Data depth: computation, applications, and beyond

Pavlo Mozharovskyi

▶ **To cite this version:**

Pavlo Mozharovskyi. Data depth: computation, applications, and beyond. Statistics [stat]. Institut Polytechnique de Paris, 2022. tel-03780415

HAL Id: tel-03780415
https://hal.science/tel-03780415

Submitted on 19 Sep 2022

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

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

INSTITUT POLYTECHNIQUE DE PARIS

Mémoire d'habilitation à diriger des recherches

Data depth: computation, applications, and beyond

Mémoire d'habilitation à diriger des recherches de l'Institut Polytechnique de Paris
préparée à Télécom Paris

École doctorale n°626 École doctorale de l'Institut Polytechnique de Paris (EDIPP)
Spécialité : Mathématiques

Mémoire présenté et soutenu à Palaiseau, le 22 juillet 2022, par

PAVLO MOZHAROVSKYI

Composition du Jury :

Gérard Biau Professor, Sorbonne Université	Examinateur
Victor-Emmanuel Brunel Professor, ENSAE, Institut Polytechnique de Paris	Rapporteur
Guillaume Lecué Professor, ENSAE, Institut Polytechnique de Paris	Examinateur
Regina Liu Distinguished Professor, Rutgers University	Présidente
Davy Paindaveine Professor, Université Libre de Bruxelles	Rapporteur
Peter Rousseeuw Professor, Katholieke Universiteit Leuven	Rapporteur
Richard J. Samworth Professor, University of Cambridge	Examinateur

Contents

Pre-word

My research during the PhD study acomplished at the University of Cologne under supervision of Karl Mosler was focused on depth-based classification and preliminary studies on computation of the halfspace depth, which is accumulated in the following work: M. (2015). As a result of the PhD, the $DD\alpha$-procedure has been developed—a fast non-parametric supervised machine learning method based on a robust data-depth-based iterative heuristic, which resulted in a series of works: Lange et al. (2014b,a), Lange and M. (2014), M. et al. (2015), Mosler and M. (2017). Further, fast depth-calculating and -approximating algorithms (see also the later composed report M., 2016) were proposed, with part of them, together with the $DD\alpha$-procedure, implemented in `R`-package `ddalpha` (Pokotylo et al., 2020).

In the current manuscript, research developments following the PhD degree (*Dr. rer. pol.*) which are closely related to data depth, as well as selected future projects, will be presented.

Story, structure, interconnections

My scientific life following the PhD-equivalent degree started with a post-doctorate at the same university. I did a year of postdoc at the University of Cologne, where still keeping exploring the computational aspects of the data depth (Dyckerhoff and M., 2016, and later works), I started to extend the area of research to higher-dimensional data (Pokotylo et al., 2019) and nonparametric data envelopment analysis (Badunenko and M., 2016), as well as likelihood estimation in spatial models for large samples (M. and Vogler, 2016). In September 2015 I moved to Rennes for a postdoc granted by the Henri Lebesgue Center for Mathematics. Here, under supervision of Julie Josse and François Husson, I continued broadening my research areas by working during a year on imputation of missing data (M. et al., 2020).

I started a tenure-track Assistant Professor position at the École National de la Statistique et de l'Analyse de l'Information (Ensai) in September 2016, where I pursued the research in the areas of machine learning and data depth. Thus, in collaboration with Myriam Vimond and Pierre Lafaye De Micheaux, we were working on development of a new notion of depth whose advantage is additionally invariance w.r.t. the functional argument (Lafaye De Micheaux et al., 2022). Together with Xiaohui Liu, Karl Mosler (Liu et al., 2019) and Rainer Dyckerhoff I participated in creation of algorithms for exact and approximate computation of depths and depth-trimmed regions. With Valentin Patilea and Laurent Rouviere we were working on tests for stability of parametric regressions.

In September 2018 I joined Télécom Paris for the permanent Associate Professor position, where I am currently working. A dynamic environment of the Image, Data, Signal department allowed me to finish some of the previous projects (notably those on approximate computation of data depth: Nagy et al., 2020, Dyckerhoff et al., 2021), consolidate my view on depth-based statistics (see the survey Mosler and M., 2022) and further broaden my research interests in directions of machine learning. First such direction concerns functional anomaly detection (Staerman et al., 2019), including depth based (Staerman et al., 2020) methods, as well as development of higher-dimensional techniques and large-scale industrial-data applications of those, which is submitted work with Guillaume Staerman, my PhD student co-supervised with Florence D'Alché-Buc and Stephan Clémençon who defended his PhD thesis on the 12^{th} of April 2022. Collaborative work with Guillaume further yielded results on robust estimation of Wasserstein distance (Staerman et al., 2021). As a by-product, in collaboration with Morgane Goibert, Stephan Clémençon and Ekhine Irurozki, a notion of data depth for ranking distributions has been proposed (Goibert et al., 2022). As a second direction of extensions of research interests, I have started to work with Jayneel Parekh—a PhD student co-supervised with Florence D'Alché-Buc—on explainability of machine learning, notably of the artificial neural networks. Until now, these developments have resulted in a collaborative report Beaudouin et al. (2020a) as well as a publications Beaudouin et al. (2020b) and Parekh et al. (2021). Further, work is also ongoing in econometric and biological applications, with examples being Badunenko and M. (2020) and Statzer et al. (2021), and more projects in pipeline.

Current manuscript gathers the part of the material that addresses the notion of statistical data depth function or is immediately related to this, which was developed during my research journey. With increasing number of research areas and scientific directions, this journey has always contained data depth as its constituent, which fascinates me until now in its generality. The work consists of chapters, whose sections are compiled from articles and conference proceedings in a manner described few paragraphs below. To maintain the practical spirit, all proofs are skipped here and can be found in the corresponding articles, their appendices, and supplementary materials. Introductory notions of the manuscript (to be found in Sections 1.1 and 3.1, and at the beginnings of Chapters 2 and 4) are minimally necessary for understanding of the subsequent material, and thus the reader is encouraged to consult additional sources, perhaps starting with those cited in this manuscript.

Introduced in the second half of the twentieth century by Tukey (1975), data depth became a universal methodology for ordering of complex data. To clearly fix the ideas and ensure understanding of the subsequent material, the data depth is defined right below, for multivariate data being the simplest case. Data depth is a statistical function that measures centrality of an observation with respect to (w.r.t.) a data distribution, with an empirical distribution (on a data set) being its most important particular case. Consider a data set in Euclidean space consisting of n observations, say $\boldsymbol{X} = \{\boldsymbol{x}_1, ..., \boldsymbol{x}_n\}$ in $\mathbb{R}^d$. For an arbitrary point of the same space $\boldsymbol{x} \in \mathbb{R}^d$, data depth is the following function:

$$D : \mathbb{R}^d \times \mathbb{R}^{n \times d} \to [0, 1], \, (\boldsymbol{x}, \boldsymbol{X}) \mapsto D(\boldsymbol{x} | \boldsymbol{X}) \, .$$

This value characterizes how central $\boldsymbol{x}$ is located in $\boldsymbol{X}$. The higher the value of the depth, the more central is $\boldsymbol{x}$, and *vice versa* low values of depth indicate apartness of $\boldsymbol{x}$.

Chapter 1 tackles the important question of computation of data depth. Being a formulated statistical notion since at least 30 years, long time data depth has rather been known as an attractive theoretical concept with little progress on the calculation part. This accusation was not unfounded: already in 1978 Johnson and Preparata have shown that solving the problem equivalent to computing the halfspace depth (of a point in Euclidean space w.r.t. a data set in that space)—one of the most celebrated depth notions—is an NP-complete problem in (n, d); in 1992 Donoho and Gasko wrote "... One gives up hope of exactly computing the suprema indicated at various places in the definition of one's estimator ...". Only very recent years testified substantial computational advances (see Section 1.1.3 for more details), which attracted greater audience to the notion of data depth. With the hope that the content of Chapter 1 has made its contribution to the stream of depth-computing literature, let us detail its sections. Sections 1.2 (based on Dyckerhoff and M., 2016) and 1.3 (based on Liu et al., 2019) treat the task of exact computation of the halfspace depth and its regions. Even with most efficient algorithms, exponential complexity shows itself with growing n and d. With approximation suggesting the remedy in this case, Section 1.4 (based on Dyckerhoff et al., 2021) proposes an algorithmic framework for this based on zero-order methods. Next, Section 1.5 (based on Nagy et al., 2020) derives very first theoretical guarantees for simplest (random) algorithms.

Several notions of data depth have been implemented in R-package `ddalpha`, which functionality is briefly described in Section A.1 (based on Pokotylo et al., 2019). The algorithms of Section 1.3 have been implemented in R-package `TukeyRegion`, briefly addressed in Section A.2. Ideas of Section 1.3 related to geometrical computation on convex polytopes inspired development of an efficient algorithm for nonparametric frontier analysis and its accompanying implementation, see Section A.3 (based on Badunenko and M., 2016) for more details. Implementation of algorithms from Section 1.4 in R-package `ddalpha` is currently ongoing.

Chapter 2 (based on M. et al., 2020) proposes an example of application of data depth related to missing data literature. Based on data depth, a universal framework for imputation of missing values is introduced, which fills the gap between model-assuming parametric and non-extrapolating nonparametric methods. It builds up naturally on the idea of imputation by iterative regression, with multiple imputation omitted from the current manuscript (and to be found in Section 5 of the article). It is necessary to underline the importance of material of the previous chapter, as well as preceding works on depth computation; only they made calculation required for depth-based imputation possible, because imputing once missing values in a data set involves computation of depth of a point w.r.t. the data set—an already demanding operation, *e.g.*, for halfspace depth—many times: (a) computing depth on numerous iterations inside optimization procedure (to find maximum conditional on existing entries), (b) this maximization procedure is repeated for each point possessing missing values, (c) the entire imputation for each such point is repeated until convergence on a data set. The reader is additionally referred to Section A.4 for an implementation note.

Chapter 3 (based on Lafaye De Micheaux et al., 2022) introduces a novel depth notion dedicated to statistical treatment of unparametrized curves. This work started with an unsuccessful attempt of application of existing multivariate functional depths to brain fibers' data stemming from the Older Australian Twins Study. Furthermore, first developments yielded only relatively meaningful practical results, though satisfying required theoretical properties. Only bringing the reference curve's halfspace probability in the denominator delivered a breakthrough in interpretation. Since such a denominator can take zero values, this substantially complicated theoretical analysis, and consequently computation algorithm. Section 3.5.3 provides a biological proof-of-concept (for a complete simulation and real-data study, see Sections 5 and 6 of the article), where with a bare eye one recognizes the difference in depth-*versus*-depth plots between dizygotic (non-identical, those who share 50% of genes) and monozygotic (those who share 100% of genes and are identical) twins. The developed curve depth has been implemented in R-package `curveDepth`, see Section A.5. Needless to emphasize the role played by exact (Section 1.2) and approximate (Section 1.4) algorithms for the halfspace depth when employing the curve depth in practice.

Chapter 4 addresses the question of anomaly detection. Anomalies are observations in rare regions of data distribution, and thus data depth constitutes a well-suited methodology for their identification. Two methods are proposed, in collaboration with the co-supervised by me PhD student Guillaume Staerman. The first one, in Section 4.1 (based on Staerman et al., 2019) extends the celebrated isolation forest (Liu et al., 2008) algorithm to the func-

tional setting. The algorithm is very flexible due to a possibility to choose a dictionary and a scalar product in the functional space, fast in implementation, and allows in practice to detect a variety of anomalies. Further, in an heuristic way, it can be seen as a data depth function (after a transformation). The second one, in Section 4.2 (based on Staerman et al., 2020), introduces a notion of functional data depth that exploits the area of the convex hulls of the functions' graphs. This new depth satisfies meaningful properties imposed on a functional depth function, provides robust estimates, and is particularly suited for detection of isolated (*i.e.*, those appearing for a very short period of time, or another functional argument) anomalies.

Chapter 5 provides insights about planned work. First two Sections 5.1 and 5.2 detail projects to be performed as PhD studies in collaboration with a (co-)supervised PhD student, each. Section 5.3 explains two more ideas to be realized. Section 5.4 concludes.

Appendix A gathers references to software packages developed to implement the methods described in this manuscript.

In hope that you find the material of this manuscript interesting, I wish you fruitful and pleasant reading.

Right below, my works are listed: publications described in this manuscript, those which are not, and reports and works in progress.

List of publications described in this manuscript:

- Mosler, K. and Mozharovskyi, P. (2022): Choosing among notions of multivariate depth statistics. *Statistical Science*, 37(3), 348–368.
- Lafaye De Micheaux, P., Mozharovskyi, P., and Vimond, M. (2021): Depth for curve data and applications. *Journal of the American Statistical Association*, 116(536), 1881–1897.
- Dyckerhoff, R., Mozharovskyi, P., and Nagy, S. (2020): Approximate computation of projection depths. *Computational Statistics and Data Analysis*, 157, 107166.
- Staerman, G., Mozharovskyi, P., and Clémençon, S. (2020): The area of the convex hull of sampled curves: a robust functional statistical depth measure. In: Chiappa, S. and Calandra, R. (eds.), *Proceedings of Machine Learning Research (AISTATS 2020)*, 108, 570–579.
- Nagy, S., Dyckerhoff, R., and Mozharovskyi, P. (2020): Uniform convergence rates for the approximated halfspace and projection depth. *Electronic Journal of Statistics*, 14(2), 3939–3975.
- Mozharovskyi, P., Josse, J., and Husson, F. (2020): Nonparametric imputation by data depth. *Journal of the American Statistical Association*, 115(529), 241–253.

- Staerman, G., Mozharovskyi, P., Clémençon, S., and D'Alché-Buc, F. (2019): Functional isolation forest. In: Lee, W. S. and Suzuki, T. (eds.), *Proceedings of Machine Learning Research (ACML 2019)*, 101, 332–347.

- Pokotylo, O., Mozharovskyi, P., and Dyckerhoff, R. (2019): Depth and depth-based classification with `R`-package `ddalpha`. *Journal of Statistical Software*, 91(5), 1–46.

- Liu, X., Mosler, K., and Mozharovskyi, P. (2019): Fast computation of Tukey trimmed regions and median in dimension $p > 2$. *Journal of Computational and Graphical Statistics*, 28(3), 682–697.

- Badunenko, O. and Mozharovskyi, P. (2016): Nonparametric frontier analysis using `Stata`. *Stata Journal*, 16(3), 550–589.

- Dyckerhoff, R. and Mozharovskyi, P. (2016): Exact computation of the halfspace depth. *Computational Statistics and Data Analysis*, 98, 19–30.

Other publications:

- Parekh, J., Parekh, S., Mozharovskyi, P., D'Alché-Buc, F., and Richard, G. (2022). Listen to interpret: Post-hoc interpretability for audio networks with NMF. *NeurIPS 2022*, in press.

- Staerman, G., Adjakossa, E., Mozharovskyi, P., Hofer, V., Sen Gupta, J., and Clémençon, S. (2022): Functional anomaly detection: a benchmark study. *International Journal of Data Science and Analytics*, in press.

- Goibert, M., Clémençon, S., Irurozki, E., and Mozharovskyi, P. (2022): Statistical depth functions for ranking distributions: definitions, statistical learning and applications. In: Camps-Valls, G., Ruiz, F. J. R., Valera , I. (eds.), *Proceedings of The Twenty Fifth International Conference on Artificial Intelligence and Statistics (AISTATS 2022)*, 151, 10376–10406.

- Parekh, J., Mozharovskyi, P., and D'Alché-Buc, F. (2021): A framework to learn with interpretation. *NeurIPS 2021.* In: Ranzato, M., Beygelzimer, A., Dauphin, Y., Liang, P.S. and Wortman Vaughan, J. (eds.), *Advances in Neural Information Processing Systems 34 (NeurIPS 2021).*

- Statzer, C., Jongsma, E., Liu, S. X., Dakhovnik, A., Wandrey, F., Mozharovskyi, P., Zülli, F., and Ewald, C. Y. (2021): Youthful and age-related matreotypes predict drugs promoting longevity. *Aging Cell*, 20, e13441.

- Staerman, G., Laforgue, P., Mozharovskyi, P., and D'Alché-Buc, F. (2021): When OT meets MoM: Robust estimation of Wasserstein distance. In: Banerjee, A. and Fukumizu, K. (eds.), *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics (AISTATS 2021)*, 130, 136–144.

- Beaudoin, V., Bloch, I., Bounie, D., Clémençon, S., D'Alché-Buc, F., Eagan, J., Maxwell, W., Mozharovskyi, P., and Parekh, J. (2020): Identifying the "right" level of explanation in a given situation. In: Saffiotti, A., Serafini, L., and Lukowicz, P. (eds.), *Proceedings of the First International Workshop on New Foundations for Human-Centered AI (NeHuAI 2020 with ECAI 2020)*, 63–66.

- Badunenko, O. and Mozharovskyi, P. (2020): Statistical inference for the Russel measure of technical efficiency. *Journal of the Operational Research Society*, 71(3), 517–527.

- Mosler, K. and Mozharovskyi, P. (2017): Fast DD-classification of functional data. *Statistical Papers*, 58(4), 1055–1089.

- Mozharovskyi, P. and Vogler, J. (2016): Composite marginal likelihood estimation of spatial autoregressive probit models feasible in very large samples. *Economics Letters*, 148, 87–90.

- Mozharovskyi, P., Mosler, K., and Lange, T. (2015): Classifying real-world data with the $DD\alpha$-procedure. *Advances in Data Analysis and Classification*, 9(3), 287–314.

- Lange, T., Mosler, K., and Mozharovskyi, P. (2014): Fast nonparametric classification based on data depth. *Statistical Papers*, 55(1), 49–69.

- Lange, T., Mosler, K., and Mozharovskyi, P. (2014): $DD\alpha$-classification of asymmetric and fat-tailed data. In: Spiliopoulou, M., Schmidt-Thieme, L., and Janning, R. (eds.), *Data Analysis, Machine Learning and Knowledge Discovery*, Springer, Berlin, 71–78.

- Lange, T. and Mozharovskyi, P. (2014): The alpha-procedure: a nonparametric invariant method for automatic classification of multi-dimensional objects. In: Spiliopoulou, M., Schmidt-Thieme, L., and Janning, R. (eds.), *Data Analysis, Machine Learning and Knowledge Discovery*, Springer, Berlin, 79–86.

Two reports and works in progress:

- Beaudouin, V., Bloch, I., Bounie, D., Clémençon, S., D'Alché-Buc, F., Eagan, J., Maxwell, W., Mozharovskyi, P., and Parekh, J. (2020): Flexible and context-specific AI explainability: A multidisciplinary approach. *Scientific report.* `arXiv:2003.07703`.

- Mozharovskyi, P. (2016): Tukey depth: linear programming and applications. *Scientific report.* `arXiv:1603.00069`.

- Mozharovskyi, P.: Anomaly detection using data depth: multivariate case.

- Malinovskaya, A., Mozharovskyi, P., and Otto, P.: Statistical monitoring of models based on artificial intelligence.

- Fojtík, V., Laketa, P., Mozharovskyi, P., and Nagy, S.: On exact computation of Turkey depth central regions.
- Staerman, G., Mozharovskyi, P., and Clémençon, S.: Affine-invariant integrated rank-weighted depth: definition, properties and finite sample analysis.
- Staerman, G., Mozharovskyi, P., Clémençon, S., and D'Alché-Buc, F.: Depth-based pseudo-metrics between probability distributions.
- Ivanovs, J. and Mozharovskyi, P.: Distributionally robust halfspace depth.
- Patilea, V., Rouviere, L., and Mozharovskyi, P.: Simple tests of stability for parametric regressions.

Chapter 1

Computation

1.1 Introduction to multivariate data depth

1.1.1 Definition of data depth

In general, a (d-variate) *depth function* is a function $D : (\boldsymbol{z}, P) \mapsto [0, 1]$, for $\boldsymbol{z} \in \mathbb{R}^d$ and P from some class $\mathcal{P}$ of d-variate probability distributions, that satisfies several postulates regarding invariance, monotonicity, convexity and continuity. $D(\boldsymbol{z}|X)$ will be written in place of $D(\boldsymbol{z}|P)$, where X denotes a random vector distributed as P.

An often-quoted set of such postulates has been given by Liu (1990) for simplicial depth and by Zuo and Serfling (2000) for general depth functions. Here a slightly terser one is used, which is due to Dyckerhoff (2002): D is a depth function if it is invariant against $\mathbb{R}^d$-transformations in some class $\mathcal{T}$, null at infinity, monotone decreasing on rays from its maximum, and upper continuous. Formally, for $\boldsymbol{z} \in \mathbb{R}^d$ and $P \in \mathcal{P}$,

- **$\mathcal{T}$-Invariance:** $D(T(\boldsymbol{z})|T(X)) = D(\boldsymbol{z}|X)$ for all $T \in \mathcal{T}$,
- **Null at infinity:** $\lim_{\|\boldsymbol{z}\|\to\infty} D(\boldsymbol{z}|X) = 0$.
- **Monotone on rays:** If a point $\boldsymbol{z}^*$ has maximal depth, that is $D(\boldsymbol{z}^*|X) = \max_{\boldsymbol{z}\in\mathbb{R}^d} D(\boldsymbol{z}|X)$ then for any $\boldsymbol{r}$ in the unit sphere of $\mathbb{R}^d$ the function $\gamma \mapsto D(\boldsymbol{z}^* + \gamma\boldsymbol{r}|X)$ does not increase with $\gamma > 0$.
- **Upper semicontinuous:** The upper level sets $D^\alpha = \{\boldsymbol{z} \in \mathbb{R}^d | D(\boldsymbol{z}|X) \geq \alpha\}$ are closed for all $\alpha \in [0, 1]$.

Any point $\boldsymbol{z}^*$ that has maximum depth is called a *median*. The postulates imply that the level sets (= *central regions*) D^α, $\alpha \in]0, 1]$, are bounded. Monotonicity on rays means that they are starshaped about $\boldsymbol{z}^*$, hence nested. Moreover, if X is centrally symmetric about some $\boldsymbol{z}^* \in \mathbb{R}^d$, then any depth function yields $\boldsymbol{z}^*$ as a median. Recall that X is *centrally symmetric* about $\boldsymbol{z}^*$ if $X - \boldsymbol{z}^*$ has the same distribution as $\boldsymbol{z}^* - X$. If the level sets are convex, D is a *quasi-concave depth function*. Mostly, $\mathcal{T}$ is specified as the class of affine transformations of $\mathbb{R}^d$, but other classes of transformations are possible and of practical interest.

Central regions are sometimes parameterized by their probability content,

$$D_\beta(X) = \bigcap_{\alpha \in A(\beta)} D^\alpha(X)\,, \quad \text{where} \quad A(\beta) = \{\alpha : P[D^\alpha(X)] \geq \beta\}\,. \tag{1.1}$$

If P is the empirical distribution on a set $\boldsymbol{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$ of data points, the depth function is mentioned as a *multivariate empirical data depth* and written $D(\boldsymbol{z}|\boldsymbol{X})$. While in Section 1.3 explicit ties-excluding assumption will be made, the rest of the manuscript allows for tied observations. Nevertheless, throughout the subsequent material let us stick to this unique even if somewhat abusing set-kind notation for $\boldsymbol{X}$ since no ambiguity shall be caused throughout the manuscript.

Well-known examples of depth functions are the *halfspace depth* (Tukey, 1975), which is also called *Tukey* or *location depth*, the *zonoid depth* (Koshevoy and Mosler, 1997), the *spatial* (Serfling, 2002), *projection* (Liu, 1992, Zuo and Serfling, 2000), *simplicial* (Liu, 1990) and *simplicial volume* (Oja, 1983) *depths*. A more recent notion is the *β-skeleton depth* (Yang and Modarres, 2018), which includes the *lens depth* (Liu and Modarres, 2011) and the *spherical depth* (Elmore et al., 2006) as special cases.

