: Tensor decomposition reveals spatio-temporal dynamics of large ecological communities”. In: *PLOS ONE* 12.11 (Nov. 14, 2017), e0188205. ISSN: 1932-6203. DOI: 10.1371/journal.pone.0188205. URL: https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0188205 (visited on 12/17/2020) (cit. on p. 56).

[FV17] Daniel Frías-Lasserre and Cristian A. Villagra. “The Importance of ncRNAs as Epigenetic Mechanisms in Phenotypic Variation and Organic Evolution”. In: *Frontiers in Microbiology* 8 (2017). ISSN: 1664-302X. DOI: 10.3389/fmicb.2017.02483. URL: https://www.frontiersin.org/articles/10.3389/fmicb.2017.02483/full (visited on 11/10/2020) (cit. on p. 15).

[Fri+08] Martin C. Frith, Eivind Valen, Anders Krogh, et al. “A code for transcription initiation in mammalian genomes”. In: *Genome Research* 18.1 (Jan. 2008), pp. 1–12. ISSN: 1088-9051. DOI: 10.1101/gr.6831208 (cit. on p. 72).

[GK13] Kanwal Garg and Deepak Kumar. “Comparing the Performance of Frequent Pattern Mining Algorithms”. In: *International Journal of Computer Applications* 69.25 (May 17, 2013), pp. 21–28. URL: https://www.ijcaonline.org/archives/volume69/number25/12129-8502 (visited on 09/14/2020) (cit. on p. 63).

[GE13] Eugenia G. Giannopoulou and Olivier Elemento. “Inferring chromatin-bound protein complexes from genome-wide binding assays”. In: *Genome Research* 23.8 (Aug. 2013), pp. 1295–1306. ISSN: 1088-9051. DOI: 10.1101/gr.149419.112. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3730103/ (visited on 06/21/2018) (cit. on p. 64).

[Gio+10] Luca Giorgetti, Trevor Siggers, Guido Tiana, et al. “Noncooperative interactions between transcription factors and clustered DNA binding sites enable graded transcriptional responses to environmental inputs”. In: *Molecular Cell* 37.3 (Feb. 12, 2010), pp. 418–428. ISSN: 1097-4164. DOI: 10.1016/j.molcel.2010.01.016 (cit. on p. 29).

[GFI16] Steven N. Goodman, Daniele Fanelli, and John P. A. Ioannidis. “What does research reproducibility mean?” In: *Science Translational Medicine* 8.341 (2016), 341ps12. ISSN: 1946-6242. DOI: 10.1126/scitranslmed.aaf5027 (cit. on p. 50).

[Hah04] Steven Hahn. “Structure and mechanism of the RNA Polymerase II transcription machinery”. In: *Nature structural & molecular biology* 11.5 (May 2004), pp. 394–403. ISSN: 1545-9993. DOI: 10.1038/nsmb763. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC1189732/ (visited on 09/18/2020) (cit. on p. 19).

[Han04] Jiawei Han. “Mining frequent patterns without candidate generation: A frequent-pattern tree approach”. In: *Data Mining and Knowledge Discovery* 8 (Jan. 1, 2004), pp. 53–87 (cit. on p. 63).

[He+14] Housheng Hansen He, Clifford A. Meyer, Sheng’en Shawn Hu, et al. “Refined DNase-seq protocol and data analysis reveals intrinsic bias in transcription factor footprint identification”. In: *Nature Methods* 11.1 (Jan. 2014), pp. 73–78. ISSN: 1548-7105. DOI: 10.1038/nmeth.2762 (cit. on p. 47).

[HVB11] Ngoc-Diep Ho, Paul Van Dooren, and Vincent D. Blondel. “Descent Methods for Nonnegative Matrix Factorization”. In: *Numerical Linear Algebra in Signals, Systems and Control*. Ed. by Paul Van Dooren, Shankar P. Bhattacharyya, Raymond H. Chan, et al. Vol. 80. Dordrecht: Springer Netherlands, 2011, pp. 251–293. ISBN: 978-94-007-0601-9 978-94-007-0602-6. DOI: 10.1007/978-94-007-0602-6_13. URL: http://link.springer.com/10.1007/978-94-007-0602-6_13 (visited on 12/22/2020) (cit. on p. 167).

[HCG17] Sijia Huang, Kumardeep Chaudhary, and Lana X. Garmire. “More Is Better: Recent Progress in Multi-Omics Data Integration Methods”. In: *Frontiers in Genetics* 8 (2017). ISSN: 1664-8021. DOI: 10.3389/fgene.2017.00084. URL: https://www.frontiersin.org/articles/10.3389/fgene.2017.00084/full (visited on 12/16/2020) (cit. on p. 44).

[Jaf+16] Mohammad A. Jafri, Shakeel A. Ansari, Mohammed H. Alqahtani, et al. “Roles of telomeres and telomerase in cancer, and advances in telomerase-targeted therapies”. In: *Genome Medicine* 8 (June 20, 2016). ISSN: 1756-994X. DOI: 10.1186/s13073-016-0324-x. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4915101/ (visited on 11/11/2020) (cit. on p. 26).

[Jai+15] Dhawal Jain, Sandro Baldi, Angelika Zabel, et al. “Active promoters give rise to false positive ‘Phantom Peaks’ in ChIP-seq experiments”. In: *Nucleic Acids Research* 43.14 (Aug. 18, 2015), pp. 6959–6968. ISSN: 0305-1048. DOI: 10.1093/nar/gkv637. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4538825/ (visited on 04/06/2019) (cit. on p. 45).

[Jar+18] Maximilian Jaritz, Raoul de Charette, Emilie Wirbel, et al. “Sparse and Dense Data with CNNs: Depth Completion and Semantic Segmentation”. In: *arXiv:1808.00769 [cs]* (Aug. 31, 2018). arXiv: 1808.00769. URL: http://arxiv.org/abs/1808.00769 (visited on 01/05/2021) (cit. on p. 98).