By exploiting geometry of data (Cascos, 2009), the depth function is fully non-parametric and thus does not rely on assumptions about the data generating process and satisfies desirable invariance properties (Zuo and Serfling, 2000) (see also Liu, 1990). Further, it is robust to both outliers and heavy tailed distributions (Donoho and Gasko, 1992). By dint of these advantages, data depth is used in a variety of tasks as a generalization of quantiles and ranks in higher dimensions and as an alternative to the distribution function (Chernozhukov et al., 2017, Hallin et al., 2021). Nowadays, these benefits make data depth vital for many applications of data analysis (Liu et al., 1999): supervised (Lange et al., 2014b) and unsupervised (Jörnsten, 2004) machine learning and anomaly detection (Staerman et al., 2020), robust optimization (Bazovkin and Mosler, 2015), financial risk assessment (Cascos, 2009), statistical quality control (Liu and Singh, 1993), extreme value theory (Einmahl et al., 2015), imputation of missing data (M. et al., 2020), to name but a few.

Below, in Section 1.1.2, notions of multivariate depths are reviewed, which are relevant for understanding the material of the rest of this manuscript.

Definition of functional depth and it's notions relevant for Chapters 3 and 4 are gathered in Section 3.1.

1.1.2 Depth notions used in this manuscript

Many depth notions have been proposed in the literature. Not all of them satisfy the above postulates. In this subsection, six notions of multivariate depth function are reviewed which will be used in subsequent sections of this chapter and useful for deeper understanding of the following chapters of this manuscript. For further depth notions and a general reference on multivariate data depth, the reader is referred to Mosler and M. (2022).

Among the definitions below, halfspace depth (1.3) is the most important notion throughout this manuscript, and especially in this chapter dedicated mainly to its computation.

Together with Mahalanobis (1.2) and zonoid (1.8) depths, it appears particularly important also in Section 1.4 and Chapter 2. Projection (1.4) and asymmetric projection (1.5) depths are referred to substantially in Section 1.4. Simplicial depth (1.6) plays an important role for several functional depth notions (see Chapter 3).

Definition 1.1 (Mahalanobis depth)

$$D_{\text{Mah}}(\boldsymbol{z}|X) = \left(1 + ||\boldsymbol{z} - \boldsymbol{\mu}_X||^2_{\boldsymbol{\Sigma}_X}\right)^{-1} \tag{1.2}$$

is called (moment) Mahalanobis depth *(Mahalanobis, 1936). Here, $\boldsymbol{\mu}_X$ and $\boldsymbol{\Sigma}_X$ denote the expectation vector and the covariance matrix of X, and $||\boldsymbol{y}||^2_{\boldsymbol{\Sigma}_X} = \boldsymbol{y}^T\boldsymbol{\Sigma}_X^{-1}\boldsymbol{y}$ is the* Mahalanobis norm *of $\boldsymbol{y} \in \mathbb{R}^d$.*

Definition 1.2 (Halfspace depth)

$$D_{\text{H}}(\boldsymbol{z}|X) = \inf\{\mathbb{P}_X[X \in H] : H \text{ closed halfspace}, \boldsymbol{z} \in H\}\,. \tag{1.3}$$

is called halfspace (=location=Tukey) depth *(Tukey, 1975, Donoho and Gasko, 1992).*

Definition 1.3 (Projection depth) Projection depth *(Liu, 1992, Zuo and Serfling, 2000) is defined as:*

$$D_{\text{Proj}}(\boldsymbol{z}|X) = \left(1 + \sup_{\boldsymbol{u}\in\mathbb{S}^{d-1}} \frac{|\langle \boldsymbol{u}, \boldsymbol{z}\rangle - med(\langle \boldsymbol{u}, X\rangle)|}{MAD(\langle \boldsymbol{u}, X\rangle)}\right)^{-1}, \tag{1.4}$$

where $med(Y)$ denotes the median of a univariate random variable Y, and $MAD(Y) = med(|Y - med(Y)|)$ its median absolute deviation from the median.

Definition 1.4 (Asymmetric projection depth) Asymmetric projection depth *(Dyckerhoff, 2004) is defined as*

$$D_{\text{asProj}}(\boldsymbol{z}|X) = \min_{\boldsymbol{u}\in\mathbb{S}^{d-1}} \left(1 + \frac{(\langle \boldsymbol{u}, \boldsymbol{z}\rangle - med(\langle \boldsymbol{u}, X\rangle))_+}{MAD^+(\langle \boldsymbol{u}, X\rangle)}\right)^{-1}, \tag{1.5}$$

with $(a)_+ = \max\{a, 0\}$ being the positive part of a and MAD^+ being the median of the positive deviations from the median.

It has been introduced to compensate for the fact that projection depth is always symmetric around its deepest point.

Definition 1.5 (Simplicial depth) Simplicial depth *(Liu, 1990) is defined as follows:*

$$D_{\text{Sim}}(\boldsymbol{z}|X) = \mathbb{P}_X\left[\boldsymbol{z} \in conv(\{X_1, \ldots, X_{d+1}\})\right], \tag{1.6}$$

where $X_1, \ldots, X_{d+1}$ are i.i.d. copies of X, and conv means convex hull.

Definition 1.6 (Zonoid depth) *For* $0 < \alpha \leq 1$, *let*

$$D^{\alpha}_{\mathrm{Zon}}(X) = \left\{\mathbb{E}_X[X\, g(X)] \ : \ g : \mathbb{R}^d \to [0, 1/\alpha] \ \text{ measurable and } \mathbb{E}_X[g(X)] = 1\right\} \tag{1.7}$$

be the zonoid α-region *of* X. *For* $\alpha = 0$ *set* $D^{0}_{\mathrm{Zon}}(X) = \mathbb{R}^d$. *The* zonoid depth *(Koshevoy and Mosler, 1997, Mosler, 2002) is defined as*

$$D_{\mathrm{Zon}}(\boldsymbol{z}|X) = \sup\{\alpha : \boldsymbol{z} \in D^{\alpha}_{\mathrm{Zon}}(X)\} \tag{1.8}$$

For a data set $\boldsymbol{X}$, *the zonoid depth then is calculated as*

$$D_{\mathrm{Zon}}(\boldsymbol{z}|\boldsymbol{X}) = \sup\left\{\alpha \ : \ \alpha\lambda_i \leq 1/n, \boldsymbol{z} = \sum_{i=1}^{n} \lambda_i \boldsymbol{x}_i, \ \sum_{i=1}^{n} \lambda_i = 1, \ \lambda_i \geq 0 \ \forall i\right\}. \tag{1.9}$$

For the rest of the depth notions, in their empirical versions, empirical measure is substituted for X.

1.1.3 A word on computation

The question of computation of data depth is important, if not crucial, for evolution of this domain. Even having undergone substantial theoretical developments, data depth has been long underemployed in applications, thus lacking interest of practitioners and incentives for further theoretical results. The computational load needed for depth calculation cannot be neglected, gaining weight with the number n of data and, even more, with dimension d. Therefore, until recently, due to computational infeasibility the use of most depth statistics was limited to small n and d in applications, and the choice of a proper depth statistic in practice was restricted to very few notions. For example, first accessible algorithm for computation of projection depth for $d \geq 3$ was developed in 2014 (Liu and Zuo, 2014b) and for halfspace depth in 2016 (Dyckerhoff and M., 2016). Depth trimmed regions for $d \geq 3$ (except the trivial cases, *e.g.*, Mahalanobis depth) were first computed in 2009 (Mosler et al., 2009), for zonoid depth.

Fortunately, in the last few years data depths have been implemented in several existing software packages. In particular, R-package `ddalpha` (Pokotylo et al., 2019, 2020) implements exact procedures for all above mentioned (and other) depth notions, except projection depth. In addition, approximate algorithms are provided for the halfspace, projection, simplicial, and simplicial volume depths. Packages `depth` (Genest et al., 2019), `DepthProc` (Kosiorowski and Zawadzki, 2019, Zawadzki et al., 2020), `fda.usc` (Febrero-Bande and Oviedo de la Fuente, 2012), `mrfDepth` (Segaert et al., 2020) implement a number of depth notions as well.

Mahalanobis depth is easily coded by hand in any programming language; ready-to-use implementations are also found in R-packages `DepthProc` and `fda.usc`. R-package `DepthProc` suggests an implementation of $\mathbb{L}_p$ depth. Halfspace depth can be computed exactly for $d \leq 3$ (and approximately) in any dimension with R-packages `depth` and `mrfDepth`, and only approximately with R-packages `DepthProc` and `fda.usc`. Exact projection depth is

computed with `MATLAB`-package `CompPD` (Liu and Zuo, 2015), while approximate procedures are included in `R`-packages `DepthProc`, `fda.usc`, and `mrfDepth`. Exact simplicial depth for $d = 2$ is calculated with `R`-packages `depth`, `fda.usc`, and `mrfDepth`. Exact simplicial volume depth is also computed using `R`-package `depth`. Spatial depth is implemented in `R`-package `depth.plot` (Mahalanobish and Karmakar, 2015).

Existing software allows for computation of depth regions as well. Tukey regions are calculated with `R`-packages `modQR` (Šiman and Boček, 2019) or `TukeyRegion` (Liu et al., 2019, including the Tukey median) and `Octave`-package `modQR` (Boček and Šiman, 2016). Trimmed regions and median of the projection depth are computed using the before mentioned `MATLAB`-package `CompPD`. The Oja median is determined using the `R`-package `OjaNP` (Fischer et al., 2020). `R`-package `WMTregions` (Bazovkin and Mosler, 2012) computes zonoid regions (see also Mosler et al., 2009), as well as trimmed regions for the entire family of weighted-mean depths (Dyckerhoff and Mosler, 2011). Onion depth regions are constructed using the QHULL implementation of the `R`-package `geometry` (Habel et al., 2019).

Obviously, this overview cannot be complete. Moreover, the packages are continuously modified by their authors.

Following sections of this chapter present author's contributions to computation of data depths. First, Section 1.2 introduces the entire family of algorithms for computation of the halfspace depth. Second, in Section 1.3 a fast algorithm (together with its fast combinatorial counterpart) for computation of halfspace trimmed regions is developed and applied to fast computation of the halfspace median. Since computational time complexity of exact algorithms for certain depth notions (including halfspace depth) grows exponentially with space dimension, a framework for approximation for the class of projection depths (introduced by Dyckerhoff, 2004) is proposed in Section 1.4. Finally, though under unrealistic assumptions, uniform convergence rates on random approximation of halfspace depth are derived in Section 1.5.

These developments caused further positive consequences. In particular, the ability to efficiently compute data depth allowed for its application to imputation of missing data (detailed in Chapter 2 of this manuscript) and creation of the novel notion of depth for curves (described in Chapter 3). For example, when imputing missing values of a data set, imputation of each point having missing values requires thousands of depth computations during iterations of an optimization routine. A procedure very resembling the one for halfspace depth computation constitutes the basis for computation of the Tukey curve depth (Chapter 3). Based on similar ideas, efficient algorithms for non-parametric frontier analysis were constructed, see Badunenko and M. (2016) for details and Section A.3 for short information.

Procedures from Sections 1.2 and 1.3 are implemented in `R`-packages `ddalpha` and `TukeyRegion`, respectively; see Sections A.1 and A.2 for more information on this software. Implementation of algorithms from Section 1.4 in `R`-package `ddalpha` is ongoing. Both exact and approximate computation of different depth notions is currently being implemented in a `Pyhton` library, to address the machine learning community.

The following definition is useful when computing halfspace depth and its regions.

Definition 1.7 (General position) *A data set* $\boldsymbol{X} = \{\boldsymbol{x}_1, ..., \boldsymbol{x}_n\}$ *in* $\mathbb{R}^d$ *is said to be in general position if any affine subspace of dimension* k, $1 \leq k < d$, *contains at most* k *of the data points* $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$.

Before jumping into subsequent sections, next subsection gathers interesting material on connections of the halfspace depth to further mathematical tools.

1.1.4 Connections of the halfspace depth

Due to its indicator-loss nature, the halfspace depth (1.3) is known to be connected to a number of problems. Although just the smallest portion of the points to be cut off by a closed halfspace is required as an output, in many algorithms the boundary hyperplane is found or can be restored based on the output information. For shortness, let $\boldsymbol{R} = \arg\min_{\boldsymbol{r}\in\mathbb{S}^{d-1}} \#\{i \ : \ \boldsymbol{x}_i^\top \boldsymbol{r} \geq \boldsymbol{z}^\top \boldsymbol{r}, \ \boldsymbol{x}_i \in \boldsymbol{X}\}$ be the set of directions $\boldsymbol{r} \subset \mathbb{S}^{d-1}$ each achieving $\frac{1}{n}\#\{i \ : \ \boldsymbol{x}_i^\top \boldsymbol{r} \geq \boldsymbol{z}^\top \boldsymbol{r}, \ \boldsymbol{x}_i \in \boldsymbol{X}\} = D_{\mathrm{H}}(\boldsymbol{z}|\boldsymbol{X})$.

Densest hemisphere. The problem of computing the halfspace depth is invariant more than just in the affine way. Thus shifting $\boldsymbol{X}$ to get $\boldsymbol{z}$ in the origin and projecting $\boldsymbol{X}$ onto the unit sphere $\mathbb{S}^{d-1}$ after that (a non-affine transformation), changes neither the value of the depth $D_{\mathrm{H}}(\boldsymbol{z}|\boldsymbol{X})$ nor the set of optimal normals to separating hyperplanes $\boldsymbol{R}$, *i.e.* $D_{\mathrm{H}}(\boldsymbol{z}|\boldsymbol{X}) = D_{\mathrm{H}}(\boldsymbol{0}|\boldsymbol{Y})$ as well as the optimal argument set with $\boldsymbol{Y} = \{\frac{\boldsymbol{x}_i - \boldsymbol{z}}{\|\boldsymbol{x}_i - \boldsymbol{z}\|} \ : \ i = 1, ..., n\} \subset \mathbb{S}^{d-1}$. The last one corresponds to the *open densest hemisphere* problem shown by Johnson and Preparata (1978) to be of non-polynomial time complexity, namely $O(n^{d-1} \log n)$ if dimension d is fixed.

Classification. By a trivial modification, the task of *supervised binary linear classification* can be narrowed down to finding an optimal argument $\boldsymbol{r}$ from (1.3), see also Ghosh and Chaudhuri (2005). Indeed, regard $\boldsymbol{X}_1 = \{\boldsymbol{x}_1, ..., \boldsymbol{x}_{n_1}\}$ and $\boldsymbol{X}_2 = \{\boldsymbol{x}_{n_1+1}, ..., \boldsymbol{x}_{n_1+n_2}\}$ being two training classes in $\mathbb{R}^d$, and let $\boldsymbol{Y} = \{\boldsymbol{x}_i - \boldsymbol{x}_j \ : \ i = 1, ..., n_1, \ j = n_1 + 1, ..., n_1 + n_2\}$. When minimizing empirical risk of a linear classifier, one is interested in a direction $\boldsymbol{r} \in \mathbb{S}^{d-1}$ in projection on which possibly many differences $\boldsymbol{x}_i - \boldsymbol{x}_j$ have the same sign, or in other words, as many (few) as possible points from $\boldsymbol{Y}$ lie on the same side of a hyperplane through $\boldsymbol{0}$. The later holds for any element of $\boldsymbol{R}$.

Regression depth. Rousseeuw and Hubert (1999) define data depth for a liner regression model based on the notion of nonfit: Given regression input $\boldsymbol{X} = \{\boldsymbol{x}_1, ..., \boldsymbol{x}_n\}$ in $\mathbb{R}^d$ and output $\boldsymbol{y} = \{y_1, ..., y_n\}$ in $\mathbb{R}$, a fit $\boldsymbol{b} = (b_0, b_1, ..., b_d)$ is called a nonfit if there exists an affine hyperplane in the input space not containing any point from $\boldsymbol{X}$ such that all the points from $\boldsymbol{X}$ lying on its same side have residuals strictly of the same sign. Regression depth of a fit $\boldsymbol{b} \in \mathbb{R}^{d+1}$ is the smallest number of observations to be removed from $\boldsymbol{X}$ sufficient for $\boldsymbol{b}$ to become a nonfit. Given a fit $\boldsymbol{b} \in \mathbb{R}^{d+1}$, after splitting $\boldsymbol{X}$ into two classes on the basis of the sign of residuals (and acting as in the preceding paragraph), regression depth is given by the empirical risk in the projection on any element of $\boldsymbol{R}$.

Maximum feasible subsystem. Again, let $\boldsymbol{Y} = \{\boldsymbol{y}_i = \frac{\boldsymbol{x}_i - \boldsymbol{z}}{\|\boldsymbol{x}_i - \boldsymbol{z}\|} \ : \ i = 1, ..., n\} \subset \mathbb{S}^{d-1}$. First consider the case $d = 2$, where $\boldsymbol{Y}$ lies on the unit circle. Here $\boldsymbol{y}_i$ defines an open halfcircle, in which each point defines a closed halfspace with the origin on its boundary and never

containing $\boldsymbol{y}_i$. Then the task of computation of the halfspace depth narrows down to finding a point on the unit circle contained in the largest number of these halfcircles; the set of such points coincides with $\boldsymbol{R}$. Extending this logic to higher dimensions (Bremner et al., 2008) leads to (two instances of) the *maximum feasible subsystem* problem in $\mathbb{R}^{d-1}$. For $d > 2$, in the same way, each $\boldsymbol{y}_i$ defines an open halfspace in which each element (=point $\in \mathbb{R}^d$) is a normal to a hyperplane defining a halfspace with the origin on its boundary and not containing $\boldsymbol{y}_i$. One is interested in a point lying in the intersection of the highest number of these halfspaces. Let $H_+ = \{(x_1, ..., x_d)^\top : x_d = 1\}$ ($H_- = \{(x_1, ..., x_d)^\top : x_d = -1\}$) be a positive (respectively negative) hyperplane. For each $\boldsymbol{y}_i$, intersection of such open halfspace with H_+ yields an open halfspace in H_+ of dimension $\mathbb{R}^{d-1}$; the same holds for H_-. Then, the maximum number such halfspaces either in H_+ or in H_- having nonempty intersection divided through n equals the halfspace depth. The two maximum feasible subsystem problems consist of linear inequalities $\boldsymbol{x}\frac{\boldsymbol{y}_{i,(1,\dots,d-1)}}{\|\boldsymbol{y}_{i,(1,\dots,d-1)}\|} \leq -\tan(\frac{\pi}{2} - \arccos \boldsymbol{y}_{i,(d)})$ for H_+ and or $\boldsymbol{x}\frac{\boldsymbol{y}_{i,(1,\dots,d-1)}}{\|\boldsymbol{y}_{i,(1,\dots,d-1)}\|} \leq \tan(\frac{\pi}{2} - \arccos \boldsymbol{y}_{i,(d)})$ for H_-. (Note that due to the general position assumption there is no difference between open and closed halfspaces, and the equator $\{(x_1, ..., x_d)^\top : x_d = 0\}$ should not be searched through because $\boldsymbol{R}$ has nonzero volume on $\mathbb{S}^{d-1}$.)

Quantile regression. Hallin et al. (2010) establish connection of the halfspace depth to the linear programming via quantile regression (see also Koenker and Basset, 1978) and exploit this to compute halfspace depth regions; for more details here, see the beginning of Section 1.3.

1.2 Computation of the halfspace depth

In this section, a theoretical framework for computing the halfspace depth is suggested, which yields a whole class of algorithms. For $k \in \{1, \dots, d-1\}$, consider a tuple of k (out of n) data points, which together with $\boldsymbol{z}$ span an affine subspace of dimension k. For each such tuple, the data are projected onto the corresponding orthogonal complement, and the halfspace depth is computed as the sum of the depth in these two orthogonal subspaces. Further, for some fixed k, the halfspace depth is obtained as the minimum of the depths over all such tuples. All proposed algorithms are capable of dealing with data that are not in general position and even with ties.

In what follows, first the theoretical framework is developed in Section 1.2.1, leading to the main result stated in Theorem 1.2, which yields the above mentioned class of algorithms. Further, three algorithms for $k = d-1, d-2, 1$ are presented in Section 1.2.2. In Section 1.2.3, implementation issues are discussed and a speed comparison of the three algorithms are provided for data in general and non-general position.

The following notation will be used across this section. The number of elements of a set I is denoted by $\#I$ and the complement of a set I by I^c. The linear hull of some points $\boldsymbol{x}_1, \dots, \boldsymbol{x}_k \in \mathbb{R}^d$ is denoted by $\text{span}(\boldsymbol{x}_1, \dots, \boldsymbol{x}_k)$. For a subspace U of $\mathbb{R}^d$, denote the orthogonal complement of U by $U^\perp$.

1.2.1 Theoretical guarantees

The *halfspace depth* of a point $\boldsymbol{z} \in \mathbb{R}^d$ w.r.t. n data points $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n \in \mathbb{R}^d$ is defined by

$$D_{\mathrm{H}}(\boldsymbol{z} \,|\, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_n) = \frac{1}{n} \min_{\boldsymbol{p} \neq \boldsymbol{0}} \#\{i \,|\, \boldsymbol{p}^\top \boldsymbol{x}_i \geq \boldsymbol{p}^\top \boldsymbol{z}\} \,. \tag{1.10}$$

The halfspace depth of a point $\boldsymbol{z}$ is therefore simply the minimum fraction of data points $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$ contained in a closed halfspace containing $\boldsymbol{z}$. For simplicity, the shorter notation $D_{\mathrm{H}}(\boldsymbol{z} \,|\, \boldsymbol{X})$ will be used.

The halfspace depth is affine invariant, in particular it is location invariant. Therefore,

$$D_{\mathrm{H}}(\boldsymbol{z} \,|\, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_n) = D_{\mathrm{H}}(\boldsymbol{0} \,|\, \boldsymbol{x}_1 - \boldsymbol{z}, \ldots, \boldsymbol{x}_n - \boldsymbol{z}) \,,$$

which shows that one can restrict w.l.o.g. to the case that the halfspace depth of the origin has to be computed. Further, it will be useful to consider the integer version of the halfspace depth,

$$nD_{\mathrm{H}}(\boldsymbol{z} \,|\, \boldsymbol{X}) = n \cdot D_{\mathrm{H}}(\boldsymbol{z} \,|\, \boldsymbol{X}) \in \mathbb{N}_0 \,.$$

In this section let us assume that $\boldsymbol{0} \notin \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$. If some of the data points are equal to $\boldsymbol{0}$, then these points are removed from the data set and their number is simply added to the (integer) halfspace depth of the origin w.r.t. the remaining points. Let us further assume that $\operatorname{span}(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n) = \mathbb{R}^d$. If this is not the case, *i.e.*, if the data points are contained in some k-dimensional subspace, $k < d$, then the data points are mapped to $\mathbb{R}^k$ by a linear transformation of rank k, and the algorithms are then applied to the transformed data points.

Under the assumptions from above, the (integer) halfspace depth can be written as

$$nD_{\mathrm{H}}(\boldsymbol{0} \,|\, \boldsymbol{X}) = \min_{\boldsymbol{p} \neq \boldsymbol{0}} \#\{i \,|\, \boldsymbol{p}^\top \boldsymbol{x}_i \geq 0\} \,.$$

A vector $\boldsymbol{p} \neq \boldsymbol{0}$ is called *optimal for the data set* $\boldsymbol{X}$ if

$$nD_{\mathrm{H}}(\boldsymbol{0} \,|\, \boldsymbol{X}) = \#\{i \,|\, \boldsymbol{p}^\top \boldsymbol{x}_i \geq 0\} \,.$$

Let us use the following notation: If $\boldsymbol{p} \neq \boldsymbol{0}$, then

$$I_{\boldsymbol{p}}^+ = \{i \,|\, \boldsymbol{p}^\top \boldsymbol{x}_i > 0\} \,, \quad I_{\boldsymbol{p}}^0 = \{i \,|\, \boldsymbol{p}^\top \boldsymbol{x}_i = 0\} \,, \quad I_{\boldsymbol{p}}^- = \{i \,|\, \boldsymbol{p}^\top \boldsymbol{x}_i < 0\} \,,$$

and the corresponding cardinalities are denoted by

$$n_{\boldsymbol{p}}^+ = \# I_{\boldsymbol{p}}^+ \,, \quad n_{\boldsymbol{p}}^0 = \# I_{\boldsymbol{p}}^0 \,, \quad n_{\boldsymbol{p}}^- = \# I_{\boldsymbol{p}}^- \,.$$

With this notation

$$nD_{\mathrm{H}}(\boldsymbol{0} \,|\, \boldsymbol{X}) = \min_{\boldsymbol{p} \neq \boldsymbol{0}} (n_{\boldsymbol{p}}^+ + n_{\boldsymbol{p}}^0) = n - \max_{\boldsymbol{p} \neq \boldsymbol{0}} n_{\boldsymbol{p}}^- \,.$$

For a subset I of indices, $\boldsymbol{X}_I$ denotes the data set $(\boldsymbol{x}_i)_{i \in I}$ of all data points with indices in I. If the data are not in general position, then the linear hull of some data points $\boldsymbol{x}_{i_1}, \ldots, \boldsymbol{x}_{i_k}$ may contain additional data points. For a set $I = \{i_1, \ldots, i_k\}$ of indices, let us denote by

$$I^* = \{j \,|\, \boldsymbol{x}_j \in \operatorname{span}(\boldsymbol{x}_{i_1}, \ldots, \boldsymbol{x}_{i_k})\}$$

the set of all indices j such that $\boldsymbol{x}_j$ is contained in the linear hull of $\boldsymbol{x}_{i_1}, \ldots, \boldsymbol{x}_{i_k}$. Obviously, $I \subset I^*$. The cardinality of I^* is denoted by n_{I^*}.

Proposition 1.1 *(Dyckerhoff and M., 2016) If $\boldsymbol{p} \neq 0$ is optimal for $\boldsymbol{X}$, then $I^0_{\boldsymbol{p}} = \varnothing$, i.e., no data points lie on the boundary of the closed halfspace defined by $\boldsymbol{p}$.*

The following lemma is needed.

Lemma 1.1 *(Dyckerhoff and M., 2016) If $\boldsymbol{p} \neq \boldsymbol{0}$ and $0 \leq \dim \operatorname{span}(\boldsymbol{X}_{I^0_{\boldsymbol{p}}}) = l < d-1$, then there exists $\boldsymbol{x}_{i_0} \notin \operatorname{span}(\boldsymbol{X}_{I^0_{\boldsymbol{p}}})$ and a vector $\tilde{\boldsymbol{p}} = \boldsymbol{p} + \boldsymbol{q} \neq \boldsymbol{0}$ with $\boldsymbol{q} \in \operatorname{span}(\boldsymbol{X}_{I^0_{\boldsymbol{p}}}, \boldsymbol{x}_{i_0})$ such that*

$$I^0_{\tilde{\boldsymbol{p}}} = \left[I^0_{\boldsymbol{p}} \cup \{i_0\}\right]^*, \quad I^+_{\boldsymbol{p}} \subset I^+_{\tilde{\boldsymbol{p}}} \cup I^0_{\tilde{\boldsymbol{p}}}, \quad I^-_{\boldsymbol{p}} \subset I^-_{\tilde{\boldsymbol{p}}} \cup I^0_{\tilde{\boldsymbol{p}}}.$$

Further, $\dim \operatorname{span}(\boldsymbol{X}_{I^0_{\tilde{\boldsymbol{p}}}}) = l + 1$.

This means that if one changes $\boldsymbol{p}$ to $\tilde{\boldsymbol{p}}$, then at least one of the data points that were contained in one of the open halfspaces defined by $\boldsymbol{p}$ is now on the boundary hyperplane defined by $\tilde{\boldsymbol{p}}$, whereas no data points did change sides.

Proposition 1.2 *(Dyckerhoff and M., 2016) If $\boldsymbol{p} \neq 0$ is optimal for $\boldsymbol{X}$, then for every k, $1 \leq k < d$, there are k linearly independent data points $\boldsymbol{x}_{i_1}, \ldots, \boldsymbol{x}_{i_k}$ and a vector $\tilde{\boldsymbol{p}} = \boldsymbol{p} + \boldsymbol{q} \neq \boldsymbol{0}$ where $\boldsymbol{q} \in \operatorname{span}(\boldsymbol{x}_{i_1}, \ldots, \boldsymbol{x}_{i_k})$ such that*

$$I^0_{\tilde{\boldsymbol{p}}} = \{i_1, \ldots, i_k\}^*, \quad I^+_{\boldsymbol{p}} \subset I^+_{\tilde{\boldsymbol{p}}} \cup I^0_{\tilde{\boldsymbol{p}}}, \quad I^-_{\boldsymbol{p}} \subset I^-_{\tilde{\boldsymbol{p}}} \cup I^0_{\tilde{\boldsymbol{p}}}.$$

Proposition 1.3 *(Dyckerhoff and M., 2016) Let $\boldsymbol{x}_{i_1}, \ldots, \boldsymbol{x}_{i_k}$ be linearly independent and $\boldsymbol{p} \neq \boldsymbol{0}$ such that $I = \{i_1, \ldots, i_k\} \subset I^0_{\boldsymbol{p}}$. Then,*

$$nD_H(\boldsymbol{0} \,|\, \boldsymbol{X}) \leq (n^+_{\boldsymbol{p}} + n^0_{\boldsymbol{p}}) - \left(n_{I^*} - nD_H(\boldsymbol{0} \,|\, \boldsymbol{X}_{I^*})\right).$$

Proposition 1.4 *(Dyckerhoff and M., 2016) For every k, $1 \leq k < d$, there is a set $I = \{i_1, \ldots, i_k\}$ of indices and a vector $\tilde{\boldsymbol{p}} \neq \boldsymbol{0}$ such that $\boldsymbol{x}_{i_1}, \ldots, \boldsymbol{x}_{i_k}$ are linearly independent, $I^0_{\tilde{\boldsymbol{p}}} = I^*$, and*

$$nD_H(\boldsymbol{0} \,|\, \boldsymbol{X}) \geq (n^+_{\tilde{\boldsymbol{p}}} + n^0_{\tilde{\boldsymbol{p}}}) - \left(n_{I^*} - nD_H(\boldsymbol{0} \,|\, \boldsymbol{X}_{I^*})\right).$$

For the following denote by $\mathcal{L}_k$ the set of all subsets I of order k of $\{1, \ldots, n\}$ such that the points $(\boldsymbol{x}_i)_{i \in I}$ are linearly independent.

Theorem 1.1 *(Dyckerhoff and M., 2016) For each k such that $1 \leq k < d$ it holds that*

$$nD_H(\boldsymbol{0} \,|\, \boldsymbol{X}) = \min_{I \in \mathcal{L}_k} \left[\left(\min_{\boldsymbol{p} \in \boldsymbol{X}_I^\perp \setminus \{\boldsymbol{0}\}} \left(n^+_{\boldsymbol{p}} + n^0_{\boldsymbol{p}}\right) \right) - \left(n_{I^*} - nD_H(\boldsymbol{0} \,|\, \boldsymbol{X}_{I^*})\right) \right].$$

Let us now show how

$$\min_{\boldsymbol{p} \in \boldsymbol{X}_I^\perp \setminus \{\boldsymbol{0}\}} (n^+_{\boldsymbol{p}} + n^0_{\boldsymbol{p}})$$

can be computed as the halfspace depth of a projection of the data points.

Let I be a subset of $\{1, \ldots, n\}$ of order k such that the data points $\boldsymbol{x}_i$, $i \in I$, are linearly independent. Let further $\boldsymbol{a}_1, \ldots, \boldsymbol{a}_{d-k}$ be a basis of the orthogonal complement of $\operatorname{span}(\boldsymbol{x}_{i_1}, \ldots, \boldsymbol{x}_{i_k})$ and $\boldsymbol{A}_I$ the matrix whose columns are the $\boldsymbol{a}_i$. Thus, every vector

$\boldsymbol{p} \in \boldsymbol{X}_I^\perp$ is a linear combination of $\boldsymbol{a}_1, \ldots, \boldsymbol{a}_{d-k}$, *i.e.*, $\boldsymbol{p} = \boldsymbol{A}_I \tilde{\boldsymbol{p}}$ for some $\tilde{\boldsymbol{p}} \in \mathbb{R}^{d-k}$, and the map $\mathbb{R}^{d-k} \to \boldsymbol{X}_I^\perp$, $\tilde{\boldsymbol{p}} \mapsto \boldsymbol{A}_I \tilde{\boldsymbol{p}} =: \boldsymbol{p}$, is a bijection. Since

$$\tilde{\boldsymbol{p}}^\top (\boldsymbol{A}_I^\top \boldsymbol{x}_i) \geq 0 \quad \iff \quad (\tilde{\boldsymbol{p}}^\top \boldsymbol{A}_I^\top) \boldsymbol{x}_i \geq 0 \quad \iff \quad \boldsymbol{p}^\top \boldsymbol{x}_i \geq 0 \,,$$

one can conclude

$$\begin{aligned} \min_{\boldsymbol{p} \in \boldsymbol{X}_I^\perp \setminus \{\boldsymbol{0}\}} (n_{\boldsymbol{p}}^+ + n_{\boldsymbol{p}}^0) &= \min_{\boldsymbol{p} \in \boldsymbol{X}_I^\perp \setminus \{\boldsymbol{0}\}} \#\{i \,|\, \boldsymbol{p}^\top \boldsymbol{x}_i \geq 0\} \\ &= \min_{\tilde{\boldsymbol{p}} \in \mathbb{R}^{d-k} \setminus \{\boldsymbol{0}\}} \#\{i \,|\, \tilde{\boldsymbol{p}}^\top (\boldsymbol{A}_I^\top \boldsymbol{x}_i) \geq 0\} \\ &= n D_{\mathrm{H}}(\boldsymbol{0} \,|\, \boldsymbol{A}_I^\top \boldsymbol{X}) \,. \end{aligned}$$

A further simplification arises from the fact that all points $\boldsymbol{x}_i$, $i \in I^*$, are mapped to the origin by $\boldsymbol{A}_I^\top$. Therefore, these points can be removed from the data set and the halfspace depth is computed w.r.t. the data set $\boldsymbol{x}_i$, $i \in (I^*)^c$. Therefore,

$$n D_{\mathrm{H}}(\boldsymbol{0} \,|\, \boldsymbol{A}_I^\top \boldsymbol{X}) = n D_{\mathrm{H}}(\boldsymbol{0} \,|\, \boldsymbol{A}_I^\top \boldsymbol{X}_{(I^*)^c}) + n_{I^*} \,,$$

and finally

$$\begin{aligned} n D_{\mathrm{H}}(\boldsymbol{0} \,|\, \boldsymbol{X}) &= \min_{I \in \mathcal{L}_k} \left[n D_{\mathrm{H}}(\boldsymbol{0} \,|\, \boldsymbol{A}_I^\top \boldsymbol{X}) - \left(n_{I^*} - n D_{\mathrm{H}}(\boldsymbol{0} \,|\, \boldsymbol{X}_{I^*}) \right) \right] \\ &= \min_{I \in \mathcal{L}_k} \left[n D_{\mathrm{H}}(\boldsymbol{0} \,|\, \boldsymbol{A}_I^\top \boldsymbol{X}_{(I^*)^c}) + n D_{\mathrm{H}}(\boldsymbol{0} \,|\, \boldsymbol{X}_{I^*}) \right] . \end{aligned}$$

In the same way as above it can be shown that

$$n D_{\mathrm{H}}(\boldsymbol{0} \,|\, \boldsymbol{X}_{I^*}) = n D_{\mathrm{H}}(\boldsymbol{0} \,|\, \boldsymbol{P}_I^\top \boldsymbol{X}_{I^*}) \,,$$

where $\boldsymbol{P}_I = [\boldsymbol{x}_{i_1}, \ldots, \boldsymbol{x}_{i_k}]$. Therefore, the following theorem holds.

Theorem 1.2 *(Dyckerhoff and M., 2016) With the notation from above, for each* $1 \leq k < d$ *it holds that*

$$n D_H(\boldsymbol{0} \,|\, \boldsymbol{X}) = \min_{I \in \mathcal{L}_k} \Big[n D_H(\boldsymbol{0} \,|\, \boldsymbol{A}_I^\top \boldsymbol{X}_{(I^*)^c}) + n D_H(\boldsymbol{0} \,|\, \boldsymbol{P}_I^\top \boldsymbol{X}_{I^*}) \Big] \,.$$

Note that for each subset I of k linearly independent data points the data points fall in one of two categories: The points whose projections on the orthogonal complement of $\mathrm{span}(\boldsymbol{X}_I)$ are different from $\boldsymbol{0}$ and those who are equal to $\boldsymbol{0}$. The former points are taken into account by $n D_{\mathrm{H}}(\boldsymbol{0} \,|\, \boldsymbol{A}_I^\top \boldsymbol{X}_{(I^*)^c})$, whereas the latter ones are considered by $n D_{\mathrm{H}}(\boldsymbol{0} \,|\, \boldsymbol{P}_I^\top \boldsymbol{X}_{I^*})$. Usually, there will be much more points of the first category than of the second one. The computation of $n D_{\mathrm{H}}(\boldsymbol{0} \,|\, \boldsymbol{A}_I^\top \boldsymbol{X}_{(I^*)^c})$ is done in dimension $d-k$, whereas the computation of $n D_{\mathrm{H}}(\boldsymbol{0} \,|\, \boldsymbol{P}_I^\top \boldsymbol{X}_{I^*})$ is done in dimension k. Thus, by the preceding theorem the calculation of one depth value in d-space is reduced to calculating many depth values in $(d-k)$-space (and in k-space). By choosing k it is possible to control how much the dimension is reduced in each step. The price for a higher dimension reduction is that more subsets I have to be considered in that step.

An important special case of the algorithm arises when the data points $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$ and $\mathbf{0}$ are in general position (see Definition 1.7 of Section 1.1.3). Since in that case any linear subspace of dimension k, $1 \leq k < d$ contains at most k data points $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$, if $\#I = k$, then $I^* = I$ and $n_{I^*} = k$. Further, $nD_{\mathrm{H}}(\mathbf{0} \,|\, \boldsymbol{P}_I^\top \boldsymbol{X}_{I^*}) = 0$, which can be seen by choosing $\boldsymbol{p}$ such that $\boldsymbol{p}^\top(\boldsymbol{P}_I^\top \boldsymbol{x}_{i_1} - \boldsymbol{P}_I^\top \boldsymbol{x}_{i_j}) = 0$, $j = 2, \ldots, k$. Therefore, the following corollary of the above theorem.

Corollary 1.1 *(Dyckerhoff and M., 2016) If the data points $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$ and $\mathbf{0}$ are in general position, then for each $1 \leq k < d$ it holds that*

$$nD_H(\mathbf{0} \,|\, \boldsymbol{X}) = \min_{I \in \mathcal{L}_k} nD_H(\mathbf{0} \,|\, \boldsymbol{A}_I^\top \boldsymbol{X}_{I^c}) \,.$$

1.2.2 A family of algorithms

The result of the previous subsection gives rise to several algorithms. In these algorithms the dimensionality of the data is reduced at different rates. When the dimension is reduced to $d = 1$ or $d = 2$, specialized algorithms may be used. For the case $d = 1$ the standard algorithm of complexity $O(n)$ is used. For bivariate data the algorithm of Rousseeuw and Ruts (1996) with complexity $O(n \log n)$ (or any other algorithm with this complexity) may be used.

Combinatorial algorithm, $k = d - 1$

If choosing $k = d - 1$, this results in the so-called *combinatorial algorithm* (Algorithm 1.1). In this algorithm, all hyperplanes defined by $d - 1$ linearly independent data points are

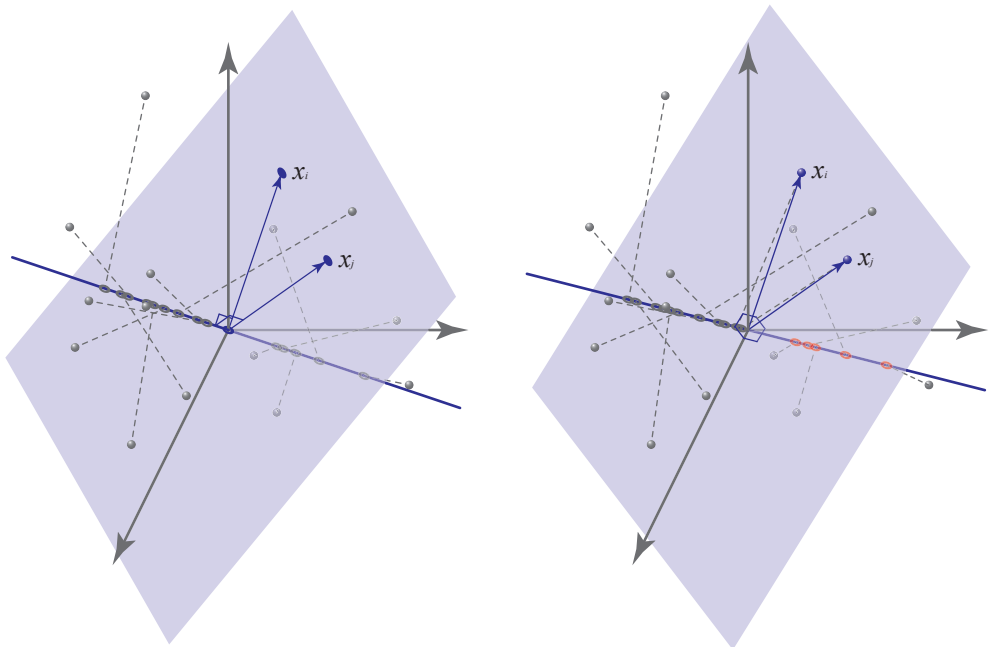


Figure 1.1: *Illustration of the combinatorial algorithm, $k = d - 1$*

considered. For each such hyperplane the data are projected in the direction normal to the hyperplane. Thus, the dimensionality is in one step reduced to dimension one. Only if there are more than $d-1$ data points in the considered hyperplane (which can only occur when the data are not in general position), then for these data points $\boldsymbol{y}_i = \boldsymbol{P}_I^\top \boldsymbol{x}_i$ is calculated and the procedure nD_{H} is recursively called for the data points $\boldsymbol{y}_{j_1}, \ldots, \boldsymbol{y}_{j_l}$. The algorithm is illustrated in Figure 1.1.

In the case of data in general position, the algorithm will never enter the recursion. Then, for each processed hyperplane the complexity of this algorithm is of order $O(n)$. Since there are $\binom{n}{d-1}$ subsets of $d-1$ data points, the overall complexity of the algorithm is $\binom{n}{d-1}O(n) = O(n^d)$.

Combinatorial algorithm, $k = d-2$

Another possibility is to use $k = d-2$ (Algorithm 1.2). In that case, the data points are directly projected into the 2-dimensional space. This has the advantage that for the projected points the algorithm of Rousseeuw and Ruts (1996) for the bivariate halfspace depth can be used. Since this algorithm has a complexity of $O(n \log n)$ and there are $\binom{n}{d-2}$ subsets of order $d-2$, the complexity of this algorithm is of order $\binom{n}{d-2}O(n \log n) = O(n^{d-1} \log n)$. Thus, this algorithm has a better complexity than the naive combinatorial algorithm with $k = d-1$. This combinatorial algorithm has been independently proposed by Liu (2017).

Algorithm 1.1 Combinatorial algorithm, $k = d-1$

function nHD_Comb$(d, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)$ ▷ Halfspace depth of 0
  if $d = 1$ **then return** nHD1$(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)$
  $n_{min} \leftarrow n$
  for each subset $I \subset \{1, \ldots, n\}$ of order $d-1$ **do**
    if $(\boldsymbol{x}_i)_{i \in I}$ linearly independent **then**
      Compute $\boldsymbol{p}_I$ such that $\boldsymbol{p}_I^\top \boldsymbol{x}_i = 0$ for all $i \in I$
      for all $\boldsymbol{x}_j$ **do**
        $z_j \leftarrow \boldsymbol{p}_I^\top \boldsymbol{x}_j$ ▷ project data points in direction $\boldsymbol{p}_I$
      $n_{new} \leftarrow \min\Big\{\#\{z_j > 0\}, \#\{z_j < 0\}\Big\}$
      if $\#\{z_j = 0\} > d-1$ **then**
        $\boldsymbol{P}_I \leftarrow \mathrm{Matrix}[(\boldsymbol{x}_i)_{i \in I}]$
        for all indices j with $z_j = 0$ **do**
          $\boldsymbol{y}_j \leftarrow \boldsymbol{P}_I^\top \boldsymbol{x}_j$
        $n_{new} \leftarrow n_{new} +$ nHD_Comb$(d-1, \boldsymbol{y}_{j_1}, \ldots, \boldsymbol{y}_{j_l})$
      if $n_{new} < n_{min}$ **then** $n_{min} \leftarrow n_{new}$
  return n_{min}

Algorithm 1.2 Combinatorial algorithm, $k = d - 2$

function NHD_COMB2($d, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$) ▷ Halfspace depth of 0
  if $d = 1$ **then return** NHD1($\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$)
  if $d = 2$ **then return** NHD2($\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$)
  $n_{min} \leftarrow n$
  for each subset $I \subset \{1, \ldots, n\}$ of order $d - 2$ **do**
    if $(\boldsymbol{x}_i)_{i \in I}$ linearly independent **then**
      Compute a basis $\boldsymbol{a}_1, \boldsymbol{a}_2$ of the orthogonal complement of $(\boldsymbol{x}_i)_{i \in I}$
      $\boldsymbol{A}_I \leftarrow \text{Matrix}[\boldsymbol{a}_1, \boldsymbol{a}_2]$
      $\boldsymbol{P}_I \leftarrow \text{Matrix}[\boldsymbol{x}_{i_1}, \ldots, \boldsymbol{x}_{i_{d-2}}]$
      for all $\boldsymbol{x}_j$ **do**
        if $\boldsymbol{A}_I^\top \boldsymbol{x}_j \neq \boldsymbol{0}$ **then** $\boldsymbol{y}_j \leftarrow \boldsymbol{A}_I^\top \boldsymbol{x}_j$
        else $\boldsymbol{z}_j \leftarrow \boldsymbol{P}_I^\top \boldsymbol{x}_j$
      $l \leftarrow \#\{j : \boldsymbol{A}_I^\top \boldsymbol{x}_j \neq \boldsymbol{0}\}$
      $n_{new} \leftarrow$ NHD2($\boldsymbol{y}_{j_1}, \ldots, \boldsymbol{y}_{j_l}$)
      if $n - l > d - 2$ **then**
        $n_{new} \leftarrow n_{new} +$ NHD_COMB2($d - 2, \boldsymbol{z}_{j_1}, \ldots, \boldsymbol{z}_{j_{n-l}}$)
      if $n_{new} < n_{min}$ **then** $n_{min} \leftarrow n_{new}$
  return n_{min}

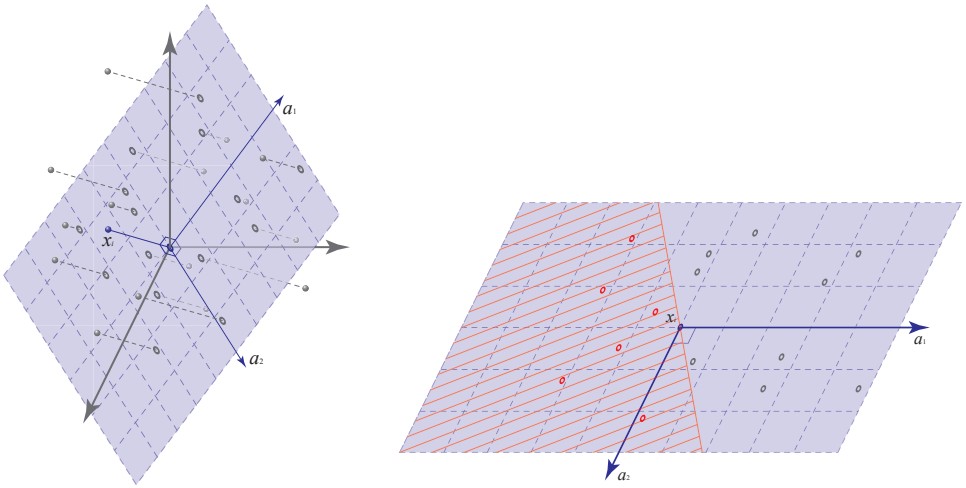


Figure 1.2: *Illustration of the recursive algorithm,* $k = 1$

Recursive algorithm, $k = 1$

The other extreme is the case, when choosing $k = 1$. This yields the so-called *recursive algorithm* (Algorithm 1.3). In this algorithm, in the outer loop all data points $\boldsymbol{x}_i$ are considered, and the data are projected on the hyperplane orthogonal to $\boldsymbol{x}_i$. For the projected data points (with the exception of the data points that are a multiple of $\boldsymbol{x}_i$ and are thus

mapped to the origin), the algorithm is called recursively. Thus, in each step the dimensionality is reduced only by one. The recursion stops when $d = 2$, in which case the algorithm of Rousseeuw and Ruts (1996) is applied. Note that the recursive algorithm can be viewed as a generalization of the algorithm for the case $d = 3$ in Rousseeuw (1998). Figure 1.2 shows an illustration of the recursive algorithm.