[JPD16] James Jenkins, Dmitri B. Papkovsky, and Ruslan I. Dmitriev. “The Ca2+/Mn2+-transporting SPCA2 pump is regulated by oxygen and cell density in colon cancer cells”. In: *The Biochemical Journal* 473.16 (Aug. 15, 2016), pp. 2507–2518. ISSN: 1470-8728. DOI: 10.1042/BCJ20160477 (cit. on p. 77).

[Jos+20] Julie Josse, Nicolas Prost, Erwan Scornet, et al. “On the consistency of supervised learning with missing values”. In: *arXiv:1902.06931 [cs, math, stat]* (July 3, 2020). arXiv: 1902.06931. URL: http://arxiv.org/abs/1902.06931 (visited on 01/05/2021) (cit. on p. 97).

[Kap+11] Tommy Kaplan, Xiao-Yong Li, Peter J. Sabo, et al. “Quantitative Models of the Mechanisms That Control Genome-Wide Patterns of Transcription Factor Binding during Early Drosophila Development”. In: *PLOS Genetics* 7.2 (Feb. 3, 2011), e1001290. ISSN: 1553-7404. DOI: 10.1371/journal.pgen.1001290. URL: https://journals.plos.org/plosgenetics/article?id=10.1371/journal.pgen.1001290 (visited on 11/16/2020) (cit. on p. 39).

[Kel+14] Manolis Kellis, Barbara Wold, Michael P. Snyder, et al. “Defining functional DNA elements in the human genome”. In: *Proceedings of the National Academy of Sciences of the United States of America* 111.17 (Apr. 29, 2014), pp. 6131–6138. ISSN: 1091-6490. DOI: 10.1073/pnas.1318948111 (cit. on p. 23).

[15] *keras-team/keras*. Mar. 28, 2015. URL: https://github.com/keras-team/keras (visited on 06/10/2020) (cit. on p. 60).

[KHZ11] Benjamin L Kidder, Gangqing Hu, and Keji Zhao. “ChIP-Seq: technical considerations for obtaining high-quality data”. In: *Nature Immunology* 12.10 (Oct. 2011). tex.publisher: Nature Publishing Group, pp. 918–922. ISSN: 1529-2908. DOI: 10.1038/ni.2117. URL: http://www.nature.com/articles/ni.2117 (cit. on p. 45).

[Kim+16] Seong Gon Kim, Mrudul Harwani, Ananth Grama, et al. “EP-DNN: A Deep Neural Network-Based Global Enhancer Prediction Algorithm”. In: *Scientific Reports* 6 (Dec. 8, 2016), srep38433. ISSN: 2045-2322. DOI: 10.1038/srep38433. URL: https://www-nature-com.lama.univ-amu.fr/articles/srep38433 (visited on 10/17/2017) (cit. on p. 67).

[KB14] Diederik P. Kingma and Jimmy Ba. “Adam: A Method for Stochastic Optimization”. In: *arXiv:1412.6980 [cs]* (Dec. 22, 2014). arXiv: 1412.6980. URL: http://arxiv.org/abs/1412.6980 (visited on 04/06/2019) (cit. on pp. 58, 98).

[KKB16] Dimitrios Kleftogiannis, Panos Kalnis, and Vladimir B. Bajic. “Progress and challenges in bioinformatics approaches for enhancer identification”. In: *Briefings in Bioinformatics* 17.6 (Nov. 2016), pp. 967–979. ISSN: 1467-5463. DOI: 10.1093/bib/bbv101. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5142011/ (visited on 04/16/2018) (cit. on pp. 25, 67).

[KPK17] Pang Wei Koh, Emma Pierson, and Anshul Kundaje. “Denoising genome-wide histone ChIP-seq with convolutional neural networks”. In: *Bioinformatics* 33.14 (July 15, 2017), p. i225. DOI: 10.1093/bioinformatics/btx243. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5870713/ (visited on 11/07/2018) (cit. on p. 64).

[Kol+12] Petros Kolovos, Tobias A. Knoch, Frank G. Grosveld, et al. “Enhancers and silencers: an integrated and simple model for their function”. In: *Epigenetics & Chromatin* 5.1 (Jan. 9, 2012), p. 1. ISSN: 1756-8935. DOI: 10.1186/1756-8935-5-1. URL: https://doi.org/10.1186/1756-8935-5-1 (visited on 11/10/2020) (cit. on p. 25).

[Kou+06] Hosein Kouros-Mehr, Euan M. Slorach, Mark D. Sternlicht, et al. “GATA-3 Maintains the Differentiation of the Luminal Cell Fate in the Mammary Gland”. In: *Cell* 127.5 (Dec. 1, 2006), pp. 1041–1055. ISSN: 0092-8674. DOI: 10.1016/j.cell.2006.09.048. URL: http://www.sciencedirect.com/science/article/pii/S0092867406014097 (visited on 08/28/2020) (cit. on pp. 29, 169).

[KC10] Andreas Krause and Volkan Cevher. “Submodular dictionary selection for sparse representation”. In: *Proceedings of the 27th International Conference on International Conference on Machine Learning*. ICML’10. Madison, WI, USA: Omnipress, June 2010, pp. 567–574. ISBN: 978-1-60558-907-7 (cit. on p. 168).

[Kro+14] Dirk P. Kroese, T. Brereton, T. Taimre, et al. “Why the Monte Carlo method is so important today”. In: (2014). DOI: 10.1002/WICS.1314 (cit. on p. 161).

[Kun+15] Anshul Kundaje, Wouter Meuleman, Jason Ernst, et al. “Integrative analysis of 111 reference human epigenomes”. In: *Nature* 518.7539 (Feb. 2015), pp. 317–330. ISSN: 1476-4687. DOI: 10.1038/nature14248. URL: https://www.nature.com/articles/nature14248 (visited on 11/24/2020) (cit. on p. 42).