In the recursive algorithm the depth w.r.t. d-variate data is computed as the minimum over n depths w.r.t. $(d-1)$-variate data. Therefore, the complexity for d-variate data is n times the complexity for $(d-1)$-variate data. Since the recursion is stopped when $d = 2$, in which case the $O(n \log n)$-algorithm of Rousseeuw and Ruts (1996) is used, this results in an overall complexity of $n^{d-2}\, O(n \log n) = O(n^{d-1} \log n)$. Note that this remains true even if the data are not in general position.

1.2.3 Numerical illustration

For any k, the algorithm can be easily implemented in any programming environment. For $k = 1, ..., d-2$ the calculation of the orthogonal complement (which can easily be done, *e.g.*, using the Gauss-Jordan method) and the routine `nHD2` by Rousseeuw and Ruts (1996) have to be implemented. For $k = d-1$ even the `nHD2`-routine is of no need.

The algorithms of the preceding subsection should not be called directly, but rather be included in a wrapping function, which does the necessary preprocessing of the data. In the preprocessing, the point $\boldsymbol{z}$ is first subtracted from the data so that the halfspace depth of the origin has to be calculated. Second, all data points which are equal to the origin are removed from the data. Their number is stored and later added to the result of `nHD`. Third, if the data points are contained in some k-dimensional subspace, $k < d$, then the data points are mapped to $\mathbb{R}^k$ as described in Section 1.2.1. In an optional fourth step, the data could

Algorithm 1.3 Recursive algorithm, $k = 1$

function nHD_Rec$(d, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)$ ▷ Halfspace depth of 0
  if $d = 1$ **then return** nHD1$(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)$
  if $d = 2$ **then return** nHD2$(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)$
  $n_{min} \leftarrow n$
  for all $\boldsymbol{x}_i$ **do**
    Compute a basis $\boldsymbol{a}_1, \ldots, \boldsymbol{a}_{d-1}$ of the hyperplane with normal $\boldsymbol{x}_i$
    $\boldsymbol{A}_I \leftarrow \text{Matrix}[\boldsymbol{a}_1, \ldots, \boldsymbol{a}_{d-1}]$
    for all $\boldsymbol{x}_j$ **do**
      if $\boldsymbol{A}_I^\top \boldsymbol{x}_j \neq \boldsymbol{0}$ **then** $\boldsymbol{y}_j \leftarrow \boldsymbol{A}_I^\top \boldsymbol{x}_j$
        else $z_j \leftarrow \boldsymbol{x}_i^\top \boldsymbol{x}_j$
    $l \leftarrow \#\{j : \boldsymbol{A}_I^\top \boldsymbol{x}_j \neq \boldsymbol{0}\}$
    $n_{new} \leftarrow$ nHD_Rec$(d-1, \boldsymbol{y}_{j_1}, \ldots, \boldsymbol{y}_{j_l}) + \min\Big\{\#\{z_j > 0\}, \#\{z_j < 0\}\Big\}$
    if $n_{new} < n_{min}$ **then** $n_{min} \leftarrow n_{new}$
  return n_{min}

Table 1.1: *Execution times for data in general position, distributed as* $N(\mathbf{0}_d, \mathbf{I}_d)$*, averaged over 10 tries, in seconds (three significant digits). Variants of the algorithm with* $k = 1, d-2, d-1$ *are presented in the first, second, and third rows of each cell, respectively.*

	$n=40$	80	160	320	640	1280	2560	5120	10240	20480	40960	81920
$d=3$	0.000	0.000	0.000	0.011	0.047	0.184	0.780	3.19	13.2	54.5	225	933
	0.002	0.002	0.003	0.016	0.063	0.250	1.03	4.22	17.4	72.2	293	1210
	0.000	0.003	0.014	0.117	0.936	7.60	61.3	519	—	—	—	—
4	0.006	0.048	0.402	3.36	28.2	235	1960	—	—	—	—	—
	0.005	0.038	0.302	2.50	20.4	166	1360	—	—	—	—	—
	0.005	0.055	0.784	12.3	203	3290	—	—	—	—	—	—
5	0.205	3.62	62.5	1070	—	—	—	—	—	—	—	—
	0.055	0.952	16.1	269	—	—	—	—	—	—	—	—
	0.047	1.24	35.7	1110	—	—	—	—	—	—	—	—
6	7.32	275	—	—	—	—	—	—	—	—	—	—
	0.506	18.4	633	—	—	—	—	—	—	—	—	—
	0.392	21.6	1250	—	—	—	—	—	—	—	—	—
7	257	—	—	—	—	—	—	—	—	—	—	—
	3.60	278	—	—	—	—	—	—	—	—	—	—
	2.66	305	—	—	—	—	—	—	—	—	—	—
8	—	—	—	—	—	—	—	—	—	—	—	—
	21.4	3550	—	—	—	—	—	—	—	—	—	—
	14.3	3470	—	—	—	—	—	—	—	—	—	—
9	—	—	—	—	—	—	—	—	—	—	—	—
	107	—	—	—	—	—	—	—	—	—	—	—
	69.8	—	—	—	—	—	—	—	—	—	—	—
10	—	—	—	—	—	—	—	—	—	—	—	—
	439	—	—	—	—	—	—	—	—	—	—	—
	289	—	—	—	—	—	—	—	—	—	—	—

be scaled to have a norm of one. This does not change the halfspace depth, but has the advantage that all data have the same order of magnitude, which should reduce numerical problems.

All algorithms based on Theorem 1.2 can be further improved by exiting the main loop as soon as n_{min} drops to zero, since in that case no further improvement is possible. However, this speed-up is data dependent and occurs only if the origin is outside the convex hull of the data. Therefore, this modification is not in incorporated into the algorithms used in the experiments (see below) to get stable computation times.

Due to the independent repetition of similar operations, the algorithms based on Theorem 1.2 (for different values of k) possess high parallelization abilities, which grow with k. Clearly, for data in general position, the complexity of the algorithms for $1 \leq k \leq d-2$ is $O(n^{d-1} \log n)$ and is $O(n^d)$ for $k = d-1$. However, the exact execution time depends on the implementation of the single steps, routines, memory structures, *etc.*, and can differ in practice from the values reported in Tables 1.1 and 1.2. As it will be seen later in this subsection, each of the considered algorithms can show the best results (compared to the remaining two algorithms) for proper constellations of n and d. The algorithms' performance may differ a lot depending on whether the data are in general position or not. In

all experiments, one kernel of the Intel Core i7-2600 (3.4 GHz) processor having enough physical memory was used.

First, consider data $\boldsymbol{X}$ drawn randomly from a multivariate standard normal distribution $N(\mathbf{0}_d, \mathbf{I}_d)$, where the depth of the origin w.r.t. $\boldsymbol{X}$ is computed. Table 1.1 presents the execution times of the algorithms in seconds (in each cell the upper, middle, and lower lines correspond to $k = 1, 2, d-1$ respectively), averaged over 10 tries. Such a small number of tries is sufficient, as the execution times are extremely stable for all chosen values of n and d and differ by a few percents only. d is varied from 3 to 10 and $n = 10 \cdot 2^i$ is increased with $i = 2, 3, ...$ till the execution time exceeds one hour. As one can see, because of the recurrent structure, for fixed n and with increasing d, the algorithm with $k = 1$ is outperformed by $k = d-2$, which is further outperformed by $k = d-1$. On the other hand, for fixed d and with increasing n, the algorithm with $k = d-1$ is outperformed by $k = 1$ and $k = d-2$ because of the better complexity of the latter algorithms. Comparing the algorithms with $k = 1$ and with $k = d-2$ the former one is superior for dimension $d = 3$, whereas the latter performs better when $d > 3$.

Table 1.2: *Execution times for data in non-general position, distributed as* $U(\{-2,-1,0,1,2\}^d)$*, averaged over 10 tries, in seconds (three significant digits). Variants of the algorithm with* $k = 1, d-2, d-1$ *are presented in the first, second, and third rows of each cell, respectively. For* $k = d-1$ *and* $d = 7, ..., 10$ *the median is reported.*

	$n=40$	80	160	320	640	1280	2560	5120	10240	20480	40960	81920	163840
$d=3$	0.000	0.002	0.002	0.009	0.036	0.139	0.530	2.11	8.33	33.1	132	530	2290
	0.000	0.002	0.005	0.013	0.052	0.198	0.773	3.14	12.6	49.9	195	786	3480
	0.002	0.019	0.188	2.11	26.1	384	—	—	—	—	—	—	—
4	0.005	0.045	0.375	3.06	24.6	196	1560	—	—	—	—	—	—
	0.005	0.036	0.291	2.35	18.5	141	1110	—	—	—	—	—	—
	0.203	5.47	235	—	—	—	—	—	—	—	—	—	—
5	0.202	3.56	60.5	1020	—	—	—	—	—	—	—	—	—
	0.063	1.06	17.4	283	—	—	—	—	—	—	—	—	—
	4.49	869	—	—	—	—	—	—	—	—	—	—	—
6	7.29	272	—	—	—	—	—	—	—	—	—	—	—
	0.570	—	—	—	—	—	—	—	—	—	—	—	—
	78.1	—	—	—	—	—	—	—	—	—	—	—	—
7	256	—	—	—	—	—	—	—	—	—	—	—	—
	4.34	315	—	—	—	—	—	—	—	—	—	—	—
	227	—	—	—	—	—	—	—	—	—	—	—	—
8	—	—	—	—	—	—	—	—	—	—	—	—	—
	24.2	—	—	—	—	—	—	—	—	—	—	—	—
	754	—	—	—	—	—	—	—	—	—	—	—	—
9	—	—	—	—	—	—	—	—	—	—	—	—	—
	144	—	—	—	—	—	—	—	—	—	—	—	—
	1750	—	—	—	—	—	—	—	—	—	—	—	—
10	—	—	—	—	—	—	—	—	—	—	—	—	—
	457	—	—	—	—	—	—	—	—	—	—	—	—
	1800	—	—	—	—	—	—	—	—	—	—	—	—

The designed framework allows for handling data for which the general position assumption is violated. The results for $\boldsymbol{X}$ distributed uniformly on $\{-2,-1,0,1,2\}^d$ are presented in Table 1.2. As mentioned above (see also Corollary 1.1), for data in general position, if $k = d-1$ or $k = d-2$, no recursion is involved. If $k = d-1$, for data in non-general position, the recursive calls can increase the execution time and, in general, the algorithmic complexity. Additionally, if n is not large enough (depending on d), the computation times depend heavily on the exact position of the points in $\mathbb{R}^d$ and become unstable. Therefore, for the algorithm with $k = d-1$ the median was taken instead of the mean when reporting the execution times for $d = 7, ..., 10$. The same effect occurs in the case $k = d-2$ as well, but the application of the two-dimensional routine (nHD2) designed by Rousseeuw and Ruts (1996) seems to compensate this increase in time by a quick handling of ties, especially when n gets larger. On the other hand, if $k = 1$, ties are rather an advantage, and the execution time of the algorithm decreases.

1.3 Computation of the halfspace depth regions

Computation of a halfspace κ-central region $D^{\kappa}_{\mathrm{H}}(\boldsymbol{X})$ (or further simply κ-region) of given data $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$ appears to be a much more challenging task than computation of the halfspace depth of a single point, as it involves a very large number of possible observational hyperplanes to be inspected. By an *observational hyperplane* or *elemental hyperplane* a hyperplane that passes through (at least) d elements of $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$ is meant. In dimension two, the task has been solved by use of a circular sequence (Edelsbrunner, 1987), which enumerates all intersections of observational hyperplanes (Ruts and Rousseeuw, 1996). Throughout this section it is assumed that the data are in general position (see Definition 1.7 of Section 1.1.3).

According to Kong and Mizera (2012), the halfspace depth κ-region $D^{\kappa}_{\mathrm{H}}(\boldsymbol{X})$ is the infinite intersection over all directions $\boldsymbol{u} \in \mathbb{S}^{d-1}$ of the inner halfspaces bordered by $\mathcal{H}_{\mathrm{KM}}(\kappa, \boldsymbol{u})$. Hereafter,

$$\mathcal{H}_{\mathrm{KM}}(\kappa, \boldsymbol{u}) = \{\boldsymbol{z} \in \mathbb{R}^d : \boldsymbol{u}^\top \boldsymbol{z} \geq q_1(\kappa, \boldsymbol{u})\}\,,$$

where $q_1(\kappa, \boldsymbol{u})$ denotes the sample κ-quantile of the projections of $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$ onto $\boldsymbol{u}$. As a convex polytope, a halfspace region is the intersection of a *finite* number of these halfspaces. The facets of the polytope lie on the hyperplanes that border the halfspaces. Clearly, by the definition of the empirical version of the halfspace depth (1.10), each of these hyperplanes must be an observational hyperplane. Consequently, the halfspace depth region is completely determined by a finite number of observational hyperplanes. Hence, to calculate it, the key step is to identify those observational hyperplanes that actually include one of the region's facets. A naïve procedure consists in checking all $\binom{n}{d}$ observational hyperplanes. For a more efficient procedure, a strategy to identify those observational hyperplanes that contain the facets is needed.

Hallin et al. (2010) and Paindaveine and Šiman (2011), hereafter HPS, point out a direct connection between a halfspace depth region and multivariate regression quantiles. Each of

these quantiles consists of, in general more than one, parallel hyperplanes, which may contain a facet of the halfspace depth region. In their pioneering work, Hallin et al. (2010) show that those directions giving the same set of hyperplanes form a polyhedral cone, and that a finite number of these cones fills $\mathbb{R}^d$. Each cone is represented by the directions generating its edges, which, again, are finitely many. HPS propose an algorithm that calculates halfspace depth regions in dimension $d > 2$ via quantile regression and parametric programming. To guarantee all cones to be addressed, a breadth-first search is used. For details of the implementations, see Paindaveine and Šiman (2012a,b). However, these procedures are rather slow and insufficient in practice (see, *e.g.*, Table 1.6 for comparison), while a handy algorithm and a corresponding software package are needed in application.

In the sequel, two new algorithms are presented, a naïve and a more sophisticated one, that calculate halfspace depth regions in arbitrary dimension $d > 2$. In building the second algorithm, certain combinatorial properties of halfspace depth regions are derived and exploited that substantially reduce the computational load. Consequently this algorithm runs much faster and requires much less RAM than the naïve algorithm as well as the algorithms by HPS.

Specifically, as a first main result, an upper bound is derived on the number of *non-redundant hyperplanes* of a halfspace depth region, that is, those observational hyperplanes that contain a facet of the region. The bound is sharp and turns out to be very useful in assessing the computational complexity and performance of the algorithms. To the best knowledge of the literature, the bound is new.

The HPS procedures are slow for two reasons. First, it appears that both their implementations yield a great number of *redundant directions*, which are normal to an observational hyperplane but provide no facet of the trimmed region. However, all these directions are considered in HPS and used to calculate regions. Given κ, the HPS procedures *actually calculate d successive regions* instead of one by breadth-first search; but many of these regions have depth $\neq \kappa$ (see Paindaveine and Šiman (2011), remark after Theorem 4.2). Second, the *cone-by-cone* search strategy is both RAM- and time-consuming. This is because, in the HPS procedures, each cone is characterized by its facets and vertices, and facets are identified by d-variate vectors. A rather large RAM is required to store these identifiers with sufficient precision.

First, a naïve combinatorial algorithm (Algorithm 1.4) is presented. It serves as a benchmark for the principal fast algorithm (Algorithm 1.5). The naïve procedure, in searching for facets' candidates, simply passes through all combinations of $d-1$ observations as the case may be. No memory-consuming structure has to be created, and the computation time is independent of κ. In contrast, the fast approach (Algorithm 1.5) uses a breadth-first search strategy. However, instead of covering $\mathbb{R}^d$ *cone-by-cone*, as is done by HPS, it searches the directions *ridge-by-ridge*, where a *ridge* corresponds to a combination of $d-1$ observations in $\mathbb{R}^d$. This strategy yields only *relevant hyperplanes* (that cut off exactly the required number of observations from $D^{\kappa}_{\mathrm{H}}(\boldsymbol{X})$), and thus examines much fewer cases. Additionally, each ridge is stored by the subscripts of its $d-1$ corresponding observations and use of some novel tricks substantially saves both RAM and computation time. Obviously, Algorithm 1.4

is exact. For Algorithm 1.5, no theoretical proof of its exactness is available though. But broad numerical evidence suggests that it computes the exact region, and is expected to do so in vast majority of the cases (except those degenerate) in practice. In all experiments Algorithm 1.5 yielded precisely the same halfspace depth region as the exact Algorithm 1.4 does.

As they involve only simple operations and no optimization techniques, both proposed algorithms are easy to program. They also show high numerical precision even in larger dimension. Particularly, Algorithm 1.5, by its speed and storage efficiency, enables the use of statistical methodology based on halfspace-region statistics. To investigate the performance of the algorithms, a simulation study (up to dimension 9) as well as real data calculations are provided. The proposed procedures have been implemented in C++ and interfaced and visualized in `R` (R Core Team, 2022). They are available in the `R`-package `TukeyRegion` (M. and Barber, 2021) and can be downloaded from CRAN.

The *Tukey median* (Tukey, 1975) is one of the most famous generalizations of the ordinary median to dimension $d > 1$. It is usually defined (Donoho, 1982) as the average of all points in the Tukey median set. Hence its computation depends essentially on the computation of this innermost halfspace depth region. As an extension of the approach, an algorithm for fast computation of the Tukey median is provided.

The rest of this section is organized as follows. After necessary notations gathered in Section 1.3.1, Section 1.3.2 presents an upper bound on the number of non-redundant hyperplanes of a halfspace depth region, together with some results that are useful for the proposed algorithms. Section 1.3.3 describes the two novel algorithms, the naïve as well as the fast one. Section 1.3.4 studies its computational performance, also compared to HPS. Section 1.3.5 suggests an algorithm for computing the Tukey median. Finally, Section 1.3.6 gathers some remarks and very recent discoveries about Algorithm 1.5.

1.3.1 Notations

Here, some notions and notations are collected for later reference in this section: Denote $m_\kappa = \lceil n\kappa \rceil$ with $\lceil \cdot \rceil$ being the ceiling function. For any hyperplane $H_{\boldsymbol{u},\alpha} = \{\boldsymbol{x} \in \mathbb{R}^d : \boldsymbol{u}^\top \boldsymbol{x} = \alpha\}$, it is said that $H_{\boldsymbol{u},\alpha}$ *cuts off* m *observations* if

$$\min\big\{\#\{i : \boldsymbol{u}^\top \boldsymbol{x}_i > \alpha\}, \#\{i : \boldsymbol{u}^\top \boldsymbol{x}_i < \alpha\}\big\} = m\,.$$

If these two numbers are different, the normal pointing to the side with less observations is called the *outer direction* of a hyperplane. An observational hyperplane that cuts off exactly $m_\kappa - 1$ observations is mentioned as a *relevant hyperplane* of the halfspace κ-region $D^\kappa_{\mathrm{H}}(\boldsymbol{X})$; its inner halfspace as a *relevant halfspace*. Obviously, every non-redundant hyperplane (*i.e.* containing a facet) is also relevant, and $D^\kappa_{\mathrm{H}}(\boldsymbol{X})$ is the intersection of all relevant halfspaces.

By a *κ-outside ridge* a $(d-2)$-dimensional affine space is meant which contains $d-1$ observations and is the intersection of two observational hyperplanes, each cutting off, on its lower side, less than m_κ observations. Note that these two observational hyperplanes are

not unique (see Figure 1 for an illustration in dimension $d = 2$) and that only some of the κ-outside ridges belong to $D^{\kappa}_{\mathrm{H}}(\boldsymbol{X})$.

1.3.2 A bound on the number of facets

In this subsection, as a first principal result, a bound on the number of facets of a halfspace depth region is derived, that is, on the number of non-redundant halfspaces defining the region.

Assume that the observed data set $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_n\} \subset \mathbb{R}^d$, $n > d \geq 2$, is in general position (see Definition 1.7). Consequently, each facet of a halfspace depth region $D^{\kappa}_{\mathrm{H}}(\boldsymbol{X})$ lies on an observational hyperplane containing exactly d observations.

Proposition 1.5 *(Liu et al., 2019) Let Π be an observational hyperplane. If Π passes through observations $\boldsymbol{x}_{i_1}, \boldsymbol{x}_{i_2}, \cdots, \boldsymbol{x}_{i_d}$ and cuts off at most $m_\kappa - 1$ other observations, then*

$$\Pi \cap D^{\kappa}_{H}(\boldsymbol{X}) \subset \mathrm{conv}(\boldsymbol{x}_{i_1}, \boldsymbol{x}_{i_2}, \cdots, \boldsymbol{x}_{i_d}) .$$

Let $\mathbf{V} = span(\boldsymbol{x}_{i_1}, \boldsymbol{x}_{i_2}, \cdots, \boldsymbol{x}_{i_{d-1}})$ be the $(D-2)$-dimensional vector space spanned by $\{\boldsymbol{x}_{i_1} - \boldsymbol{x}_{i_{d-1}}, \boldsymbol{x}_{i_2} - \boldsymbol{x}_{i_{d-1}}, \cdots, \boldsymbol{x}_{i_{d-2}} - \boldsymbol{x}_{i_{d-1}}\}$, and $\mathbf{V}^{\perp}$ be its orthogonal complement. (Observe that $\mathbf{V} = \{\mathbf{0}\}$ if $d = 2$.) Consider the projection of $D^{\kappa}_{\mathrm{H}}(\boldsymbol{X})$ onto the two-dimensional vector space $\mathbf{V}^{\perp}$, which is $proj_{\mathbf{V}^{\perp}}\big(D^{\kappa}_{\mathrm{H}}(\boldsymbol{X})\big) = \{\boldsymbol{x}' \in \mathbf{V}^{\perp} : \boldsymbol{x}' + \boldsymbol{x}'' \in D^{\kappa}_{\mathrm{H}}(\boldsymbol{X}) \text{ with } \boldsymbol{x}'' \in \mathbf{V}\}$. Clearly, this projection is a polygone. Relying on Proposition 1.5, one obtains the following result, which will be useful in constructing the fast algorithm (Algorithm 1.5 in Section 1.3.3).

Proposition 1.6 *(Liu et al., 2019) Consider the vector space $\mathbf{V}^{\perp}$ as before, and the κ-outside ridge containing the observations $\boldsymbol{x}_{i_1}, \boldsymbol{x}_{i_2}, \cdots, \boldsymbol{x}_{i_{d-1}}$. The projection of $\boldsymbol{x}_{i_j}$ onto $\mathbf{V}^{\perp}$ is either a vertex or no element of $proj_{\mathbf{V}^{\perp}}\big(D^{\kappa}_{H}(\boldsymbol{X})\big)$, $j = 1, 2, \ldots, d-1$.*

For $\kappa \in \{1/n,\ 2/n,\ \cdots,\ \kappa^*\}$ with $\kappa^* = \sup_{\boldsymbol{x} \in \mathbb{R}^d} D_{\mathrm{H}}(\boldsymbol{X}))$, it is seen from Proposition 1.6 and the convexity of $D^{\kappa}_{\mathrm{H}}(\boldsymbol{X})$ that there exist at least two relevant hyperplanes each containing the observations $\boldsymbol{x}_{i_1}, \boldsymbol{x}_{i_2}, \cdots, \boldsymbol{x}_{i_{d-1}}$ included in this κ-outside ridge plus another observation, which is found in the two-dimensional space $\mathbf{V}^{\perp}$. This fact will be used in Step 2(a) of Algorithm 1.4 below.

Moreover, the intersection of the respective halfspaces contains the corresponding $(d-2)$-dimensional affine space. It is easy to see that among the halfspaces bordered by them two exist, say $\mathcal{H}_1$ and $\mathcal{H}_2$, such that $D^{\kappa}_{\mathrm{H}}(\boldsymbol{X}) \subset \mathcal{H}_1 \cap \mathcal{H}_2$. Figure 1.3 illustrates this; it also demonstrates that a relevant hyperplane can be redundant. Hence, the following result.

Proposition 1.7 *(Liu et al., 2019) Consider $\kappa \in \{1/n,\ 2/n,\ \cdots,\ \kappa^*\}$. If the halfspace depth region $D^{\kappa}_{H}(\boldsymbol{X})$ is* not *a singleton, the number of its facets (= number of its non-redundant hyperplanes) is bounded from above by $2\binom{n}{d-1}/d$.*

By the convexity of the halfspace depth region, an outer direction vector yields at most one of its facets. In this sense, Proposition 1.7 actually also provides an upper bound for

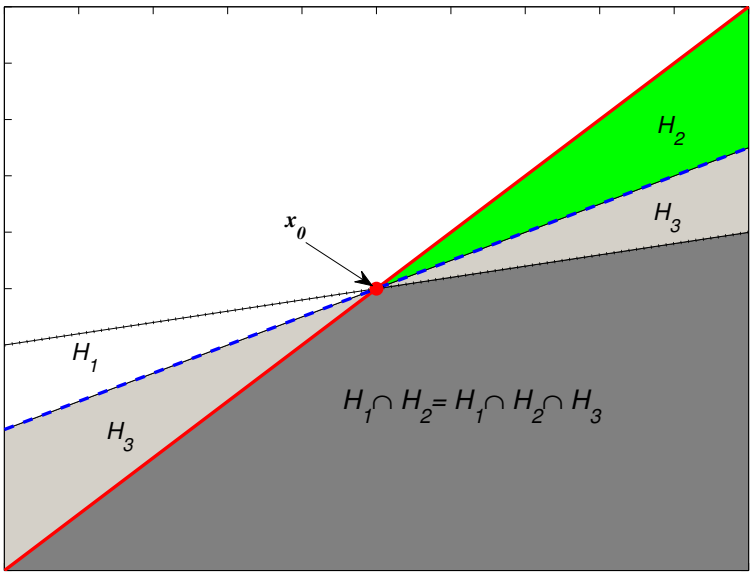


Figure 1.3: *Intersection of three relevant halfspaces. Here* $\boldsymbol{x}_0$ *denotes the projection of* $\boldsymbol{x}_{i_1}, \boldsymbol{x}_{i_2}, \cdots, \boldsymbol{x}_{i_{d-1}}$ *onto* $\mathbf{V}^{\perp}$*. Clearly,* $\bigcap_{k=1}^{3} \mathcal{H}_k = \mathcal{H}_1 \cap \mathcal{H}_2$*, and* $\mathcal{H}_3$ *is redundant.*

the number of outer directions of facets. It is useful in assessing the performance of an algorithm, and will be used in Step 4 of Algorithm 1.5

It is worth mentioning that the upper bound $2\binom{n}{d-1}dp$ is *attainable* and thus cannot be further improved. For instance, let $n = d + 1$. Then the halfspace depth region at depth $\kappa = \frac{1}{d+1}$ is the convex hull of the data. It has $d + 1$ facets, which equals the upper bound, $2\binom{d+1}{d-1}/d = d+1$. Though the bound is sharp, in many instances the number of facets comes out to be much smaller, as will be seen from the numerical study below.

Propositions 1.6 and 1.7 reveal further important properties of κ-outside ridges, which are useful in constructing a fast algorithm to calculate $D^{\kappa}_{\mathrm{H}}(\boldsymbol{X})$ for $\kappa \in \{1/n,\ 2/n,\ \cdots,\ \kappa^*\}$:

- Every κ-outside ridge can be utilized to compute at least two directions that are normal to relevant hyperplanes.

- The κ-outside ridges are connected with each other in the following sense: Given a κ-outside ridge, one may consider its defining observations $\boldsymbol{x}_{i_1}, \boldsymbol{x}_{i_2}, \cdots, \boldsymbol{x}_{i_{d-1}}$ and add another observation $\boldsymbol{x}^*$, so that the hyperplane through $\boldsymbol{x}_{i_1}, \boldsymbol{x}_{i_2}, \cdots, \boldsymbol{x}_{i_{d-1}}, \boldsymbol{x}^*$ is relevant. Then another κ-outside ridge is obtained by replacing one of the defining observations with $\boldsymbol{x}^*$. This can be utilized in the computation of $D^{\kappa}_{\mathrm{H}}(\boldsymbol{X})$.

- Observe that a κ-outside ridge is also a κ'-*outside* for all $\kappa' \in \{\frac{m_\kappa+1}{n},\ \frac{m_\kappa+2}{n},\ \cdots,\ \kappa^*\}$. This is for example true for the observations projected into the darkest area of Figure 1.3. Each of them can be used in the construction of a κ'-outside ridge. Hence, if one has to calculate more than one region, one may store all κ-outside ridges during

the computation of $D_{\mathrm{H}}^{\kappa}(\boldsymbol{X})$ and recycle them when computing a more central trimmed region $D_{\mathrm{H}}^{\kappa'}(\boldsymbol{X})$ with $\kappa' > \kappa$.

1.3.3 Two algorithms

This subsection presents two new algorithms to compute a halfspace depth region of given depth. Let us start by introducing Algorithm 1.4, which is a naïve application of Proposition 1.6. Algorithm 1.4 is simple and intuitive; it will later serve to verify the correctness of Algorithm 1.5. After this, the fast algorithm, namely Algorithm 1.5, is described.

The naïve combinatorial algorithm

Algorithm 1.4 simply passes through all combinations $\{\boldsymbol{x}_{i_1}, \cdots, \boldsymbol{x}_{i_{d-1}}\}$ of $d-1$ out of n points and searches for hyperplanes passing through d points and cutting off exactly $m_\kappa - 1$ observations. To do this, first for each choice of observations $\boldsymbol{x}_{i_1}, \cdots, \boldsymbol{x}_{i_{d-1}}$, the $(d-2)$-dimensional vector space spanned by them is calculated (Step 2a), and the sample is projected onto its orthogonal complement, which is a two-dimensional vector space (Step 2b). Then the search narrows down to finding two lines in this plane passing through the point to which $\boldsymbol{x}_{i_1}, \cdots, \boldsymbol{x}_{i_{d-1}}$ are projected and another point from the sample and cutting off $m_\kappa - 1$ observations (Step 2c). Figure 1.4 (right) visualizes the set of hyperplanes found during one iteration in Steps 2a to 2c. Each found hyperplane is stored as a number to the basis n. This requires $\lceil d \log_2(n) \rceil$ binary digits (bits) of memory space (Step 2d). Note that, under the assumption of general position made above, ties cannot occur, as any hyperplane contains at most d observations. Since the number of those combinations is $\binom{n}{d-1}$ and search in each of them is algorithmically dominated by the angle-sorting procedure having time complexity $O\big(n \log(n)\big)$, the time complexity of the algorithm amounts to $O\big(n^d \log(n)\big)$. Clearly, this presumes that the time complexity of Step 2d(iii) is not larger than that of Step 2d(i), which can be achieved by using appropriate store-search structures such as a search tree (access time complexity $O\big(d \log(n)\big)$) or a binary hypermatrix (access time complexity $O(1)$). As Algorithm 1.4 does not take account of any space ordering it requires very little memory and saves computation time, which otherwise is needed for multiple access of search structures and may grow substantially with n and d. In addition, due to the same reason its execution time only negligibly depends on the geometry of the data cloud and the depth value κ.

Algorithm 1.4 (Naïve combinatorial algorithm)

Input: $\boldsymbol{x}_1, \cdots, \boldsymbol{x}_n \in \mathbb{R}^d$, κ.

Step 1. *Set* $\mathcal{H}_\kappa = \varnothing$.

Step 2. *For each subset* $\{i_1, ..., i_{d-1}\} = I \subset \{1, \cdots, n\}$ *do:*

(a) Consider the plane normal to span$(\boldsymbol{x}_{i_1}, ..., \boldsymbol{x}_{i_{d-1}})$ *and find a basis of it. Let* $\boldsymbol{B}_I$ *denote the* basis matrix*, that is the* $(d \times 2)$ *matrix* $\boldsymbol{B}_I$ *containing the two basis vectors as columns.*

(b) *Compute* $\boldsymbol{y}_i = \boldsymbol{B}_I^\top \boldsymbol{x}_i$ *for* $i = 1, ..., n$.

(c) *Find a subset* $I_\kappa \subset \{1, \cdots, n\} \setminus I$ *such that for each* $i_0 \in I_\kappa$ *holds* $\#\{j : \boldsymbol{u}^\top \boldsymbol{y}_j > \boldsymbol{u}^\top \boldsymbol{y}_{i_0}, j = 1, ..., n\} = m_\kappa - 1$ *whenever* $\boldsymbol{u}^\top(\boldsymbol{y}_{i_1} - \boldsymbol{y}_{i_0}) = 0$.

(d) *For each* $i_0 \in I_\kappa$ *do:*

i. $(k_1, ..., k_d) = sort\big(i_0, i_1, ..., i_{d-1}\big)$.

ii. *Compute* $h = k_1 + k_2 n + k_3 n^2 + ... + k_d n^{d-1}$.

iii. *If* $h \notin \mathcal{H}_\kappa$ *then add* h *to* $\mathcal{H}_\kappa$.

Output: $\mathcal{H}_\kappa$.

The ridge-by-ridge breadth-first search strategy

Now, the main procedure can be presented, *i.e.* Algorithm 1.5 that computes the halfspace κ-region for given κ in a fast way. The algorithm has been implemented in function `TukeyRegion` of the `R`-package `TukeyRegion` and can be downloaded from CRAN. Its output depends on the user's request. A detailed description of the algorithm with reference to user-accessible options is given below.

To *find all relevant hyperplanes* of a halfspace κ-region, Algorithm 1.5 follows the breadth-first search idea. This idea can be briefly explained as follows: push an initial set of ridges into the queue, retrieve a ridge from the queue and push all those "neighbors" into the queue that have not been seen before (which will be specified below in Step 4); continue until the queue is empty (Steps 1 to 5). After this has been done, the region is constructed as an intersection of relevant halfspaces. For this, firstly, an inner point of the region has to be found, which is used for a dual transformation of the relevant hyperplanes (Step 6); then the QHULL (Barber et al., 1996) algorithm is run to *eliminate the redundant hyperplanes* (Step 7). Vertices, facets and the barycenter of the halfspace depth region are computed from the remaining (non-redundant) hyperplanes (Step 8). The input consists of the observations, the depth level κ and a precision parameter ϵ.

Algorithm 1.5 (Algorithm for computing the halfspace depth region)

Input: $\boldsymbol{x}_1, \cdots, \boldsymbol{x}_n \subset \mathbb{R}^d$, κ, ϵ.

Step 1. Initialization:

Set $\mathcal{A} = (\texttt{false}_n)^{d-1}$, $\mathcal{H}_\kappa = \varnothing$, *an empty queue* $\mathcal{Q}$.

Step 2. Construct initial set of ridges:

(a) *Find a subset* $\{i_1, ..., i_{d-1}\} = I \subset \{1, \cdots, n\}$ *such that* $(\boldsymbol{x}_{i_1}, ..., \boldsymbol{x}_{i_{d-1}})$ *define a ridge of* $conv(\boldsymbol{x}_1, ..., \boldsymbol{x}_n)$. *Set* $A\big(sort(I)\big) = \texttt{true}$ *and push* $sort(I)$ *into* $\mathcal{Q}$.

(b) *Compute a basis matrix* $\boldsymbol{B}_I$ *of the plane normal to* $span(\boldsymbol{x}_{i_1}, ..., \boldsymbol{x}_{i_{d-1}})$.

(c) *Compute* $\boldsymbol{y}_i = \boldsymbol{B}_I^\top \boldsymbol{x}_i$ *for* $i = 1, ..., n$.

(d) Find two indices l_1, l_2 *such that holds* $\#\{j : \boldsymbol{u}_k^\top \boldsymbol{y}_j > \boldsymbol{u}_k^\top \boldsymbol{y}_{l_k}, j = 1, ..., n\} = m_\kappa - 1$ *whenever* $\boldsymbol{u}_k^\top(\boldsymbol{y}_{l_k} - \boldsymbol{y}_{i_1}) = 0$ *for* $k = 1, 2$.
For $k = 1, 2$ *and for each subset* $J \subset I$ *of order* $d-2$, *set* $A\big(sort(J \cup \{l_k\})\big) =$ `true` *and push* $sort(J \cup \{l_k\})$ *into* $\mathcal{Q}$.

(e) For $k = 1, 2$ *and for each* l *such that holds* $\boldsymbol{u}_k^\top \boldsymbol{y}_l > \boldsymbol{u}_k^\top \boldsymbol{y}_{l_k}$ *and for each subset* $J \subset I$ *of order* $d-2$, *set* $A\big(sort(J \cup \{l\})\big) =$ `true` *and push* $sort(J \cup \{l\})$ *into* $\mathcal{Q}$.

Step 3. *Retrieve a ridge* $I = \{i_1, \cdots, i_{d-1}\}$ *from* $\mathcal{Q}$.

Step 4. Spread to neighboring ridges:

(a) Compute a basis matrix $\boldsymbol{B}_I$ *of the plane normal to* $span(\boldsymbol{x}_{i_1}, ..., \boldsymbol{x}_{i_{d-1}})$.

(b) Compute $\boldsymbol{y}_i = \boldsymbol{B}_I^\top \boldsymbol{x}_i$ *for* $i = 1, ..., n$.

(c) Find a subset $I_\kappa \subset \{1, \cdots, n\} \setminus I$ *such that for each* $i_0 \in I_\kappa$ *holds* $\#\{j : \boldsymbol{u}^\top \boldsymbol{y}_j > \boldsymbol{u}^\top \boldsymbol{y}_{i_0}, j = 1, ..., n\} = m_\kappa - 1$ *whenever* $\boldsymbol{u}^\top(\boldsymbol{y}_{i_1} - \boldsymbol{y}_{i_0}) = 0$.

(d) For each $i_0 \in I_\kappa$ *do:*

i. If $(I \cup \{i_0\}) \notin \mathcal{H}_\kappa$ *then add* $(I \cup \{i_0\})$ *to* $\mathcal{H}_\kappa$.

ii. For each subset $J \subset I_\kappa$ *of order* $d-2$ *do:*

If $A\big(sort(J \cup \{i_0\})\big) =$ `false` *then set* $A\big(sort(J \cup \{i_0\})\big) =$ `true` *and push* $sort(J \cup \{i_0\})$ *into* $\mathcal{Q}$.

Step 5. *If* $\mathcal{Q}$ *is not empty then go to* **Step 3**, *else go to the following step.*

(So far, all d-tuples of observations that define relevant halfspaces *are stored in* $\mathcal{H}_\kappa$. *The intersection of these halfspaces is the halfspace* κ*-region. Details given below.)*

Step 6. Find an inner point of the region:

(a) For each $(i_1, ..., i_d) \in \mathcal{H}_\kappa$ $(l = 1, ..., n_\kappa = \#\mathcal{H}_\kappa)$ *do:*

i. Find $\boldsymbol{u}_l \perp span(\boldsymbol{x}_{i_1}, ..., \boldsymbol{x}_{i_d})$ *such that* $\#\{j : \boldsymbol{u}_l^\top \boldsymbol{x}_j > \boldsymbol{u}_l^\top \boldsymbol{x}_{i_1}, j = 1, ..., n\} = m_\kappa - 1$ *and* $\|\boldsymbol{u}_l\|_2 = 1$.

ii. Compute $b_l = \boldsymbol{u}_l^\top \boldsymbol{x}_{i_1}$.

(b) Compute $\boldsymbol{x}_0 = \arg\max_{\boldsymbol{x} \in \mathbb{R}^d} \{\boldsymbol{x}^\top (1, 0, ..., 0)^\top : \boldsymbol{u}_l^\top \boldsymbol{x} \leq b_l - \epsilon, l = 1, ..., n_\kappa\}$.

(c) If $\boldsymbol{x}_0$ *cannot be found then* **stop**.

Step 7. Eliminate redundant halfspaces:

(a) For $j = 1, ..., n_\kappa$ *do:*

$$\boldsymbol{w}_j = \frac{1}{b_j - \boldsymbol{u}_j^\top \boldsymbol{x}_0} \boldsymbol{u}_j.$$

(b) $(\boldsymbol{i}_1, ..., \boldsymbol{i}_{n_\kappa^v})^\top = \mathrm{QHULL}(\boldsymbol{w}_1, ..., \boldsymbol{w}_{n_\kappa})$ *with* $\boldsymbol{i}_j = (i_{j1}, ..., i_{jd})$.

Step 8. Compute elements that define the region:

(a) For $j = 1, ..., n_\kappa^v$ *do:*

$$\boldsymbol{v}_j = \big((\boldsymbol{u}_{i_{j1}}, ..., \boldsymbol{u}_{i_{jd}})^\top\big)^{-1}(b_{i_{j1}}, ..., b_{i_{jd}})^\top.$$

(b) For each $i \in$ *unique*$(\boldsymbol{i}_1, ..., \boldsymbol{i}_{n_\kappa^v})$ $(j = 1, ..., n_\kappa^f)$ *do:*

i. $\boldsymbol{d}_j = \boldsymbol{u}_i$.

ii. $t_j = b_i$.

(c) $\boldsymbol{c} = ave\big(conv(\boldsymbol{v}_1, ..., \boldsymbol{v}_{n_\kappa^v})\big)$, *the barycenter of* $conv(\boldsymbol{v}_1, ..., \boldsymbol{v}_{n_\kappa^v})$.

Output:

(a) Vertices: $\mathcal{V} = \{\boldsymbol{v}_1, ..., \boldsymbol{v}_{n_\kappa^v}\}$.

(b) Facets' (non-redundant) hyperplanes: $\mathcal{F} = \{\boldsymbol{d}_1, ..., \boldsymbol{d}_{n_\kappa^f}\}$ *(outside-pointing normals) and* $\mathcal{T} = \{t_1, ..., t_{n_\kappa^f}\}$ *(thresholds on these normals).*

(c) Barycenter: $\boldsymbol{c}$.

In **Step 1**, $(\texttt{false}_n)^{d-1}$ is a $(d-1)$-dimensional logical matrix having format $n \times \cdots \times n$, that is, in the beginning an n-dimensional vector of logical zeros, brought to power $d-1$ as a Cartesian product. (*E.g.*, if $d = 4$ this is a cube of size $n \times n \times n$.) Indeed, only one upper corner of this matrix is used, which includes those cells having strictly decreasing subscripts. Further, a single bit of RAM suffices to store a logical value, which amounts to eight values per byte. However, this matrix is memory demanding when n and d are large. Fortunately, in this last case the matrix is very probable to be sparse, and thus some dynamic storing structure may be used, *e.g.* a search tree. $\mathcal{H}_\kappa$ is a set for storing relevant hyperplanes as d-tuples of integer numbers, and $\mathcal{Q}$ is literally a queue of ridges supporting operations of pushing an element on one side and retrieving it from another one.

Step 2 aims at finding a set of $(d-1)$-tuples defining ridges to be used as starting points for the algorithm. First, a ridge of the convex hull of $\{\boldsymbol{x}_1, ..., \boldsymbol{x}_n\}$ defined by, say, $\{\boldsymbol{x}_{j_1}, \boldsymbol{x}_{j_2}, \cdots, \boldsymbol{x}_{j_{d-1}}\}$ is found (Step 2a). This is implemented in the QHULL algorithm. Further, using the logic of Steps 2a to 2c of Algorithm 1.4, two points $\boldsymbol{x}_{l_1}$ and $\boldsymbol{x}_{l_2}$ are found (Steps 2b to 2d). Thus, each of the two hyperplanes defined by $\{\boldsymbol{x}_{j_1}, \boldsymbol{x}_{j_2}, \cdots, \boldsymbol{x}_{j_{d-1}}, \boldsymbol{x}_{l1}\}$ and $\{\boldsymbol{x}_{j_1}, \boldsymbol{x}_{j_2}, \cdots, \boldsymbol{x}_{j_{d-1}}, \boldsymbol{x}_{l2}\}$ is relevant. This is always guaranteed as long as the sample is in general position and $m_\kappa \leq \lfloor \frac{n-(d-1)}{2} \rfloor$; here $\lfloor \cdot \rfloor$ denotes the floor function. Then, all $(d-1)$-tuples containing $(d-2)$ points out of $\{\boldsymbol{x}_{j_1}, \boldsymbol{x}_{j_2}, \cdots, \boldsymbol{x}_{j_{d-1}}\}$ and one of the points cut off by or lying in one of these hyperplanes are chosen. Together with the ridge on the convex hull of the data set, this gives exactly $1 + 2m_\kappa(d-1)$ initial ridges, see Figure 1.4 (left) for illustration.

Steps 3 and **5** wrap the search step procedure of Step 4 by implementing the queue. Step 3 retrieves a ridge to be processed from the head of the queue while Step 5 returns to Step 3 if the queue contains at least one element.

Step 4 provides the identification of neighboring ridges and, by that, controls the ridge-by-ridge search strategy. First, a set I_κ of indices is found that defines, together with the current ridge $\{\boldsymbol{x}_{j_1}, \boldsymbol{x}_{j_2}, \cdots, \boldsymbol{x}_{j_{d-1}}\}$, a relevant hyperplane. (Here one follows the logic of

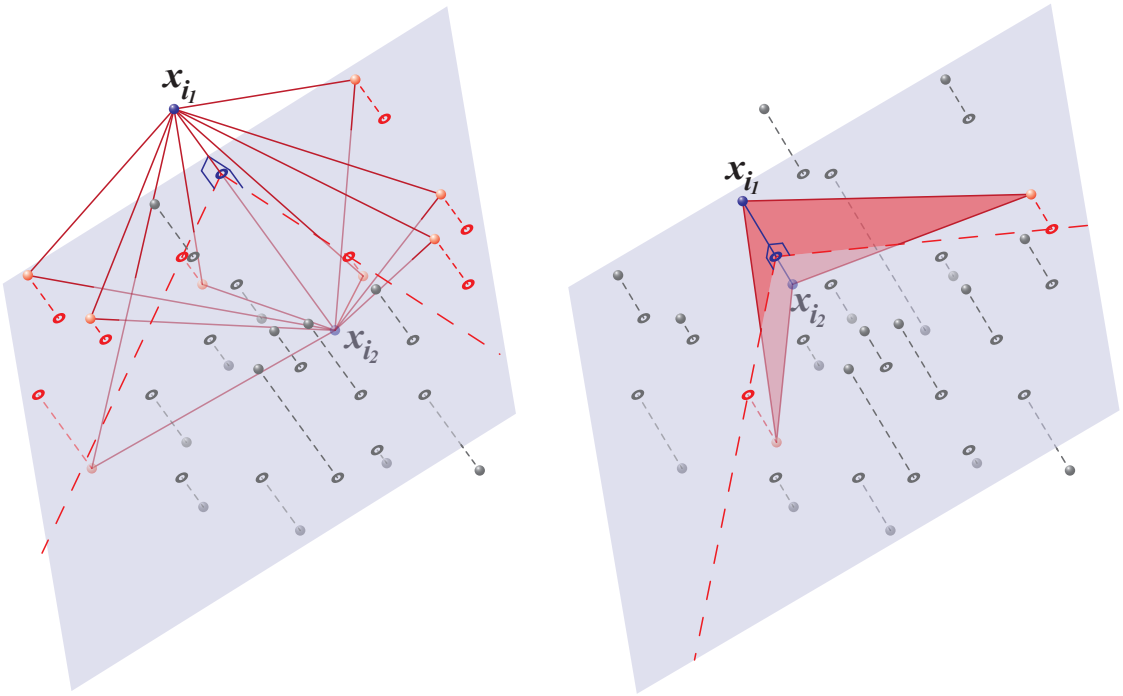


Figure 1.4: *Left: Step 2 of Algorithm 1.5, solid red lines indicate initial ridges. Right: The generic Step 2 (a) – (c) of Algorithm 1.4 and Step 4 (a) – (c) of Algorithm 1.5, solid red lines indicate new ridges added to the queue.*

Steps 2a to 2c of Algorithm 1.4 as before.) The information gained by each $i_0 \in I_\kappa$ is twofold. First, an i_0 defines a relevant hyperplane determined by a d-tuple $\{\boldsymbol{x}_{i_1}, \boldsymbol{x}_{i_2}, \cdots, \boldsymbol{x}_{i_{d-1}}, \boldsymbol{x}_{i_0}\}$. It is checked whether this hyperplane is visited for the first time and add it to $\mathcal{H}_\kappa$ if this is the case (Step 4d(i)). Second, each such hyperplane contains d ridges defined by $(d-1)$-tuples of points. $d-1$ of these ridges – those containing i_0 – can potentially lead to a hyperplane not visited before. Thus for each of these $(d-1)$ ridges it is checked whether it has been visited before or not. If a ridge has not been visited, it is added to the queue $\mathcal{Q}$ and mark it as visited in $\mathcal{A}$ (Step 4d(ii)). The set of ridges found in Step 4 of a single iteration is visualized in Figure 1.4 (right).