[Kuw+01] Koichiro Kuwahara, Yoshihiko Saito, Emiko Ogawa, et al. “The Neuron-Restrictive Silencer Element–Neuron-Restrictive Silencer Factor System Regulates Basal and Endothelin 1-Inducible Atrial Natriuretic Peptide Gene Expression in Ventricular Myocytes”. In: *Molecular and Cellular Biology* 21.6 (Mar. 2001), pp. 2085–2097. ISSN: 0270-7306. DOI: 10.1128/MCB.21.6.2085-2097.2001. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC86819/ (visited on 11/11/2020) (cit. on p. 26).

[Lam+18] Samuel A. Lambert, Arttu Jolma, Laura F. Campitelli, et al. “The Human Transcription Factors”. In: *Cell* 172.4 (2018), pp. 650–665. ISSN: 1097-4172. DOI: 10.1016/j.cell.2018.01.029 (cit. on p. 21).

[Lan+12] Stephen G. Landt, Georgi K. Marinov, Anshul Kundaje, et al. “ChIP-seq guidelines and practices of the ENCODE and modENCODE consortia”. In: *Genome Research* 22.9 (Sept. 1, 2012), pp. 1813–1831. ISSN: 1088-9051, 1549-5469. DOI: 10.1101/gr.136184.111. URL: http://genome.cshlp.org/content/22/9/1813 (visited on 04/28/2020) (cit. on p. 46).

[LGC09] Michael Lawrence, Robert Gentleman, and Vincent Carey. “rtracklayer: an R package for interfacing with genome browsers”. In: *Bioinformatics* 25.14 (July 15, 2009), pp. 1841–1842. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/btp328. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2705236/ (visited on 12/21/2020) (cit. on p. 157).

[Leh+18] Jaakko Lehtinen, Jacob Munkberg, Jon Hasselgren, et al. “Noise2Noise: Learning Image Restoration without Clean Data”. In: *arXiv:1803.04189 [cs, stat]* (Oct. 29, 2018). arXiv: 1803.04189. URL: http://arxiv.org/abs/1803.04189 (visited on 10/31/2019) (cit. on p. 64).

[Li14] Heng Li. “Toward better understanding of artifacts in variant calling from high-coverage samples”. In: *Bioinformatics (Oxford, England)* 30.20 (Oct. 15, 2014), pp. 2843–2851. ISSN: 1367-4811. DOI: 10.1093/bioinformatics/btu356 (cit. on p. 47).

[Li+11] Qunhua Li, James B. Brown, Haiyan Huang, et al. “Measuring reproducibility of high-throughput experiments”. In: *Annals of Applied Statistics* 5.3 (Sept. 2011), pp. 1752–1779. ISSN: 1932-6157, 1941-7330. DOI: 10.1214/11-AOAS466. URL: https://projecteuclid.org/euclid.aoas/1318514284 (visited on 11/24/2020) (cit. on p. 41).

[Li17] Xiang Li. “Classification with Large Sparse Datasets: Convergence Analysis and Scalable Algorithms”. In: *Electronic Thesis and Dissertation Repository* (July 24, 2017). URL: https://ir.lib.uwo.ca/etd/4682 (cit. on p. 97).

[LLW16] Xiang Li, Charles X. Ling, and Huaimin Wang. “The Convergence Behavior of Naive Bayes on Large Sparse Datasets”. In: *ACM Transactions on Knowledge Discovery from Data* 11.1 (July 20, 2016), 10:1–10:24. ISSN: 1556-4681. DOI: 10.1145/2948068. URL: https://doi.org/10.1145/2948068 (visited on 01/05/2021) (cit. on p. 97).

[Li+15] Yifeng Li, Chih-Yu Chen, Alice M. Kaye, et al. “The identification of cis-regulatory elements: A review from a machine learning perspective”. In: *Bio Systems* 138 (Dec. 2015), pp. 6–17. ISSN: 1872-8324. DOI: 10.1016/j.biosystems.2015.10.002 (cit. on p. 66).

[LSW16] Yifeng Li, Wenqiang Shi, and Wyeth W. Wasserman. “Genome-Wide Prediction of cis-Regulatory Regions Using Supervised Deep Learning Methods”. In: *bioRxiv* (Feb. 28, 2016), p. 041616. DOI: 10.1101/041616. URL: https://www.biorxiv.org/content/early/2016/02/28/041616 (visited on 02/21/2018) (cit. on p. 67).

[LWN16] Yifeng Li, Fang-Xiang Wu, and Alioune Ngom. “A review on machine learning principles for multi-view biological data integration”. In: *Briefings in Bioinformatics* (Dec. 22, 2016). ISSN: 1477-4054. DOI: 10.1093/bib/bbw113 (cit. on p. 43).

[Lia+04] Gangning Liang, Joy C. Y. Lin, Vivian Wei, et al. “Distinct localization of histone H3 acetylation and H3-K4 methylation to the transcription start sites in the human genome”. In: *Proceedings of the National Academy of Sciences* 101.19 (May 11, 2004), pp. 7357–7362. ISSN: 0027-8424, 1091-6490. DOI: 10.1073/pnas.0401866101. URL: https://www.pnas.org/content/101/19/7357 (visited on 11/04/2020) (cit. on p. 18).

[LPW16] Hannah K. Long, Sara L. Prescott, and Joanna Wysocka. “Ever-changing landscapes: transcriptional enhancers in development and evolution”. In: *Cell* 167.5 (Nov. 17, 2016), pp. 1170–1187. ISSN: 0092-8674. DOI: 10.1016/j.cell.2016.09.018. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5123704/ (visited on 11/11/2020) (cit. on p. 25).

[Lop+19] F. Lopez, G. Charbonnier, Y. Kermezli, et al. “Explore, edit and leverage genomic annotations using Python GTF toolkit”. In: *Bioinformatics* (Mar. 12, 2019). DOI: 10.1093/bioinformatics/btz116. URL: https://academic.oup.com/bioinformatics/advance-article/doi/10.1093/bioinformatics/btz116/5320559 (cit. on p. 175).

[LFV19] José María Luna, Philippe Fournier-Viger, and Sebastián Ventura. “Frequent itemset mining: A 25 years review”. In: *WIREs Data Mining and Knowledge Discovery* 9.6 (2019), e1329. DOI: 10.1002/widm.1329. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/widm.1329. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/widm.1329 (cit. on p. 61).