As the total number of relevant hyperplanes (after the first five steps of Algorithm 1.5 have been performed) can potentially be as large as their maximum number, the complexity of the algorithm is the same as that of Algorithm 1.4, *i.e.* $O\big(n^d \log(n)\big)$. On the other hand, this worst case happens only for degenerate data sets. Thus it is reasonable to expect that in most cases the number of relevant hyperplanes is substantially lower than the upper bound, and this number actually defines the computation speed of the algorithm. Some empirical insights to this question will follow in Section 1.3.4.

Steps 1 to **5** aim to compute relevant hyperplanes. This set will be used to check the correctness of the algorithm when comparing it empirically with the output of Algorithm 1.4. The implementation of Algorithm 1.5 in the `R`-package `TukeyRegion` performs these five first steps always independently of the chosen options. The following steps are optional. If the

algorithm is terminated after Step 5, only the relevant hyperplanes are output, each as a d-tuple $\{j_1, j_2, \cdots, j_d\}$ of indices of points from the sample.

Step 6 searches for a point strictly belonging to the interior of the halfspace depth region. If such a point cannot be found then the algorithm stops. (See also Section 1.3.5.) Step 6 is performed either if the interior point is explicitly requested (flag `retInnerPoint` is set) or if any option for the following steps is chosen.

Each element of $\mathcal{H}_\kappa$ is a d-tuple of indices $(j_1, j_2, \cdots, j_d)$ defining a halfspace by the hyperplane containing $\{\boldsymbol{x}_{j_1}, \boldsymbol{x}_{j_2}, \cdots, \boldsymbol{x}_{j_d}\}$. First, for each of these hyperplanes the normal pointing outside the halfspace depth region and the threshold defining the hyperplane's position are calculated (Step 6a); each of them defines a condition for the interior point. Then, the inner point $\boldsymbol{x}_0$ of the region is searched by means of linear programming (`R`-package `Rglpk` used here; Theussl and Hornik, 2019) as a point satisfying these conditions (Step 4b). If the inner point is not found this means that the halfspace depth region of depth κ has (numerically, with *precision* ϵ) zero volume or does not exist. In this case the algorithm stops.

Step 7 determines non-redundant hyperplanes. It is performed either if these are requested (flag `retHyperplanesNR`) or if any option for the following step is chosen. First, a duality transformation is applied. It represents each halfspace by a vector $\boldsymbol{w}_j$, which is its outer normal multiplied by the inverse distance of the hyperplane to the inner point (Step 7a). Second, the QHULL algorithm is applied to the set of all $\boldsymbol{w}_j$'s. It returns the convex hull of the $\boldsymbol{w}_j$'s, an $n_\kappa^v \times d$ matrix, where each row defines a facet by indices of d points (Step 7b). Since the data is in general position, each facet of this convex hull is defined by exactly d points.

Step 8 computes the elements of the region; it is the step that provides the practically important output. This step depends on whether region's vertices (flag `retVertices` is set), facets (flag `retFacets` is set) and/or barycenter (flag `retBarycenter` is set) are requested and requires the successful execution of all preceding steps. Each facet of the convex hull of $\boldsymbol{w}_j$'s in the dual space (corresponding to a row of the matrix returned by QHULL in Step 7b) defines a vertex of the halfspace depth region, *i.e.* each such vertex is computed as an intersection of d non-redundant hyperplanes (Step 8a).

The facets of the halfspace depth region are contained in non-redundant hyperplanes defined by pairs $(\boldsymbol{d}_j, t_j)$, $j = 1, ..., n_\kappa^f$ obtained as non-repeating entries of the matrix $(\boldsymbol{i}_1, ..., \boldsymbol{i}_{n_\kappa^v})^\top$ (Step 8b). To obtain facets as polygons one can apply QHULL to the set of regions' vertices $\{\boldsymbol{v}_1, ..., \boldsymbol{v}_{n_\kappa^v}\}$. In addition, the barycenter of a halfspace depth region can be computed as the weighted average of the triangulated (doable by QHULL as well) region, where points are the means of vertices of simplices (d-dimensional triangles) and weights are the volumes of these simplices (Step 8c).

To illustrate the output of the algorithm, Figure 1.5 exhibits[1.1] $\mathcal{D}(\kappa)$ for $\kappa = 0.025$, 0.05, 0.1, 0.15, 0.2, 0.25, 0.3 for the 748 3-dimensional observations of the Blood Transfusion data set, which is downloadable from the UCI Machine Learning Repository (Dua and Graff, 2017,

[1.1] Figures 1.5, 1.6, and 1.7 were generated with the QHULL software written at the Geometry Center, University of Minnesota.

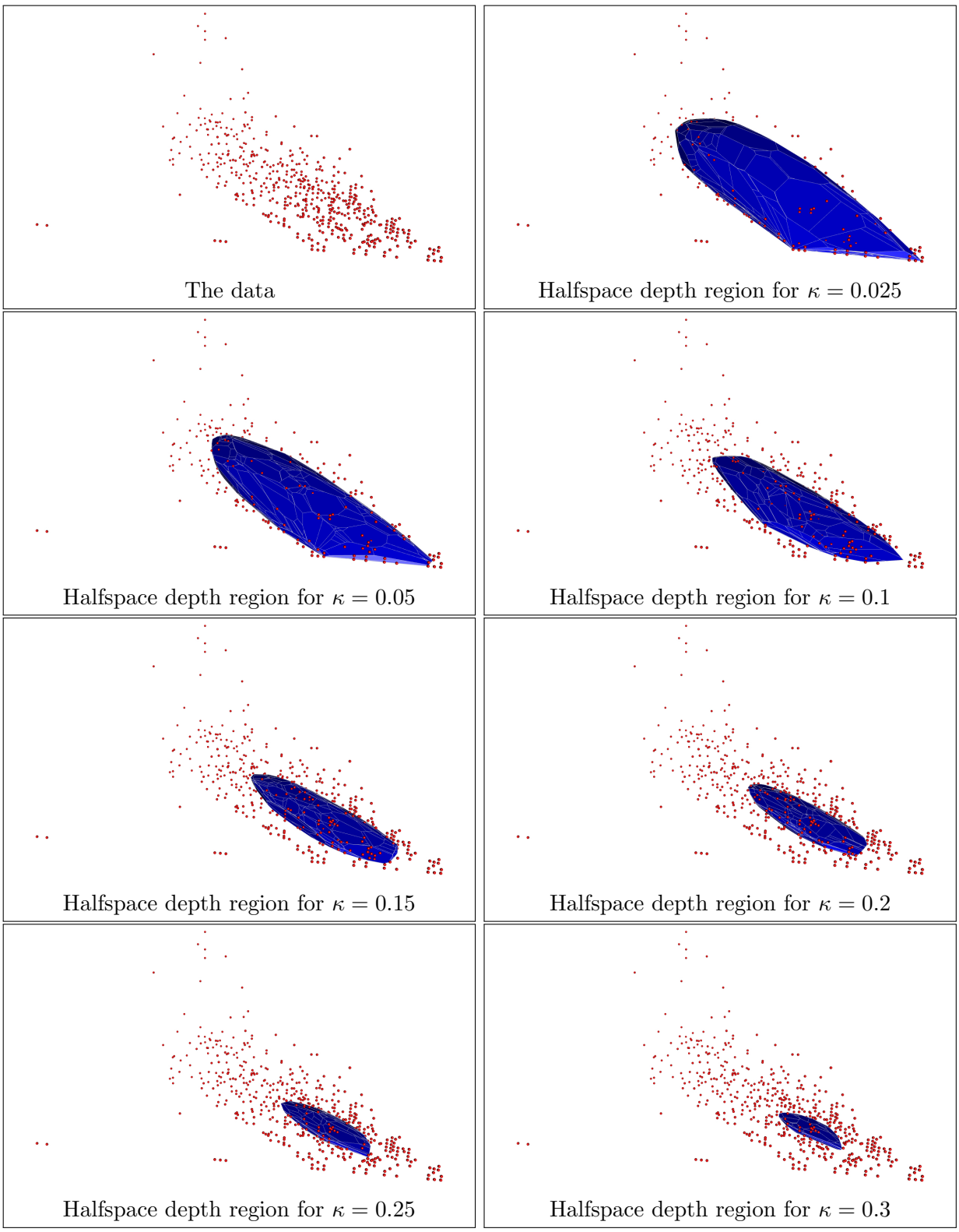


Figure 1.5: *Halfspace κ-regions for the Blood Transfusion data set.*

Yeh et al., 2009). The variables are "Recency – months since last donation", "Frequency – total number of donation", and "Time – months since first donation" (taken from `R`-package `ddalpha`; Pokotylo et al., 2019, 2020). (A fourth variable "Monetary – total blood donated

in c.c." has been removed as it is highly correlated with the third one; see Li et al. (2012).) As one can see from Figure 1.5, a few contours represent the geometry of the data set (`R`-package `rgl` used for visualization; Adler et al., 2021). Further, Table 1.3 indicates that the Algorithm 1.5 is faster than Algorithm 1.4 due to a smaller number of processed ridges and that most of the found relevant hyperplanes (always coinciding for the two algorithms) are redundant, in particular for deep regions.

Table 1.3: *Ratio of computation times and portion of processed ridges by Algorithm 1.5 compared to Algorithm 1.4, and ratio (n_κ^f/n_κ) of facets of halfspace depth regions over the number of relevant hyperplanes for the Blood Transfusion data set.*

κ	0.025	0.05	0.1	0.15	0.2	0.25	0.3
Times ratio	0.034	0.1	0.27	0.45	0.61	0.76	0.87
Ridges ratio	0.034	0.098	0.26	0.43	0.59	0.74	0.85
n_κ^f/n_κ	0.097	0.029	0.0068	0.0036	0.0022	0.0011	0.00047

1.3.4 Numerical illustrations and comparison

This subsection presents the results of a simulation study that explores computational performance and algorithmic complexity. Results on execution times (Tables 1.4, 1.5, as well as 1.6) are obtained using statistical software `R` on a Macbook Pro laptop possessing processor Intel(R) Core(TM) i7-4980HQ (2.8 GHz) having 16 GB of physical memory and macOS Sierra (Version 10.12.6) operating system. The validation experiment as well as the comparison of computation times (Tables 1.6 and 1.7) have been conducted on the École Nationale de la Statistique et de l'Analyse de l'Information's computing cluster consisting of the Dell Poweredge M820 servers equipped with 8-kernel processors of type Intel(R) Xeon(R) E5-4627 v2 @ (3.3 GHz) and having 384 GB of physical memory.

To start, let us measure the time taken for the execution of Algorithm 1.5. Computation times for several pairs (d, n) and depth levels are indicated in Table 1.4. All the calculations are restricted to those of no more than one hour for obtaining relevant and non-redundant hyperplanes. First, the table demonstrates the applicability of the algorithm even for a substantial number of points (up to 5000) in dimension three (the still visualizable case) and its capability of computing relevant hyperplanes of halfspace depth regions even in dimension nine as well as calculating the exact shape (facets and vertices) of halfspace depth regions up to dimension six. Nevertheless, one observes an exponential increase of computation time, which indicates an eventual computational intractability of Algorithm 1.5 when the number of observations or the dimension of the data become even larger. Further, one can see that in higher dimensions, while relevant hyperplanes can be found, the QHULL algorithm fails in determining the non-redundant ones in a reasonable time. However, in all indicated cases an interior point has been rapidly found in Step 6 of Algorithm 1.5. Note that Algorithm 1.5 improves remarkably over Algorithm 1.4 for smaller values of κ. On the other hand when computing deeper regions, the ridge-by-ridge strategy can be less efficient

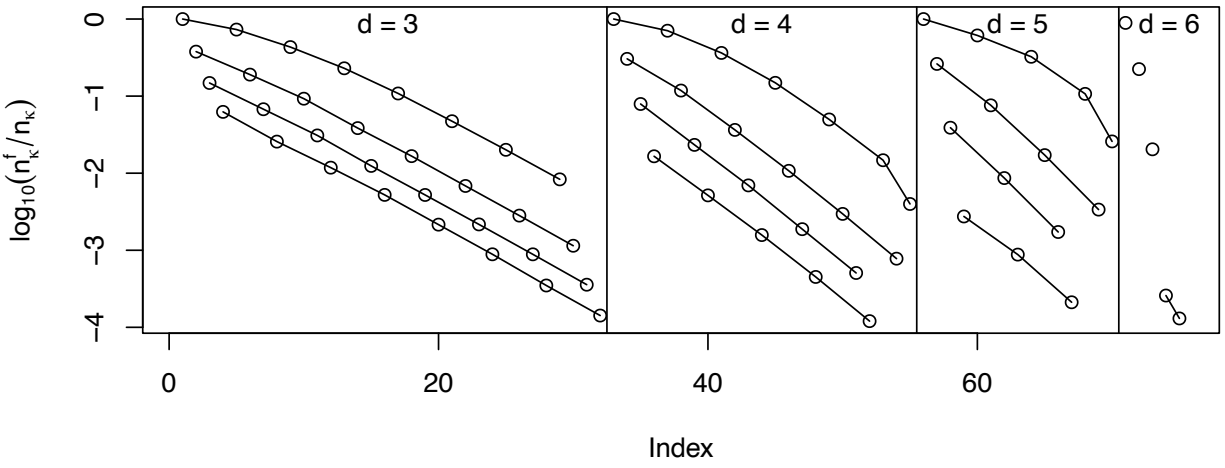


Figure 1.6: *Logarithmized ratio of the number of facets of the halfspace depth region to the number of relevant hyperplanes,* $\log_{10} \frac{n_\kappa^f}{n_\kappa}$. *The index of the points is row-wise(*κ *increasing)/column-wise(*n *increasing) for each dimension* $d = 3, 4, 5, 6$ *of Table 1.5; e.g. sixth point stands for* $(n, \kappa) = (80, 0.1)$ *for* $d = 3$*; lines correspond to rows of Table 1.5.*

and in certain cases even harmful; *cf.* time for $d = 6$, $n = 40$ and $\kappa = 0.3$ (Table 2 of Liu et al., 2019). Such a behavior can be explained by the fact that in this last case a substantial part of $\mathcal{A}$ has to be explored anyway, and the ridge-by-ridge transition takes additional time compared to Algorithm 1.4.

Next, let us take a closer look at the numbers of relevant hyperplanes and facets of a halfspace depth region found by Algorithm 1.5. These results are summarized in Table 1.5. One observes that in most cases the number of facets is substantially smaller than the number of all relevant hyperplanes (see Figure 1.6); *i.e.* most of the computing time is still utilized unnecessarily. (Note that this refers to the general result by Hallin et al. (2010) and the successive algorithms by Paindaveine and Šiman (2012a,b)). Exceptions are limited to relatively small n, and thus do not constitute a computation problem. Further, one can see that the number of facets of the halfspace depth region is large for middle values of κ, while it is rather small for both low and high values of the depth. An extreme example of this phenomenon arises with the multivariate Cauchy distribution, where the outer depth regions can be bordered by rather few facets. Also, note that the Tukey median can constitute an arbitrarily tiny polytope having few facets as well.

The results for $\kappa = 0.025 = 1/40$, $n = 40$, and $d = 6$ attract attention as some of the relevant hyperplanes found – though all belonging to the convex hull – are considered redundant. This happens due to precision merging of the QHULL algorithm in higher dimensions. The number of the relevant hyperplanes itself, on the other hand, shows exponential growth in n and d, which limits the computational feasibility of the Algorithm 1.5. Further, observe that in the experimental settings the upper bound on the facets is never achieved, ranging from close to 0.134 (in a few rare cases) to 0.0000258.

Table 1.4: *Computation times (in seconds) of Algorithm 1.5, taken by finding all relevant hyperplanes (Steps 1 to 5) and by identifying those non-redundant (Steps 6 and 7, in parentheses). For intractable cases, "t" indicates reaching the time limit (of one hour) and "m" reaching the memory limit (of 16 GB).*

d	$\kappa \setminus$	Computation times							
		40	80	160	320	640	1280	2560	5120
3	0.025	0.0009	0.0015	0.0069	0.042	0.31	2.2	18	143
		(0.0005)	(0.001)	(0.0022)	(0.0067)	(0.026)	(0.1)	(0.9)	(7)
	0.1	0.0012	0.0071	0.044	0.31	2.38	19	154	1 270
		(0.0009)	(0.003)	(0.0097)	(0.041)	(0.22)	(1.3)	(8)	(62)
	0.2	0.0027	0.0155	0.11	0.75	5.92	48	389	3 230
		(0.0016)	(0.0056)	(0.023)	(0.1)	(0.55)	(3.4)	(22)	(158)
	0.3	0.0039	0.023	0.15	1.14	9.15	75	602	t
		(0.0024)	(0.0074)	(0.032)	(0.16)	(0.86)	(5.2)	(33)	—
4	0.025	0.001	0.008	0.084	0.9	11	162	2 440	t
		(0.002)	(0.014)	(0.078)	(0.4)	(3)	(18)	(173)	—
	0.1	0.01	0.094	1.28	17	249	3 850	t	—
		(0.017)	(0.07)	(0.42)	(3.6)	(33)	(422)	—	—
	0.2	0.027	0.34	4.36	65	988	t	—	—
		(0.025)	(0.16)	(1.29)	(13)	(132)	—	—	—
	0.3	0.046	0.62	8.38	121	1 880	—	—	—
		(0.056)	(0.23)	(2.28)	(23)	(294)	—	—	—
5	0.025	0.005	0.05	0.7	14	322	t	—	—
		(0.02)	(7.23)	(28.6)	(340)	(1 990)	—	—	—
	0.1	0.06	1.16	26.2	695	t	—	—	—
		(1)	(8.25)	(81.3)	(590)	—	—	—	—
	0.2	0.24	5.55	149	t	—	—	—	—
		(0.51)	(5.87)	(88.5)	—	—	—	—	—
	0.3	0.45	11.9	340	—	—	—	—	—
		(2.39)	(6.97)	(154)	—	—	—	—	—
6	0.025	0.02	0.3	6	201	m	—	—	—
		(1.8)	(t)	(t)	(t)	—	—	—	—
	0.1	0.3	12	497	m	—	—	—	—
		(1 630)	(t)	(t)	—	—	—	—	—
	0.2	1.9	83	t	—	—	—	—	—
		(35)	(t)	—	—	—	—	—	—
	0.3	4.8	214	—	—	—	—	—	—
		(219)	(1 480)	—	—	—	—	—	—
7	0.025	0.06	1	50	m	—	—	—	—
		(t)	(t)	(t)	—	—	—	—	—
	0.1	2	101	m	—	—	—	—	—
		(t)	(t)	—	—	—	—	—	—
	0.2	11	1 040	—	—	—	—	—	—
		(t)	(t)	—	—	—	—	—	—
8	0.025	0.2	7	m	—	—	—	—	—
		(t)	(t)	—	—	—	—	—	—
	0.1	6	789	—	—	—	—	—	—
		(t)	(t)	—	—	—	—	—	—
	0.2	55	m	—	—	—	—	—	—
		(t)	—	—	—	—	—	—	—
9	0.025	0.6	30	m	—	—	—	—	—
		(t)	(t)	—	—	—	—	—	—
	0.1	25	m	—	—	—	—	—	—
		(t)	—	—	—	—	—	—	—
	0.2	259	—	—	—	—	—	—	—
		(t)	—	—	—	—	—	—	—

Table 1.5: *Number of relevant hyperplanes and those non-redundant (in parentheses) delivered by Algorithm 1.5. For intractable cases, "t" indicates reaching the time limit (of one hour) and "m" reaching the memory limit (of 16 GB).*

d	κ \	Number of hyperplanes							
		40	80	160	320	640	1280	2560	5120
3	0.025	29	95	369	1 380	5 280	20 400	82 000	326 000
		(29)	(69)	(160)	(318)	(571)	(965)	(1 650)	(2 710)
	0.1	185	753	2 890	11 500	46 000	184 000	732 000	2 930 000
		(70)	(143)	(268)	(441)	(765)	(1 250)	(2 060)	(3 350)
	0.2	449	1 830	7 340	29 100	116 000	467 000	1 860 000	7 460 000
		(66)	(124)	(227)	(362)	(606)	(1 010)	(1 650)	(2 670)
	0.3	677	2 720	11 200	44 700	181 000	719 000	2 880 000	t
		(42)	(70)	(132)	(234)	(388)	(637)	(1 000)	—
4	0.025	100	510	3 250	21 900	153 000	1 160 000	9 020 000	t
		(100)	(360)	(1 180)	(3 250)	(7 620)	(17 100)	(36 000)	—
	0.1	1 130	8 180	61 600	482 000	3 780 000	30 100 000	t	—
		(343)	(966)	(2 250)	(5 170)	(11 200)	(23 400)	—	—
	0.2	3 810	30 000	236 000	1 890 000	15 100 000	t	—	—
		(301)	(700)	(1 640)	(3 550)	(7 680)	—	—	—
	0.3	6 750	55 500	448 000	3 580 000	28 800 000	—	—	—
		(112)	(286)	(705)	(1 610)	(3 470)	—	—	—
5	0.025	328	2 550	24 400	272 000	3 550 000	t	—	—
		(328)	(1 570)	(7 920)	(29 100)	(91 700)	—	—	—
	0.1	5 850	76 100	1 060 000	15 700 000	t	—	—	—
		(1 530)	(5 780)	(18 200)	(52 800)	—	—	—	—
	0.2	26 100	396 000	6 140 000	t	—	—	—	—
		(1 020)	(3 430)	(10 600)	—	—	—	—	—
	0.3	52 200	863 000	13 900 000	—	—	—	—	—
		(143)	(758)	(2 950)	—	—	—	—	—
6	0.025	968	11 900	169 000	3 110 000	m	—	—	—
		(864)	(t)	(t)	(t)	—	—	—	—
	0.1	25 200	603 000	15 200 000	m	—	—	—	—
		(5 650)	(t)	(t)	—	—	—	—	—
	0.2	143 000	4 320 000	t	—	—	—	—	—
		(2 930)	(t)	—	—	—	—	—	—
	0.3	327 000	11 000 000	—	—	—	—	—	—
		(85)	(1 430)	—	—	—	—	—	—
7	0.025	2 630	48 400	1 070 000	m	—	—	—	—
		(t)	(t)	(t)	—	—	—	—	—
	0.1	95 300	3 980 000	m	—	—	—	—	—
		(t)	(t)	—	—	—	—	—	—
	0.2	678 000	39 200 000	—	—	—	—	—	—
		(t)	(t)	—	—	—	—	—	—
8	0.025	6 930	189 000	m	—	—	—	—	—
		(t)	(t)	—	—	—	—	—	—
	0.1	320 000	24 400 000	—	—	—	—	—	—
		(t)	(t)	—	—	—	—	—	—
	0.2	2 790 000	m	—	—	—	—	—	—
		(t)	—	—	—	—	—	—	—
9	0.025	18 800	670 000	m	—	—	—	—	—
		(t)	(t)	—	—	—	—	—	—
	0.1	1 040 000	m	—	—	—	—	—	—
		(t)	—	—	—	—	—	—	—
	0.2	10 300 000	—	—	—	—	—	—	—
		(t)	—	—	—	—	—	—	—

Time and complexity comparison

Algorithm 1.5 has time complexity $O\big(n^d \log(n)\big)$. For the algorithm developed by Paindaveine and Šiman (2012a), henceforth *Algorithm PSa*, time complexity amounts to $O(n^{d+1})$. The authors conjecture that for a random $\kappa \in (0, 0.5)$ the average time complexity is $O(n^d)$. The simulation study of the procedure given in Paindaveine and Šiman (2012b), henceforth *Algorithm PSb*, points to a similar complexity. As Algorithm 1.5 is of a different nature than PSa and PSb, and the complexities disregard κ, they cannot be directly compared.

To gain further insights, let us compare (average) computation times of the three algorithms. Table 1.6 indicates how much faster Algorithm 1.5 is than the two competitors (while always yielding the same relevant hyperplanes). At first sight, Algorithm 1.5 is faster by several orders. However, the speed gain decreases with n, suggesting that Algorithm 1.5 has higher time complexity. To explain this last observation, see Table 1.7. First, the (average) portion of ridges processed by Algorithm 1.5 compared to Algorithm 1.4 is indicated, where it is constant and equal to $\binom{n}{d-1}$. These numbers suggest that Algorithm 1.5 can be much more efficient than Algorithm 1.4, as it normally processes fewer ridges and this difference increases with decreasing κ. Second, the ratio of the number of the relevant hyperplanes found by Algorithm PSa to the number of ridges processed by Algorithm 1.5 is reported. As the number of relevant hyperplanes corresponds to the number of cones processed by Algorithm PSa, this ratio indicates that the number of the elements (ridges for Algorithm 1.5, cones for Algorithm PSa) processed by both Algorithm 1.5 and Algorithm PSa is of the same order. Nevertheless, processing one element has time complexity $O\big(n \log(n)\big)$ in Algorithm 1.5 and $O(n)$ in Algorithm PSa. For Algorithm PSb a similar logic applies. On the other hand, Table 1.6 shows that Algorithm 1.5 is faster even for large n. This is due to the simplicity of its operations and the lightness of the involved structures. In particular, neither post-optimization nor the QHULL algorithm are used when searching for relevant hyperplanes, and a ridge is saved as a simple combination of data points.

1.3.5 Computing Tukey median

The *Tukey median* is the gravity center of the Tukey median set, that is, the halfspace depth region having maximum depth. With the above algorithms it is possible to compute halfspace depth regions at any given depth level. However the maximum halfspace depth, κ^*, depends on the data and has to be determined from them. For data in general position it is known (Liu et al., 2020) that

$$\frac{1}{n}\left\lceil \frac{n}{d+1} \right\rceil \leq \kappa^* \leq \frac{1}{n}\left\lfloor \frac{n-d+2}{2} \right\rfloor . \tag{1.11}$$

In two-dimensional space, Rousseeuw and Ruts (1998) have developed an algorithm, which has time complexity $O\big(n^2 \log^2(n)\big)$. Their algorithm employs a bisection strategy that starts with a lower and an upper bound on κ^* and updates these bounds until they coincide. In updating, the algorithm by Rousseeuw and Ruts (1998) calculates the mean $\overline{\kappa}$ of the two active bounds and checks whether $D^{\overline{\kappa}}_{\mathrm{H}}(\boldsymbol{X})$ exists. If yes, the lower bound is changed to $\overline{\kappa}$; if

Table 1.6: *Ratios of average computation times of Algorithms PSa and PSb over that of Algorithm 1.5. (10 repetitions, standard multivariate normal distribution, implementation in R-package* `modQR` *(Šiman and Boček, 2019).)*

d	κ	Algorithm \	Computation time ratios							
			40	80	160	320	640	1280	2560	5120
3	0.025	PSa	4 920	8 340	6 490	2 980	2 060	676	268	149
		PSb	3 760	7 240	7 230	4 100	2 370	600	269	157
	0.1	PSa	23 500	20 100	9 670	5 090	2 190	1 000	420	—
		PSb	22 900	22 000	17 300	6 300	2 540	982	361	—
	0.2	PSa	25 300	25 400	11 100	4 410	2 450	924	—	—
		PSb	26 100	29 200	17 800	5 950	2 080	784	—	—
	0.3	PSa	21 200	27 500	11 400	4 910	2 400	1 030	—	—
		PSb	23 400	35 500	11 900	5 050	2 240	883	—	—
4	0.025	PSa	16 500	13 700	6 320	3 510	1 830	1 620	—	—
		PSb	7 580	11 700	9 860	2 860	1 760	843	—	—
	0.1	PSa	66 700	22 300	16 200	6 960	—	—	—	—
		PSb	57 500	28 300	19 600	4 850	—	—	—	—
	0.2	PSa	57 300	36 000	19 000	—	—	—	—	—
		PSb	57 500	35 400	10 100	—	—	—	—	—
	0.3	PSa	55 700	40 600	17 400	—	—	—	—	—
		PSb	62 100	40 800	12 600	—	—	—	—	—

Table 1.7: *Portion of ridges processed by Algorithm 1.5 compared to Algorithm 1.4, and (in parentheses) the ratio of the number of relevant hyperplanes found by Algorithm PSa to the number of ridges processed by Algorithm 1.5.*

d	κ	Portion of processed ridges (non-redundant hyperplanes)							
		40	80	160	320	640	1280	2560	5120
3	0.025	0.059	0.044	0.035	0.029	0.026	0.025	0.023	0.023
		(0.68)	(0.92)	(1.64)	(2.18)	(2.64)	(2.9)	(3.08)	(3.26)
	0.1	0.29	0.26	0.23	0.22	0.21	0.2	0.2	—
		(1.69)	(2.28)	(2.68)	(2.98)	(3.18)	(3.29)	(3.38)	—
	0.2	0.62	0.58	0.55	0.53	0.52	0.51	—	—
		(2.41)	(2.76)	(3)	(3.23)	(3.3)	(3.35)	—	—
	0.3	0.87	0.84	0.81	0.79	0.78	0.77	—	—
		(2.77)	(2.99)	(3.13)	(3.27)	(3.35)	(3.41)	—	—
4	0.025	0.02	0.012	0.0075	0.0055	0.0045	0.004	—	—
		(0.47)	(0.72)	(1.17)	(1.89)	(2.51)	(2.98)	—	—
	0.1	0.18	0.14	0.12	0.1	—	—	—	—
		(1.25)	(2.01)	(2.63)	(3.09)	—	—	—	—
	0.2	0.52	0.45	0.41	—	—	—	—	—
		(2.09)	(2.7)	(3.2)	—	—	—	—	—
	0.3	0.84	0.78	0.74	—	—	—	—	—
		(2.75)	(3.18)	(3.46)	—	—	—	—	—

no, the upper bound becomes $\overline{\kappa}$. The bisection approach extends easily to data of dimension greater than 2. With the new Algorithm 1.5 at hand, it is natural to investigate its possible use and accuracy in computing higher-dimensional Tukey medians.

In searching for the maximum level of halfspace depth, a modified bisection strategy is employed. Given a halfspace depth region $D^{\kappa}_{\mathrm{H}}(\boldsymbol{X})$ at some level κ, consider its barycenter. If the region is nonempty, its barycenter is more central than its boundary points, and thus may have a larger depth value. Observe that computing the depth of a single point is computationally cheaper (*i.e.* has lower time complexity) than computing a trimmed region. Hence, after having checked that at a given depth level κ the region $D^{\kappa}_{\mathrm{H}}(\boldsymbol{X})$ is nonempty, one may further compute the barycenter of $D^{\kappa}_{\mathrm{H}}(\boldsymbol{X})$ to possibly come closer to the depth maximum. This motivates us to construct the following algorithm. In the algorithm, $\texttt{HD}(\boldsymbol{x}\,;\,\boldsymbol{x}_1,\cdots,\boldsymbol{x}_n)$ stands for any procedure that computes the halfspace depth of a point $\boldsymbol{x}\in\mathbb{R}^d$ w.r.t. data $\boldsymbol{x}_1,\cdots,\boldsymbol{x}_n\subset\mathbb{R}^d$, like those in the previous Section 1.2 or Liu (2017). Further, $\texttt{Alg1}(\boldsymbol{x}\,;\,\boldsymbol{x}_1,\cdots,\boldsymbol{x}_n;\kappa;\epsilon)$ signifies a halfspace depth region resulting from Algorithm 1.5.

Algorithm 1.6 (Algorithm for computing the Tukey median)

Input: $\boldsymbol{x}_1,\cdots,\boldsymbol{x}_n\subset\mathbb{R}^d$, ϵ.

Step 1. Initialize bounds on κ^*:

(a) *Compute* $\boldsymbol{x}_0=\big(med(\boldsymbol{x}_{11},\cdots,\boldsymbol{x}_{n1}),med(\boldsymbol{x}_{12},\cdots,\boldsymbol{x}_{n2}),\cdots,med(\boldsymbol{x}_{1d},\cdots,\boldsymbol{x}_{nd})\big)^{\top}$.

(b) *Compute* $d_0=HD(\boldsymbol{x}_0\,;\,\boldsymbol{x}_1,\cdots,\boldsymbol{x}_n)$.

(c) *Set* $\kappa_{low}=\max\big\{\frac{1}{n}\lceil\frac{n}{d+1}\rceil,d_0\big\}$, $\kappa_{up}=\frac{1}{n}\lfloor\frac{n-d+2}{2}\rfloor+\frac{1}{n}$.

Step 2. Update bounds:

Let $\overline{\kappa}=\frac{1}{n}\lfloor\frac{n(\kappa_{low}+\kappa_{up})}{2}\rfloor$, *and compute the region* $D^{\overline{\kappa}}_{H}(\boldsymbol{x}_1,\cdots,\boldsymbol{x}_n)=$
$=Alg1(\boldsymbol{x}_1,\cdots,\boldsymbol{x}_n\,;\,\overline{\kappa}\,;\,\epsilon)$.

(a) *If* $D^{\overline{\kappa}}_{H}(\boldsymbol{x}_1,\cdots,\boldsymbol{x}_n)$ *does not exist (that is,* Algorithm 1.5 *stops at its* Step 6*), then set* $\kappa_{up}=\overline{\kappa}$.

(b) *If* $D^{\overline{\kappa}}_{H}(\boldsymbol{x}_1,\cdots,\boldsymbol{x}_n)$ *exists, then calculate the barycenter* $\mathbf{c}$ *of* $D^{\overline{\kappa}}_{H}(\boldsymbol{x}_1,\cdots,\boldsymbol{x}_n)$ *and set* $\kappa_{low}=HD(\boldsymbol{c}\,;\,\boldsymbol{x}_1,\cdots,\boldsymbol{x}_n)$.

(c) *If* $\kappa_{low}<\kappa_{up}-\frac{1}{n}$, *then repeat* **Step 2**, *else stop.*

Output: $Alg1(\boldsymbol{x}_1,\cdots,\boldsymbol{x}_n\,;\,\kappa_{low}\,;\,\epsilon)$.

Step 1c initializes the lower bound either according to (1.11) or with the depth of the coordinate-wise median, which is cheaply computed and often has a rather high depth value. In Step 2b, the depth of the barycenter of the halfspace depth region is computed (in case it exists) to obtain a higher value for the lower bound, which substantially contributes to the speed of Algorithm 1.6.

Simulation study presented in Section 5 of Liu et al. (2019) illustrates that Algorithm 1.6 can be faster than using the bisection strategy as it has been done by Rousseeuw and Ruts (1998) for bivariate data (their median-computing algorithm has time complexity

Table 1.8: *An artificial data set for illustrating the different location of the sample mean, the coordinate-wise median and the Tukey median.*

#	x_1	x_2	x_3
1	1	0	0
2	0	1	0
3	0	0	1
4	1.5	1.5	1.5
5	0.309	0.287	0.654
6	0.733	0.04	0.316
7	0.159	0.305	0.558
8	0.056	0.19	0.913
9	0.517	0.533	0.192
10	1.012	0.059	0.099
11	0.118	0.164	0.92
12	0.175	0.919	0.222
13	0.24	0.454	0.17
14	0.906	0.056	0.12

location estimators	x_1	x_2	x_3	depth
mean	0.480	0.393	0.476	$\frac{1}{14}$
coordinate-wise median	0.275	0.239	0.269	0
Tukey median	0.454	0.27	0.413	$\frac{4}{14}$

$O\big(n^2 \log^2(n)\big)$), with this tendency more expressed when considering more data and data stemming from a skewed distribution. Therefore, the new Algorithm 1.6 is recommended for larger sample sizes n, while the bisection approach is to be preferred for smaller n.

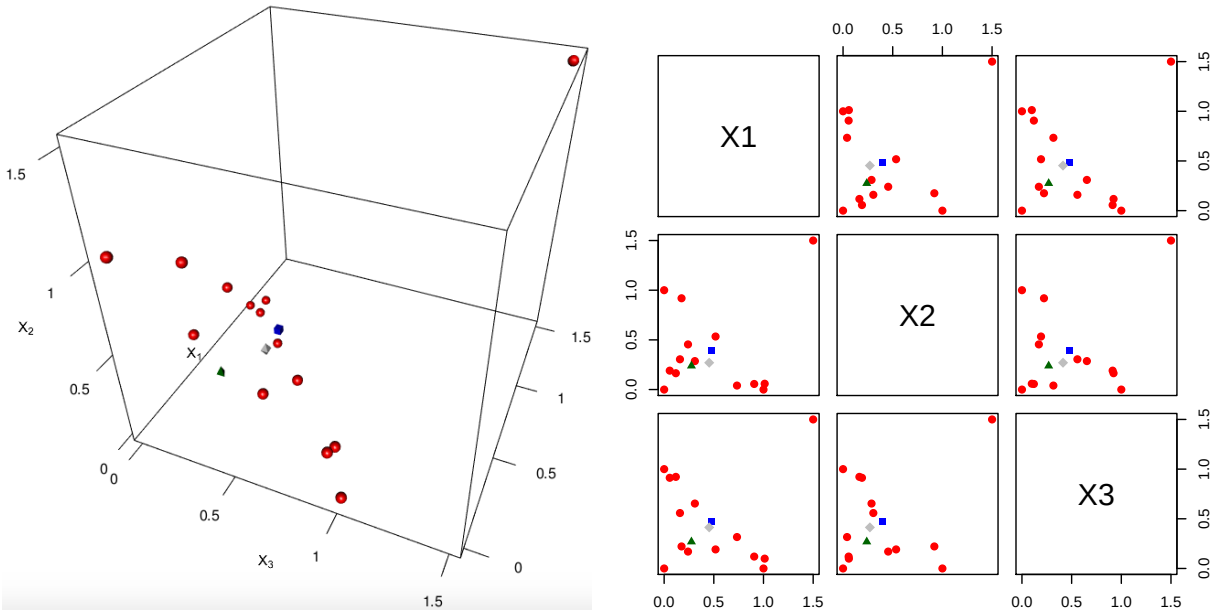


Figure 1.7: *Data (red spheres), with mean (blue cube), coordinate-wise median (green tetrahedron) and Tukey median (gray rhombus); see Table 1.8.*

The positions of the sample mean, the coordinate-wise median and the Tukey median can be quite different. To illustrate this, let us start with a three-dimensional artificial data set in Table 1.8, which is inspired by Chapter 7.1a of Rousseeuw and Leroy (1987). The data is constructed in the following way: The first three points correspond to the three canonical unit vectors. The fourth point has coordinates $(1.5, 1.5, 1.5)^\top$; it represents an outlier. Further ten points have coordinates $Z + U \cdot (\frac{1}{\sqrt{3}}, \frac{1}{\sqrt{3}}, \frac{1}{\sqrt{3}})^\top$, where Z is a random vector having a Dirichlet distribution with parameters $(1, 1, 1)$ and U is a random variable uniformly distributed on $[-\frac{1}{4}, \frac{1}{4}]$. The sample mean, the coordinate-wise median and the Tukey median are given in Table 1.8, together with their depths in the data cloud. Table 1.8 indicates that the coordinate-wise median lies outside the convex hull of the data set, while the sample mean – being sensitive to the outlier – lies outside the convex hull of the main data (= data without the outlier at $(1.5, 1.5, 1.5)^\top$), which is visualized in Figure 1.7. Moreover, the mean and the coordinate-wise median are found on different sides. On the other hand, the Tukey median lies in the middle of the convex hull of the main data.

1.3.6 Further developments

Computational reasoning employed in this section appears to be useful for another task, namely nonparametric efficiency analysis. Application of the convex hull algorithm allows to reduce the constraints of the linear program (when this appears to be necessary), which dramatically reduces the computational cost of the procedure. These advances are published as Badunenko and M. (2016) and are mentioned here in Section A.3.

For completeness, it is important to mention the following developments of the few pas months. Recent collaboration with Vít Fojtík, Petra Laketa, and Stanislav Nagy from the Charles University in Prague has allowed to construct data samples for which Step 2 of Algorithm 1.5 is insufficient to find all relevant hyperplanes. This can happen because Steps 3–5 of Algorithm 1.5 cover one so-called "orbit" of depth κ, and thus leave a possibility for existence of another "orbit" not connected to the current one by the search of neighboring ridges in Step 4. In this case, only upper bound (envelop) on the halfspace depth region is found. For example, it can happen that $d + 1$ observational hyperplanes, which form a simplex, are missed by Algorithm 1.5. Although such configurations of data points should be artificially constructed and (as first investigations show) are rather unstable to even small perturbations, and never appeared neither during the validation (see Section 4.1 of Liu et al., 2019) nor during the simulation study (see Section 1.3.4), they are still possible. The improvement can be expected when altering Step 2 of Algorithm 1.5 following the—computationally more expensive—idea of the third bullet point at the end of Section 1.3.2.

The reader is referred to Fojtík et al. (2022) for more details.

1.4 Approximation of projection depths

Exact procedures have been implemented in the `R`-package `ddalpha` to calculate the following nine data depths: Mahalanobis, $\mathbb{L}_2$, spatial, zonoid, lens, onion, halfspace, simplicial volume,

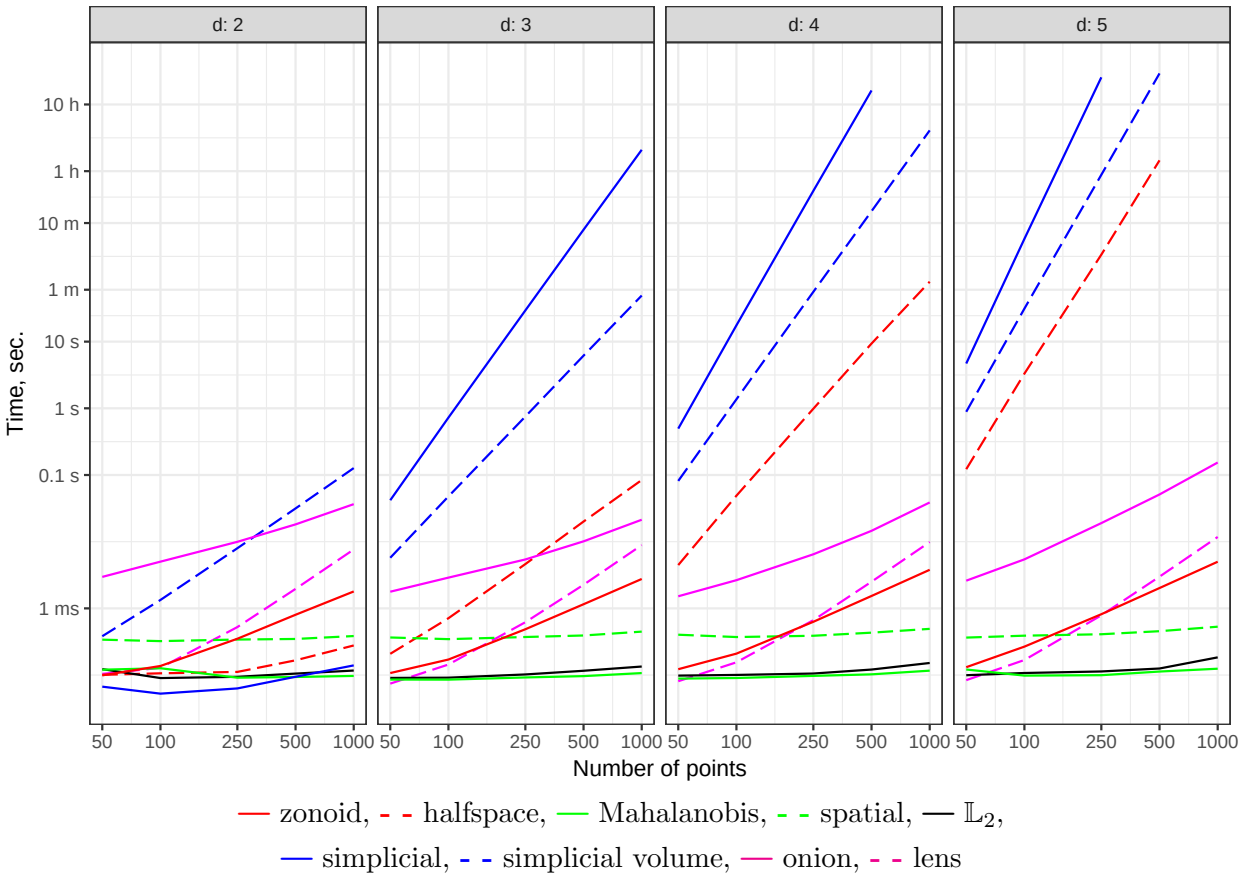


Figure 1.8: *Calculation time of various depth functions on double logarithmic scale (sample size n and time t).*

and simplicial depth. Note that for most of these depths the naïve direct calculation is not feasible, and more sophisticated procedures of lower computational complexity have to be used.

Figure 1.8 exhibits average computation times (when computing depth of a single point w.r.t. a sample of size n) for each of the depths, depending on sample size n and dimension d, where n runs up to 1000 and $d = 2, 3, 4, 5$. Given n and d, 30 samples have been drawn, depth has been calculated for 25 points of each sample, and an average has been taken over these 25 points and 30 samples.

All calculations have been performed by means of the `R`-package `ddalpha` (Pokotylo et al., 2019) on a machine having processor Intel(R) Core(TM) i7-4980HQ (2.8 GHz) with 16 GB of physical memory and macOS Sierra (Version 10.13.4) operating system.

As is seen from Figure 1.8, while the depth notions differ greatly in their computation times, some of them are sharply increasing with n and d. (In each panel, times and sample sizes are measured on a double logarithmic scale.) Table 1.9 lists the complexities of the various algorithms, including references to the literature.

One observes a trade-off between statistical properties and computational complexity. For example most attractive depths, *i.e.* those being robust and non-parametrically (not involving estimation of the covariance matrix to achieve this property) affine invariant,

Table 1.9: *Time complexities for exact and approximate computation of several depths (of a given point in $\mathbb{R}^d$ regarding a sample). For approximate computation, k stands for the number of random directions for halfspace, projection and zonoid depth, resp. for the number of considered simplices for simplicial and simplicial volume depth, resp. for the number of considered pairs of points for β-skeleton depth.*

Depth notion	Exact	Approximate
Mahalanobis	$O(n)$	—
$\mathbb{L}_p$	$O(n)$	—
halfspace	$O\big(n^{d-1}\log(n)\big)$, $O(n^d)$	$O(kn)$
	Rousseeuw (1998)	Dyckerhoff (2004)
	Dyckerhoff and M. (2016)	M. et al. (2015)
projection	$O(n^d)$, Liu and Zuo (2014b)	$O(kn)$, Dyckerhoff (2004)
simplicial	$O(n^{d+1})$	$O(k)$ number-approx.
		$O(n^{d+1})$ portion-approx.
simplicial volume	$O(n^d)$	$O(k)$ number-approx.
		$O(n^d)$ portion-approx.
zonoid	Dyckerhoff et al. (1996)	$O(kn)$, Dyckerhoff (2004)
spatial	$O(n)$	—
β-skeleton	$O(n^2)$	$O(k)$
onion	$O(n^{\lfloor d/2\rfloor}/d^d)$, Barber et al. (1996)	—

like halfspace or projection depth, have their computational time complexity exponentially increasing with the dimension d.