[MP15] Shaun Mahony and B. Franklin Pugh. “Protein-DNA binding in high-resolution”. In: *Critical Reviews in Biochemistry and Molecular Biology* 50.4 (2015), pp. 269–283. ISSN: 1549-7798. DOI: 10.3109/10409238.2015.1051505 (cit. on p. 35).

[Mai+09] Julien Mairal, Francis Bach, Jean Ponce, et al. “Online dictionary learning for sparse coding”. In: *Proceedings of the 26th Annual International Conference on Machine Learning - ICML ’09*. the 26th Annual International Conference. Montreal, Quebec, Canada: ACM Press, 2009, pp. 1–8. ISBN: 978-1-60558-516-1. DOI: 10.1145/1553374.1553463. URL: http://portal.acm.org/citation.cfm?doid=1553374.1553463 (visited on 06/28/2019) (cit. on p. 167).

[McD09] John H. McDonald. *Handbook of Biological Statistics*. Google-Books-ID: AsRTywAACAAJ. Sparky House Publishing, 2009. 313 pp. (cit. on p. 72).

[ML14] Clifford A. Meyer and X. Shirley Liu. “Identifying and mitigating bias in next-generation sequencing methods for chromatin biology”. en. In: *Nature Reviews Genetics* 15.11 (Nov. 2014), pp. 709–721. ISSN: 1471-0064. DOI: 10.1038/nrg3788. URL: https://www.nature.com/articles/nrg3788 (visited on 09/18/2020) (cit. on pp. 36, 37).

[MBA15] Felix Muerdter, Łukasz M. Boryń, and Cosmas D. Arnold. “STARR-seq — Principles and applications”. en. In: *Genomics*. Recent advances in functional assays of transcriptional enhancers 106.3 (Sept. 2015), pp. 145–150. ISSN: 0888-7543. DOI: 10.1016/j.ygeno.2015.06.001. URL: http://www.sciencedirect.com/science/article/pii/S0888754315300100 (visited on 09/18/2020) (cit. on p. 38).

[Mui+16] Paul Muir, Shantao Li, Shaoke Lou, et al. “The real cost of sequencing: scaling computation to keep pace with data generation”. In: *Genome Biology* 17.1 (Mar. 23, 2016), p. 53. ISSN: 1474-760X. DOI: 10.1186/s13059-016-0917-0. URL: https://doi.org/10.1186/s13059-016-0917-0 (visited on 11/16/2020) (cit. on p. 40).

[Ngu+18] Nga Thi Thuy Nguyen, Bruno Contreras-Moreira, Jaime A. Castro-Mondragon, et al. “RSAT 2018: regulatory sequence analysis tools 20th anniversary”. In: *Nucleic Acids Research* 46 (W1 July 2, 2018), W209–W214. ISSN: 0305-1048. DOI: 10.1093/nar/gky317. URL: https://academic.oup.com/nar/article/46/W1/W209/4990780 (visited on 11/20/2020) (cit. on p. 39).

[Ogr+96] Vasily V. Ogryzko, R. Louis Schiltz, Valya Russanova, et al. “The Transcriptional Coactivators p300 and CBP Are Histone Acetyltransferases”. In: *Cell* 87.5 (Nov. 29, 1996), pp. 953–959. ISSN: 0092-8674, 1097-4172. DOI: 10.1016/S0092-8674(00)82001-2. URL: https://www.cell.com/cell/abstract/S0092-8674(00)82001-2 (visited on 09/22/2020) (cit. on p. 18).

[OO03] Donald E. Olins and Ada L. Olins. “Chromatin history: our view from the bridge”. In: *Nature Reviews Molecular Cell Biology* 4.10 (Oct. 2003), pp. 809–814. ISSN: 1471-0080. DOI: 10.1038/nrm1225. URL: https://www.nature.com/articles/nrm1225 (visited on 10/28/2020) (cit. on p. 17).

[OHC19] Oluwatosin Oluwadare, Max Highsmith, and Jianlin Cheng. “An Overview of Methods for Reconstructing 3-D Chromosome and Genome Structures from Hi-C Data”. In: *Biological Procedures Online* 21.1 (Apr. 24, 2019), p. 7. ISSN: 1480-9222. DOI: 10.1186/s12575-019-0094-0. URL: https://doi.org/10.1186/s12575-019-0094-0 (visited on 11/20/2020) (cit. on p. 38).

[OPH18] E. Ordway-West, P. Parveen, and A. Henslee. “Autoencoder Evaluation and Hyper-Parameter Tuning in an Unsupervised Setting”. In: *2018 IEEE International Congress on Big Data (BigData Congress)*. 2018 IEEE International Congress on Big Data (BigData Congress). July 2018, pp. 205–209. DOI: 10.1109/BigDataCongress.2018.00034 (cit. on p. 101).

[Pal+11] Sharmistha Pal, Ravi Gupta, Hyunsoo Kim, et al. “Alternative transcription exceeds alternative splicing in generating the transcriptome diversity of cerebellar development”. In: *Genome Research* 21.8 (Aug. 2011), pp. 1260–1272. ISSN: 1549-5469. DOI: 10.1101/gr.120535.111 (cit. on p. 24).

[Par09] Peter J. Park. “ChIP–seq: advantages and challenges of a maturing technology”. In: *Nature Reviews Genetics* 10.10 (Oct. 2009). tex.publisher: Nature Publishing Group, pp. 669–680. ISSN: 1471-0056. DOI: 10.1038/nrg2641. URL: http://www.nature.com/articles/nrg2641 (cit. on p. 34).

[Ped+11] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, et al. “Scikit-learn: Machine Learning in Python”. In: *Journal of Machine Learning Research* 12 (Oct. 2011), pp. 2825–2830. URL: http://jmlr.csail.mit.edu/papers/v12/pedregosa11a.html (visited on 02/08/2018) (cit. on p. 59).

[PD15] Ana Pombo and Niall Dillon. “Three-dimensional genome architecture: players and mechanisms”. In: *Nature Reviews Molecular Cell Biology* 16.4 (Apr. 2015), pp. 245–257. ISSN: 1471-0080. DOI: 10.1038/nrm3965.

URL: https://www.nature.com/articles/nrm3965 (visited on 11/11/2020) (cit. on p. 26).

[Pon+05] Nikolay Ponomarenko, Vladimir Lukin, Mikhail Zriakhov, et al. "Lossy Compression of Images with Additive Noise". In: *Advanced Concepts for Intelligent Vision Systems*. Ed. by Jacques Blanc-Talon, Wilfried Philips, Dan Popescu, et al. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2005, pp. 381–386. ISBN: 978-3-540-32046-3 (cit. on p. 61).

[Qiu20] Peng Qiu. "Embracing the dropouts in single-cell RNA-seq analysis". In: *Nature Communications* 11.1 (Mar. 3, 2020), p. 1169. ISSN: 2041-1723. DOI: 10.1038/s41467-020-14976-9. URL: https://www.nature.com/articles/s41467-020-14976-9 (visited on 11/12/2020) (cit. on p. 33).

[QX15] Daniel Quang and Xiaohui Xie. "DanQ: a hybrid convolutional and recurrent deep neural network for quantifying the function of DNA sequences". In: *bioRxiv* (Dec. 20, 2015), p. 032821. DOI: 10.1101/032821. URL: https://www.biorxiv.org/content/early/2015/12/20/032821 (visited on 02/27/2018) (cit. on p. 67).

[QH10] Aaron R. Quinlan and Ira M. Hall. "BEDTools: a flexible suite of utilities for comparing genomic features". In: *Bioinformatics* 26.6 (Mar. 15, 2010), pp. 841–842. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/btq033. URL: https://doi.org/10.1093/bioinformatics/btq033 (visited on 12/16/2020) (cit. on p. 52).

[RTM12] Dietmar Rieder, Zlatko Trajanoski, and James McNally. "Transcription factories". In: *Frontiers in Genetics* 3 (2012). ISSN: 1664-8021. DOI: 10.3389/fgene.2012.00221. URL: https://www.frontiersin.org/articles/10.3389/fgene.2012.00221/full (visited on 11/11/2020) (cit. on p. 26).

[Ros+12] Caryn S. Ross-Innes, Rory Stark, Andrew E. Teschendorff, et al. "Differential oestrogen receptor binding is associated with clinical outcome in breast cancer". In: *Nature* 481.7381 (Jan. 2012), pp. 389–393. ISSN: 1476-4687. DOI: 10.1038/nature10730. URL: https://www.nature.com/articles/nature10730 (visited on 08/28/2020) (cit. on pp. 18, 29, 169).

[RHW86] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. "Learning representations by back-propagating errors". In: *Nature* 323.6088 (Oct. 1986), pp. 533–536. ISSN: 1476-4687. DOI: 10.1038/323533a0. URL: https://www.nature.com/articles/323533a0 (visited on 01/18/2021) (cit. on p. 88).

[San+17] David Santiago-Algarra, Lan T.M. Dao, Lydie Pradel, et al. “Recent advances in high-throughput approaches to dissect enhancer function”. In: *F1000Research* 6 (June 19, 2017), p. 939. ISSN: 2046-1402. DOI: 10.12688/f1000research.11581.1. URL: https://f1000research.com/articles/6-939/v1 (visited on 10/24/2017) (cit. on p. 38).

[Sch+12] Sophie Schbath, Véronique Martin, Matthias Zytnicki, et al. “Mapping reads on a genomic sequence: an algorithmic overview and a practical comparative analysis”. In: *Journal of Computational Biology: A Journal of Computational Molecular Cell Biology* 19.6 (June 2012), pp. 796–813. ISSN: 1557-8666. DOI: 10.1089/cmb.2012.0022 (cit. on p. 33).

[Sch15] Jürgen Schmidhuber. “Deep learning in neural networks: An overview”. In: *Neural Networks* 61 (Jan. 1, 2015), pp. 85–117. ISSN: 0893-6080. DOI: 10.1016/j.neunet.2014.09.003. URL: http://www.sciencedirect.com/science/article/pii/S0893608014002135 (visited on 12/18/2020) (cit. on p. 88).

[SH76] M. I. Shamos and D. Hoey. “Geometric intersection problems”. In: *17th Annual Symposium on Foundations of Computer Science (sfcs 1976)*. 17th Annual Symposium on Foundations of Computer Science (sfcs 1976). Oct. 1976, pp. 208–215. DOI: 10.1109/SFCS.1976.16 (cit. on p. 162).

[SS16] Sara Sheehan and Yun Song. “Deep Learning for Population Genetic Inference”. In: *PLOS Computational Biology* 12 (Mar. 28, 2016), e1004845. DOI: 10.1371/journal.pcbi.1004845 (cit. on p. 88).

[She+16] Susan Q. Shen, Connie A. Myers, Andrew E. O. Hughes, et al. “Massively parallel cis-regulatory analysis in the mammalian central nervous system”. In: *Genome Research* 26.2 (Feb. 2016), pp. 238–255. ISSN: 1549-5469. DOI: 10.1101/gr.193789.115 (cit. on p. 41).

[Shi+09] Hyunjin Shin, Tao Liu, Arjun K. Manrai, et al. “CEAS: cis-regulatory element annotation system”. In: *Bioinformatics* 25.19 (Oct. 1, 2009), pp. 2605–2606. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/btp479. URL: https://academic.oup.com/bioinformatics/article/25/19/2605/182052 (visited on 11/10/2020) (cit. on p. 24).

[Sim+18] Boris Simovski, Chakravarthi Kanduri, Sveinung Gundersen, et al. “Coloc-stats: a unified web interface to perform colocalization analysis of genomic features”. In: *Nucleic Acids Research* 46 (W1 July 2, 2018), W186–W193. ISSN: 0305-1048. DOI: 10.1093/nar/gky474. URL: https://academic.oup.com/nar/article/46/W1/W186/5033159 (visited on 11/03/2018) (cit. on pp. 64, 159).

[Siw+16] Geoffrey Siwo, Andrew Rider, Asako Tan, et al. “Prediction of fine-tuned promoter activity from DNA sequence”. In: *F1000Research* 5 (2016), p. 158. ISSN: 2046-1402. DOI: 10.12688/f1000research.7485.1 (cit. on p. 67).

[SK03] Stephen T. Smale and James T. Kadonaga. “The RNA polymerase II core promoter”. In: *Annual Review of Biochemistry* 72 (2003), pp. 449–479. ISSN: 0066-4154. DOI: 10.1146/annurev.biochem.72.121801.161520 (cit. on p. 23).

[Sny+20] Michael P. Snyder, Thomas R. Gingeras, Jill E. Moore, et al. “Perspectives on ENCODE”. en. In: *Nature* 583.7818 (July 2020), pp. 693–698. ISSN: 1476-4687. DOI: 10.1038/s41586-020-2449-8. URL: https://www.nature.com/articles/s41586-020-2449-8 (visited on 09/18/2020) (cit. on p. 40).

[SF12] François Spitz and Eileen E. M. Furlong. “Transcription factors: from enhancer binding to developmental control”. eng. In: *Nature Reviews. Genetics* 13.9 (Sept. 2012), pp. 613–626. ISSN: 1471-0064. DOI: 10.1038/nrg3207 (cit. on p. 29).

[Ste+08] William Stedman, Hyojeung Kang, Shu Lin, et al. “Cohesins localize with CTCF at the KSHV latency control region and at cellular c-myc and H19/Igf2 insulators”. In: *The EMBO Journal* 27.4 (Feb. 20, 2008), pp. 654–666. ISSN: 0261-4189. DOI: 10.1038/emboj.2008.1. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2262040/ (visited on 11/09/2019) (cit. on p. 29).

[SA00] B. D. Strahl and C. D. Allis. “The language of covalent histone modifications”. In: *Nature* 403.6765 (Jan. 6, 2000), pp. 41–45. ISSN: 0028-0836. DOI: 10.1038/47412 (cit. on p. 28).

[Sub+20] Indhupriya Subramanian, Srikant Verma, Shiva Kumar, et al. “Multi-omics Data Integration, Interpretation, and Its Application”. In: *Bioinformatics and Biology Insights* 14 (Jan. 1, 2020), p. 1177932219899051. ISSN: 1177-9322. DOI: 10.1177/1177932219899051. URL: https://doi.org/10.1177/1177932219899051 (visited on 11/11/2020) (cit. on p. 43).

[Sup19] Warisa Thangjai Suparat Niwitpong. “Confidence intervals for the signal-to-noise ratio of gamma distributions”. In: *Ijeit* (2019). DOI: 10.17605/OSF.IO/CFKM7. URL: https://osf.io/cfkm7/ (visited on 01/05/2021) (cit. on p. 239).

[TT19] Y.-h Taguchi and Turki Turki. “Tensor decomposition-Based Unsupervised Feature Extraction Applied to Single-Cell Gene Expression Analysis”. In: *bioRxiv* (Aug. 10, 2019), p. 684225. DOI: 10.1101/684225. URL: https://www.biorxiv.org/content/10.1101/684225v2 (visited on 12/17/2020) (cit. on p. 56).

[Tak+12] Hazuki Takahashi, Sachiko Kato, Mitsuyoshi Murata, et al. “CAGE- Cap Analysis Gene Expression: a protocol for the detection of promoter and transcriptional networks”. In: *Methods in molecular biology (Clifton, N.J.)* 786 (2012), pp. 181–200. ISSN: 1064-3745. DOI: 10.1007/978-1-61779-

292-2_11. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4094367/ (visited on 11/24/2020) (cit. on p. 42).

[TB17] Aurélie Teissandier and Déborah Bourc'his. "Gene body DNA methylation conspires with H3K36me3 to preclude aberrant transcription". In: *The EMBO Journal* 36.11 (June 1, 2017), pp. 1471–1473. ISSN: 0261-4189. DOI: 10.15252/embj.201796812. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5452023/ (visited on 11/04/2020) (cit. on p. 18).

[Ten+14] Li Teng, Bing He, Peng Gao, et al. "Discover context-specific combinatorial transcription factor interactions by integrating diverse ChIP-Seq data sets". In: *Nucleic Acids Research* 42.4 (Feb. 1, 2014), e24–e24. ISSN: 0305-1048. DOI: 10.1093/nar/gkt1105. URL: https://academic.oup.com/nar/article/42/4/e24/2435199 (visited on 04/30/2018) (cit. on p. 64).

[Ten+16] Mingxiang Teng, Michael I. Love, Carrie A. Davis, et al. "A benchmark for RNA-seq quantification pipelines". In: *Genome Biology* 17.1 (Apr. 23, 2016), p. 74. ISSN: 1474-760X. DOI: 10.1186/s13059-016-0940-1. URL: https://doi.org/10.1186/s13059-016-0940-1 (visited on 11/24/2020) (cit. on p. 41).

[Ter+18] Christopher Terranova, Ming Tang, Elias Orouji, et al. "An Integrated Platform for Genome-wide Mapping of Chromatin States Using High-throughput ChIP-sequencing in Tumor Tissues". In: *JoVE (Journal of Visualized Experiments)* 134 (Apr. 2018), e56972. ISSN: 1940-087X. DOI: 10.3791/56972. URL: https://www.jove.com/v/56972/an-integrated-platform-for-genome-wide-mapping-chromatin-states-using (visited on 09/18/2020) (cit. on p. 28).

[Tey+13] Leonid Teytelman, Deborah M. Thurtle, Jasper Rine, et al. "Highly expressed loci are vulnerable to misleading ChIP localization of multiple unrelated proteins". In: *Proceedings of the National Academy of Sciences of the United States of America* 110.46 (Nov. 12, 2013), pp. 18602–18607. ISSN: 1091-6490. DOI: 10.1073/pnas.1316064110 (cit. on p. 45).

[The12] The ENCODE Consortium. "An integrated encyclopedia of DNA elements in the human genome". In: *Nature* 489.7414 (Sept. 2012), pp. 57–74. ISSN: 1476-4687. DOI: 10.1038/nature11247. URL: https://www.nature.com/articles/nature11247 (visited on 11/24/2020) (cit. on pp. 29, 41, 164).

[Tib96] Robert Tibshirani. "Regression Shrinkage and Selection via the Lasso". In: *Journal of the Royal Statistical Society. Series B (Methodological)* 58.1 (1996), pp. 267–288. ISSN: 0035-9246. URL: https://www.jstor.org/stable/2346178 (visited on 01/18/2021) (cit. on p. 97).

[TB14] Maria Tsompana and Michael J. Buck. “Chromatin accessibility: a window into the genome”. In: *Epigenetics & Chromatin* 7.1 (2014), p. 33. ISSN: 1756-8935. DOI: 10.1186/1756-8935-7-33 (cit. on p. 47).

[UKA04] Takeaki Uno, Masashi Kiyomi, and Hiroki Arimura. “LCM ver. 2: Efficient Mining Algorithms for Frequent/Closed/Maximal Itemsets”. In: Jan. 1, 2004 (cit. on p. 63).

[Van+17] Jimmy Vandel, Oceane Cassan, Sophie Lebre, et al. “Modeling transcription factor combinatorics in promoters and enhancers”. In: *bioRxiv* (Oct. 2, 2017), p. 197418. DOI: 10.1101/197418. URL: https://www.biorxiv.org/content/early/2017/10/02/197418 (visited on 02/20/2018) (cit. on p. 64).

[Van+15] Laurent Vanhille, Aurélien Griffon, Muhammad Ahmad Maqbool, et al. “High-throughput and quantitative assessment of enhancer activity in mammals by CapStarr-seq”. In: *Nature Communications* 6 (Apr. 15, 2015), p. 6905. ISSN: 2041-1723. DOI: 10.1038/ncomms7905 (cit. on p. 38).

[VS12] Nadine L Vastenhouw and Alexander F Schier. “Bivalent histone modifications in early embryogenesis”. In: *Current Opinion in Cell Biology*. Nucleus and gene expression 24.3 (June 1, 2012), pp. 374–386. ISSN: 0955-0674. DOI: 10.1016/j.ceb.2012.03.009. URL: http://www.sciencedirect.com/science/article/pii/S095506741200052X (visited on 11/11/2020) (cit. on p. 29).

[Ver+20] Thijs C. J. Verheul, Levi van Hijfte, Elena Perenthaler, et al. “The Why of YY1: Mechanisms of Transcriptional Regulation by Yin Yang 1”. In: *Frontiers in Cell and Developmental Biology* 8 (2020). ISSN: 2296-634X. DOI: 10.3389/fcell.2020.592164. URL: https://www.frontiersin.org/articles/10.3389/fcell.2020.592164/full (visited on 11/05/2020) (cit. on p. 22).

[VLS11] Jilles Vreeken, Matthijs Leeuwen, and Arno Siebes. “KRIMP: Mining itemsets that compress”. In: *Data Min. Knowl. Discov.* 23 (July 1, 2011), pp. 169–214. DOI: 10.1007/s10618-010-0202-x (cit. on p. 63).

[WIB15] Kai Wei, Rishabh Iyer, and Jeff Bilmes. “Submodularity in Data Subset Selection and Active Learning”. In: *International Conference on Machine Learning*. International Conference on Machine Learning. PMLR, June 1, 2015, pp. 1954–1963. URL: http://proceedings.mlr.press/v37/wei15.html (visited on 09/22/2020) (cit. on p. 167).

[WF10] Elizabeth G. Wilbanks and Marc T. Facciotti. “Evaluation of Algorithm Performance in ChIP-Seq Peak Detection”. In: *PLOS ONE* 5.7 (July 8, 2010), e11471. ISSN: 1932-6203. DOI: 10.1371/journal.pone.0011471. URL: https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0011471 (visited on 04/29/2020) (cit. on p. 45).

[Xu+19] Jinrui Xu, Michelle M. Kudron, Alec Victorsen, et al. “To mock or not: a comprehensive comparison of mock IP and DNA input for ChIP-seq”. In: *bioRxiv* (Dec. 18, 2019), p. 2019.12.17.880013. DOI: 10.1101/2019.12.17.880013. URL: https://www.biorxiv.org/content/10.1101/2019.12.17.880013v1 (visited on 12/23/2020) (cit. on pp. 45, 46).

[XLY19] Xiaodan Xu, Huawen Liu, and Minghai Yao. “Recent Progress of Anomaly Detection”. In: *Complexity* 2019 (Jan. 13, 2019). Ed. by David Gil, p. 2686378. ISSN: 1076-2787. DOI: 10.1155/2019/2686378. URL: https://doi.org/10.1155/2019/2686378 (cit. on p. 60).

[Yan+07] Chuhu Yang, Eugene Bolotin, Tao Jiang, et al. “Prevalence of the Initiator over the TATA box in human and yeast genes and identification of DNA motifs enriched in human TATA-less core promoters”. In: *Gene* 389.1 (Mar. 1, 2007), pp. 52–65. ISSN: 0378-1119. DOI: 10.1016/j.gene.2006.09.029. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC1955227/ (visited on 11/10/2020) (cit. on p. 24).

[Yan+04] Jing Yang, Sendurai A. Mani, Joana Liu Donaher, et al. “Twist, a master regulator of morphogenesis, plays an essential role in tumor metastasis”. In: *Cell* 117.7 (June 25, 2004), pp. 927–939. ISSN: 0092-8674. DOI: 10.1016/j.cell.2004.06.006 (cit. on p. 29).

[Zak00] M. J. Zaki. “Scalable algorithms for association mining”. In: *IEEE Transactions on Knowledge and Data Engineering* 12.3 (May 2000), pp. 372–390. ISSN: 1558-2191. DOI: 10.1109/69.846291 (cit. on p. 63).

[ZC15] M. Zhou and L. Carin. “Negative Binomial Process Count and Mixture Modeling”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 37.2 (Feb. 2015), pp. 307–320. ISSN: 1939-3539. DOI: 10.1109/TPAMI.2013.211 (cit. on p. 160).

Annexes

A. Modelisation of Cap-STARR-Seq data

Problem The goal of Cap-STARR-Seq is to estimate the enhancer activity of a region. More details about the procedure are presented in section 1.2.3. In the following, let "clone" designate one fragment (after sonication) of the genome. As sonication is random, there will be several clones for each genomic region of interest. For each clone, two values are obtained experimentally: the input (aka. "library") value, given by the RNA-Seq intensity of simply sequencing the Cap-STARR-Seq vector, and a "cDNA" value given by the same sequencing in the case where the vector was put inside a transfected cell.

Simulated reads For each clone i, let I_i be the input value and C_i the cDNA value. Activity was defined as $a_i = \frac{C_i}{I_i}$. The likelihood of the activities is assessed relative to a model where the null hypothesis is based on the number of reads observed in input. But all possible input values are not present in the data, so simulated reads are generated by N. Sadouni and D. Van Essen to fill in the gaps using the following procedure: $\forall i \in [1; l]$ where l is the number of clones, two sets of simulated read counts A_i and B_i were generated under the null hypothesis with a Poisson law of $\mathsf{Pois}(\lambda = I_i)$.

The likelihood of observing a given read counts value of k is given by the distribution of the values of B_i for clones where $A_i = k$. My interpretation of this is that, if the true λ is unknown, we want to express the probability mass function of one sample in one set as a function of the value observed in another, knowing that for the same i the values were generated by a Poisson law of the same (unknown) λ. Crucially, the two samplings are independent. This is tantamount to estimating this, where λ is unknown :

$$P(B_i = \beta | A_i = \alpha) = P_{\mathsf{Pois}(\lambda)}(\beta)$$

This does not depend on A_i since the samplings are are independent, but λ is unknown. Indeed, our best estimator for λ is A_i, but this estimator has a variance of, well, $\lambda/n = \lambda$ since $n = 1$ (only one observation of α). Empirically, we observed that this results in negative binomial distribution with a mean of $\mu = A_i$, but a variance of $\sigma^2 = 2 \times A_i$.

Indeed, if we use a Gamma prior for the value of λ, this is a conjugate prior with the Poisson distribution. As a result, it is possible to express this probability depending on the parameters given in the prior. Those parameters can be estimated from the observed read counts, on which λ is based. If we have the prior $\lambda \hookrightarrow \mathsf{Gamma}(\lambda; \alpha, \beta)$ in a generalized three-parameter gamma distribution, then we have $P(\lambda|x) = \mathsf{Gamma}(\lambda; \alpha + x, \beta + 1)$ according to Bayes' theorem after simplifying some terms. Rigorous demonstration is pending.

Signal-noise ratio Very low values might result in falsely significant ratio. For example $\frac{2}{1} = \frac{2000}{1000}$ but the former ratio may observed simply owing to variance in the counting, but this is is unlikely for the latter ratio. This signal-noise problem is reminescent of that which I encountered when working on ChIP-seq counts when studying alternative promoters (see section 2.2.1).

This is mitigated by having several clones per region, but is still an issue. It is compounded by the fact that we are studying candidate silencers, which means we are looking at depletions compared to the already low input values. Empirically, we have observed that the logarithms of the cDNA and input values follow an exponential distribution, which we model as a Gamma distribution (of which it is a variant). As a result, the signal-to-noise ratio of μ/σ has a confidence interval that depends on the α parameter (Suparat Niwitpong 2019). With an exponential distribution, α is low, and as such I believe we should discard all values below the standard deviation for the counts σ to avoid false positives. This can be appreciated by drawing the Lorentz curve: empirically the inflexion point is indeed around σ.

Hypothesis deciding It is assumed that the I (input counts) follow Negative Binomial distributions, with a different distribution for each possible value of input. This goes back to a very common postulate that RNA-seq counts data follows a Negative Binomial distribution of mean equal to the "true" biological value. A criticism I raised at this point is that I see a discrepancy in moving from a Poisson distribution to a Negative Binomial in this modeling, as the counts were assumed to follow a Poisson law before.

Finally, we want to decide for each clone if its activity is significantly different from one. The likelihood ratio is defined by D. Van Essen as:

$$LR = \frac{\mathscr{L}(a_i = a_{obs} | \mathscr{NB})}{\mathscr{L}(a_i = 1 | \mathscr{NB})}$$

In this modeling, the random variable of interest would be $Y = a_i * X$, where X follows a Negative Binomial distribution based on the input value, and a_i is the activity. With the Negative Binomial distribution fixed, a_i is the only parameter of the model. I proposed to instead used a modeling where Y follows a Negative Binomial law with a mean of $\mu = a_i * E(X)$ and variance of $\sigma^2 = a_i^2 * V(X)$ to add more granularity.

This modeling works because $a = a_{\text{observed}}$ is the Maximum Likelihood estimation of the activity needed to observe this particular cDNA for this particular output. The Chi-Squared test can then be used to assess the significance of this value, according to Wilkes' theorem.

My concern here was that this is only works if we have a discrete activity "cutoff" and we seek to determine whether the activity a_i is equal to the cutoff. A more general test seeking to detemin whether the activity is lower than or equal to this cutoff would violate Wilks' theorem applicability conditions, as the parameters are not in the interior of the parameter space. This does not negate the power of the likelihood ratio under the Neyman-Pearson lemma, though. Furthermore, we need to verify that

the likelihood are normal-distributed, otherwise Wilks' theorem is unapplicable. I suggested using the Lagrange multiplier test instead, or its variant with the Fisher information. This could be done by using a numerical estimation of the infinite integral.