Clearly, if an exact procedure is available and time and memory space allow, the depth of a point should be calculated by the exact procedure. This is the obvious "gold standard". However, it may be non-feasible in practice if d and/or n are too large or if the depth has to be very often evaluated

- as in classification, bootstrap or permutation procedures, or
- when a whole central region is calculated.

1.4.1 The class of projection depths

Dyckerhoff (2004) described a class of depths which satisfy the weak *projection property*. Out of the existing variety, these can be calculated in a universal way by minimizing depth in univariate projections. For this, in each direction, only the univariate depth of n observations should be computed, which often has computational time complexity only $O(n)$. This class of depths constitutes the focus of the current section. Among the depths that satisfy the projection property, Mahalanobis (Mahalanobis, 1936), zonoid (Koshevoy and

Mosler, 1997), halfspace (Tukey, 1975), projection (Zuo and Serfling, 2000) and asymmetric projection (Dyckerhoff, 2004) depths are considered here.

Several authors have already applied approximation techniques to the (sample) depth computation, notably for the halfspace depth and projection depth. Purely random methods seem most intuitive and have been used, *e.g.*, by Cuesta-Albertos and Nieto-Reyes (2008) for the halfspace depth and Liu and Zuo (2014b) for the projection depth. More sophisticated procedures were proposed as well. Rousseeuw (1998) minimize univariate halfspace depth after projecting data onto directions based on a random combination of sample points. Chen et al. (2013) first project data on subspaces orthogonal to linear spans of $0 < k \leq d$ points from the sample, and then employ a brute-force approximation of the halfspace depth in these projections. M. et al. (2015) accelerate the problem of approximation of the halfspace depth of many points (and of the sample itself) w.r.t. a sample by preliminary sorting the data in all projections. Dutta and Ghosh (2012) use the Nelder-Mead algorithm (Nelder and Mead, 1965, run in $\mathbb{R}^d$) for approximation of the projection depth.

The projection property is defined as follows:

Definition 1.8 *(Dyckerhoff, 2004) A depth D satisfies the (weak)* projection property*, if for each point $\boldsymbol{y} \in \mathbb{R}^d$ and each sample $\boldsymbol{X}$ it holds:*

$$D(\boldsymbol{y}|\boldsymbol{X}) = \inf\{D(\langle \boldsymbol{u}, \boldsymbol{y}\rangle \,|\, \langle \boldsymbol{u}, \boldsymbol{X}\rangle)|\boldsymbol{u} \in \mathbb{S}^{d-1}\}\,,$$

where $\langle \boldsymbol{u}, \boldsymbol{X}\rangle$ stands for observation-wise inner product.

If a depth satisfies the projection property, its computation is equivalent to *minimization* of the (possibly non-differentiable) objective function

$$\phi_{\boldsymbol{z},\boldsymbol{X}} : \mathbb{S}^{d-1} \to \mathbb{R}_0^+,\ \boldsymbol{u} \mapsto D(\langle \boldsymbol{u}, \boldsymbol{z}\rangle \,|\, \langle \boldsymbol{u}, \boldsymbol{X}\rangle)\,.$$

Therefore, classical optimization methods can be used to compute the depth. Since this is a minimization problem, clearly, the obtained depth value will be the *upper bound* on the true (sample) depth. Particular attention should be paid here to the domain of the function $\phi_{\boldsymbol{z},\boldsymbol{X}}$ which is the the unit sphere $\mathbb{S}^{d-1}$. Of course, the function $\phi_{\boldsymbol{z},\boldsymbol{X}}$ could be easily extended to a function $\tilde{\phi}_{\boldsymbol{z},\boldsymbol{X}}$ defined on $\mathbb{R}^d \setminus \{\boldsymbol{0}\}$ by setting $\tilde{\phi}_{\boldsymbol{z},\boldsymbol{X}}(\boldsymbol{u}) = D(\langle \boldsymbol{u}, \boldsymbol{z}\rangle \,|\, \langle \boldsymbol{u}, \boldsymbol{X}\rangle)$. However, because of the affine invariance of the depth, $\tilde{\phi}_{\boldsymbol{z},\boldsymbol{X}}$ is constant on lines through the origin. Therefore, it should be advantageous to use optimization methods which are adapted to the particular domain $\mathbb{S}^{d-1}$. This claim will be confirmed in the subsequent simulation study.

Univariate depth on the geodesic sphere

Let us take a closer look at the class of the depths described above and ask the question: how possible a good approximation of these depths can be?

To get some insights in the behavior of the objective function $\phi_{\boldsymbol{z},\boldsymbol{X}}$, *e.g.*, whether there are local minima or not, several plots of $\phi_{\boldsymbol{z},\boldsymbol{X}}$ are presented in the case $d = 3$ for different depths and data sets in Figure 1.9. The figures suggest that (at least for common distributions and in the case $d = 3$) local minima seem not to be a major problem.

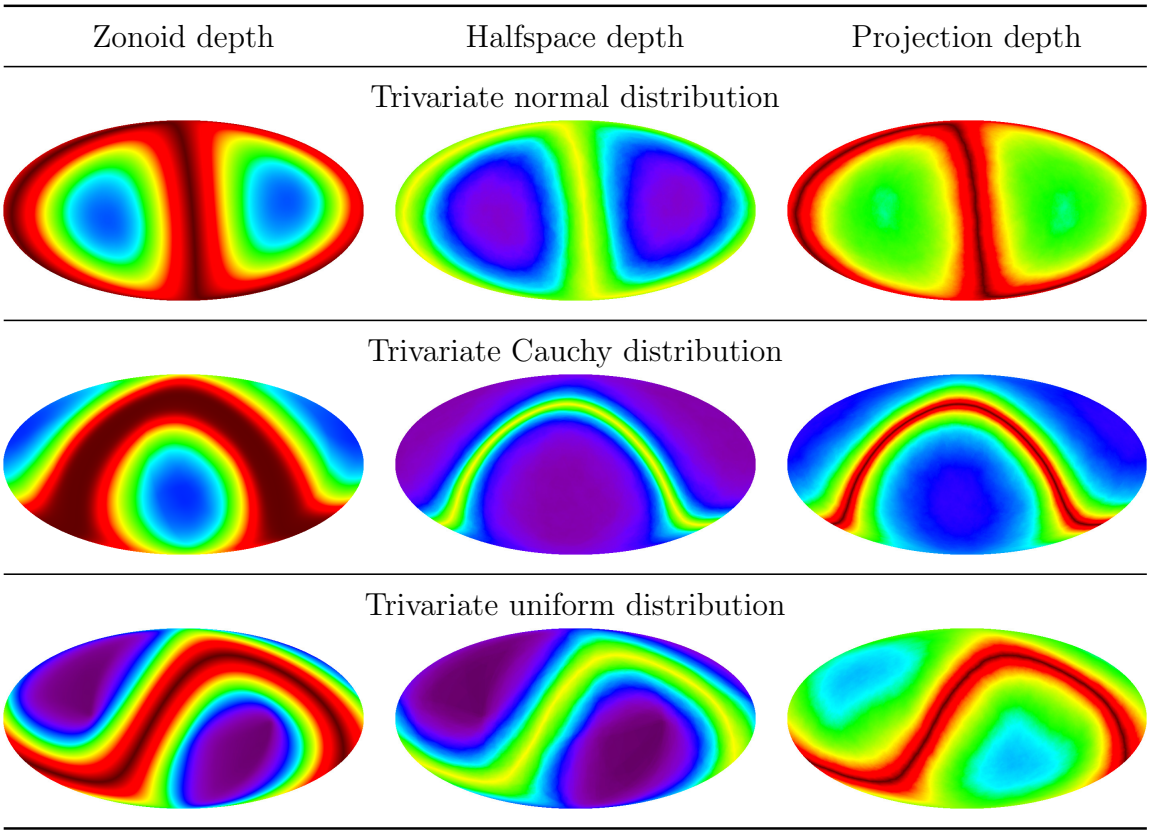


Figure 1.9: *The map $\phi_{\boldsymbol{z},\boldsymbol{X}}$ for trivariate data. A total of $n = 1000$ data points were simulated from a trivariate distribution. The univariate depth (of a single randomly chosen point $\boldsymbol{z}$) in direction $\boldsymbol{u}$ is shown on a color scale from violet (low univariate depth) to dark red (high univariate depth). The sphere $\mathbb{S}^2$ is mapped on the plane using the so-called Mollweide projection, see Snyder (1987).*

A further important observation is the following. All of the above depths are bounded above by unity. Therefore, the range of $\phi_{\boldsymbol{z},\boldsymbol{X}}$ depends on the depth of $\boldsymbol{z}$, *i.e.*, if $D(\boldsymbol{z} \mid \boldsymbol{X}) = c_0$, then the range of $\phi_{\boldsymbol{z},\boldsymbol{X}}$ is a subset of $[c_0, 1]$. The larger the depth of $\boldsymbol{z}$, the smaller is the variation of $\phi_{\boldsymbol{z},\boldsymbol{X}}$. In the extreme case that $c_0 = 1$ the function $\phi_{\boldsymbol{z},\boldsymbol{X}}$ is constant, $\phi_{\boldsymbol{z},\boldsymbol{X}}(\boldsymbol{u}) = 1$ for all $\boldsymbol{u} \in \mathbb{S}^{d-1}$. Therefore, evaluating the univariate depth for a single direction only already gives the exact multivariate depth. In that sense, it should be easier to compute the depth of a point with high depth.

1.4.2 Simulation study

The projection property provides a powerful tool for depth approximation. To obtain an efficient approximation method, 8 different (those existing in the literature adapted to the geometry of the unit sphere and new ones) algorithms are compared in a comprehensive simulation study comprising 9 distributions in 4 different dimensions, for 5 depth notions, using 4 quality measures (2 ordinal and 2 cardinal ones). To assure the honest comparison, the algorithms are first fine-tuned over predefined parameter ranges, using 6 distributions and 2 (ordinal) measures; see Section 4.2 of Dyckerhoff et al. (2021) for details. The 3 resting

distributions and 2 (cardinal) quality measures (that were not used during fine-tuning) are included in the final comparative study only.

Distributional setting

The simulation study is based on the depth computation of a point $\boldsymbol{z}$ w.r.t. a sample $\boldsymbol{X}$ of n i.i.d. d-variate points. The point $\boldsymbol{z}$ is taken to be the average of 10 arbitrary points of the sample. This guarantees that it belongs to the convex hull of the data so that the depth is always strictly positive, but also does not place it too deep in the data set to preserve the random nature of the choice of $\boldsymbol{z}$ (since only 10 points out of 1000 are averaged). The following *nine distributions* are used (due to the affine-invariance of the considered depths no correlation structure is introduced):

- the standard normal distribution $\mathcal{N}(\mathbf{0}_d, \boldsymbol{I}_d)$;
- the spherically-symmetric Student t_5 distribution;
- the spherically-symmetric Student t_1 (Cauchy) distribution;
- the uniform distribution on $[0,1]^d$;
- the skewed normal distribution generated in the following way (Azzalini, 2013): let $U \sim \mathcal{U}([0,1])$, $Z \sim \mathcal{N}(\mathbf{0}_d, \boldsymbol{I}_d)$, U and Z stochastically independent, and $\boldsymbol{\delta} \in \mathbb{R}^d$ be a skewness parameter. Then the skewed normal random vector equals

$$X \overset{\mathrm{d}}{=} \begin{cases} Z & \text{if } U \leq \Phi(\boldsymbol{\delta}^\top Z)\,, \\ -Z & \text{if } U > \Phi(\boldsymbol{\delta}^\top Z)\,, \end{cases}$$

  where $\Phi(\cdot)$ is the c.d.f. of the standard normal distribution, $\boldsymbol{\delta} = (5, 0, ..., 0)^\top$, and $\overset{\mathrm{d}}{=}$ denotes equality in distribution;
- a product of d independent exponential distributions (with parameter $\lambda = 1$);
- the hemispherical shell distribution (abbreviated as "Shell") generated as follows: let $(S_1, ..., S_{d-1}, S_d)^\top \overset{\mathrm{d}}{=} S \sim \mathcal{U}(\mathbb{S}^{d-1})$, $U \sim \mathcal{U}([0.9, 1])$, S and U stochastically independent. The random vector stemming from the shell distribution equals

$$X \overset{\mathrm{d}}{=} U \cdot (S_1, ..., S_{d-1}, |S_d|)^\top\,;$$

- the bimodal normal mixture generated as follows: let $B \sim \mathcal{B}(0.5)$, $Z \sim \mathcal{N}(\mathbf{0}_d, \boldsymbol{I}_d)$, B and Z stochastically independent. To model the bimodal normal mixture the following random vector is used:

$$X \overset{\mathrm{d}}{=} \begin{cases} Z + 2\boldsymbol{e}_1 & \text{if } B = 1\,, \\ Z - 2\boldsymbol{e}_1 & \text{if } B = 0\,, \end{cases}$$

  where $\boldsymbol{e}_j$ is the j-th canonical unit vector;

- the multimodal normal mixture generated as follows: let $I \sim \mathcal{U}(\{1, ..., d\})$, $Z \sim \mathcal{N}(\mathbf{0}_d, \boldsymbol{I}_d)$, I and Z stochastically independent, then the considered random vector equals
$$X \stackrel{\mathrm{d}}{=} Z + 3\boldsymbol{e}_I \,.$$

Since all the considered algorithms report an upper bound on the actual depth, for the same $\boldsymbol{z}$ and a data set one can compare them according to the obtained depth values, because lower obtained depth is always closer to the exact value. *Four quality measures* are reported:

- the average rank (AveRank) of the obtained depth approximation (among all considered algorithms) over 1000 runs (the lower the better, with 1 being the best);
- the percentage when the considered algorithm achieved the smallest depth value (among all considered algorithms) (denoted shortly as AveRank) over 1000 runs (the higher the better, with 100% being the best);
- mean absolute error (MAE) is calculated as the difference between the obtained depth approximation and the exact depth, averaged over all $N_{sim} = 1000$ runs;
- mean relative error (MRE) is calculated as MAE divided by the exact depth and averaged again over all $N_{sim} = 1000$ runs.

Note that the computation of the exact depth values is (in reasonable time) possible only for the Mahalanobis depth and the zonoid depth. Therefore, when computing MAE and MRE for the other depths, the exact depth value is replaced by the minimum depth achieved by any of the algorithms to be compared.

The eight algorithms that have been employed are:

- Random search (RS);
- Grid search (GS);
- Refined random search (RRS);
- Refined grid search (RGS);
- Random simplex (RaSi);
- Simulated annealing (SA);
- Coordinate descent (CD);
- Nelder-Mead (NM);

see Section 3 of Dyckerhoff et al. (2021) for a detailed description of the algorithms.

In the simulation study, five mentioned above depth notions are considered: Mahalanobis, zonoid, halfspace, projection, and asymmetric projection depths. An explicit formula makes it unnecessary to approximate the Mahalanobis depth, which is in addition

a quadratic (and thus everywhere smooth) function. Since rather good results for optimization techniques are expected, it is included for a qualitative comparison with random algorithms. Since the zonoid depth can be computed efficiently even in higher dimensions using the algorithm of Dyckerhoff et al. (1996), it is also included as a benchmark.

1.4.3 Presentation of results

After having fixed the parameters of the methods during fine-tuning, $N \approx 100$, 1000, and 10000 projections are considered. Refined grid search (RGS) could not be used for $N \approx 100$ projections since for 10 refinements (the number found optimal during the fine-tuning) only ten projections could be used for the grid in each refinement step which is too small even for $d = 5$. Here, only results for $N \approx 1000$ directions are analyzed, since those for the other numbers of directions are similar. The complete results in graphical and tabular form can be found in Supplementary Materials, Sections 2.1 and 2.2 of Dyckerhoff et al. (2021), respectively.

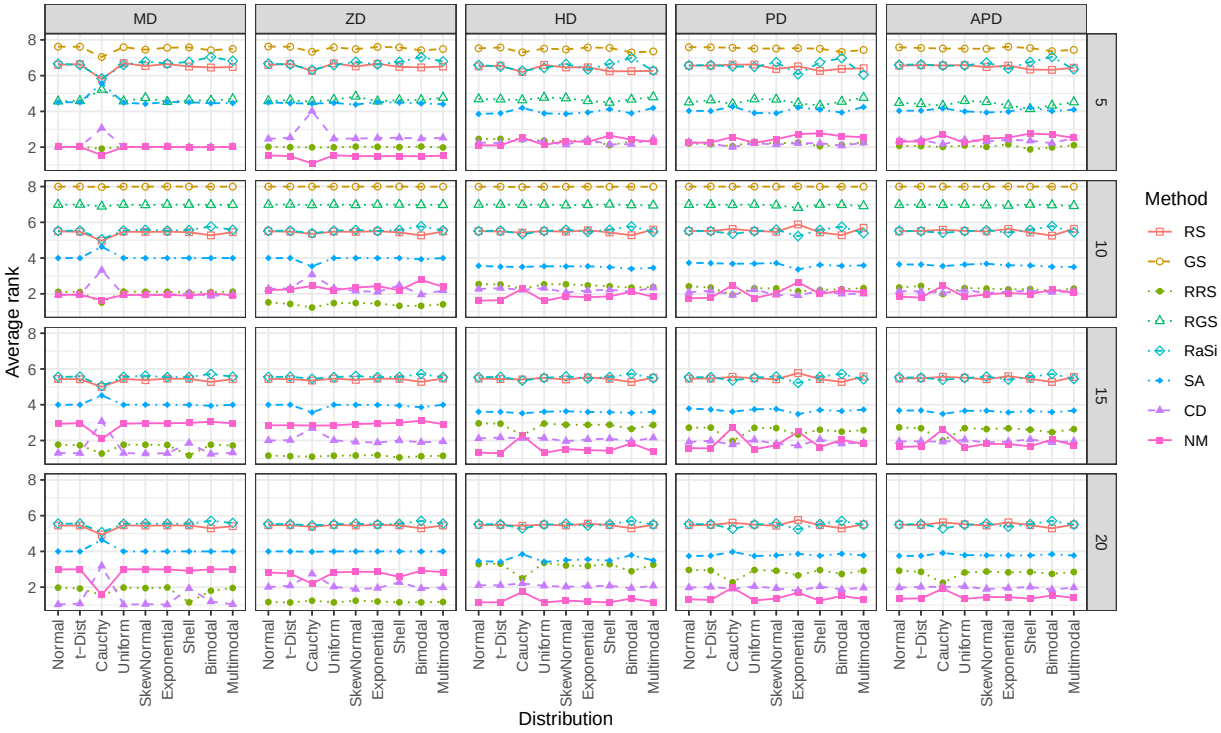


Figure 1.10: AveRank *statistics for the eight considered approximation methods when using* $N \approx 1000$ *projections.*

Figure 1.10 exhibits AveRank for each of the eight considered algorithms for different depth notions, distributions, and dimensions; see Supplementary Materials of Dyckerhoff et al. (2021) for further statistics and values of N. Several observations can be made:

- There is a group of methods which have poor performance that further degrades with increasing dimension: random search (RS), grid search (GS), refined grid search (RGS), and random simplices (RaSi). Moreover, GS and RGS are not considered in

dimension $d > 10$, because 1000 directions are not sufficient to generate even a very sparse grid in such a high dimension.

- Refined random search (RRS), coordinate descent (CD), and Nelder-Mead (NM) show rather good performance.
- NM shows superior behavior in this latter group, since it possesses almost always lower AveRank compared to the two other methods for the halfspace, projection, and asymmetric projection depths. Thus it can be seen as a general winner. However, it is closely followed by CD.

Comparison of AveRank with PercBest, MAE and MRE statistics reveals similarity of the results with very close overall ranking of the methods. Table 1.10 illustrates concordance of the four statistics for the halfspace depth (in most of the cases). Since only RRS, CD and NM appear as best-performing methods w.r.t. the four considered performance statistics, only these three methods are included in the table. Tables which show the best performing methods for the other four dephts are contained in Section 2.3 of Supplementary Materials of Dyckerhoff et al. (2021).

To get more insights into the dynamic of the optimization process, regard the flow of the minimal reached depth with the number of random directions. A typical behavior of the optimization, on the example of the normal distribution in dimension 20, is indicated in Figure 1.11. Similar figures for all nine considered distributions are gathered in Section 2.4 of Supplementary Materials of Dyckerhoff et al. (2021).

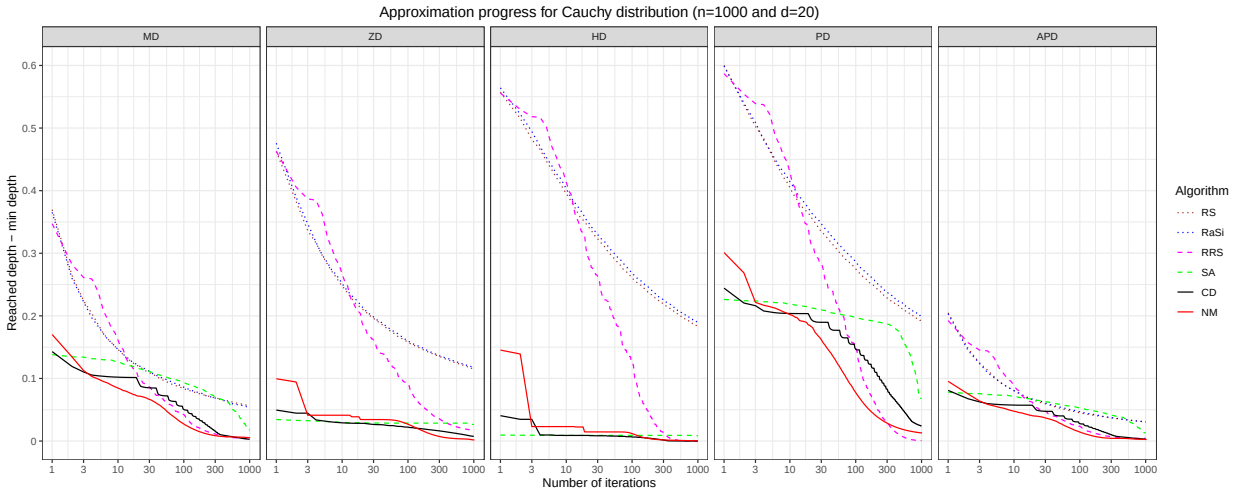


Figure 1.11: *Average (over* 1000 *runs) difference between the reached depth and the minimally achieved depth (by all methods for the current triplet distribution—depth—dimension) during the optimization process (normal distribution, dimension* $d = 20$*). (The lines of RS and RaSi almost coincide in the graphs.)*

The most important observation is a high performance of the optimization techniques (SA, RRS, CD, and NM) compared to the random methods (RS, RaSi). The two latter ones seem to (approximately) follow the bounds mentioned below in Section 1.5 (see Nagy et al.

Table 1.10: *Best performing methods in sense of* AveRank *(top-left square),* PercBest *(top-right square),* MAE *(bottom-left square), and* MRE *(bottom-right square) for the half-space depth (HD).*

HD		Number of projections								
		100			1000			10000		
d	Distribution	RRS	CD	NM	RRS	CD	NM	RRS	CD	NM
5	Normal			■			■		◨	◪
	t-Dist			■			■			■
	Cauchy	◪	◪		◨	◧	◪	◨	◪	◪
	Uniform			■	◨		◪			■
	SkewNormal		■			■			■	
	Exponential		■		◨		◪			■
	Shell	◨	◧		■				■	
	Bimodal	■			◨	◪			■	
	Multimodal	◨	◪		◨		◪	◨		◪
10	Normal			■			■			■
	t-Dist			■			■			■
	Cauchy			■	◨	◧	◪		■	◪
	Uniform			■			■			■
	SkewNormal			■			■		◪	◪
	Exponential			■			■			■
	Shell			■			■		■	
	Bimodal			■		◪	◨		■	
	Multimodal			■			■			■
15	Normal			■			■			■
	t-Dist			■			■			■
	Cauchy			■		◧	◨		◪	◪
	Uniform			■			■			■
	SkewNormal			■			■			■
	Exponential			■			■			■
	Shell			■			■			■
	Bimodal		■				■			■
	Multimodal			■			■			■
20	Normal		■				■			■
	t-Dist		■				■			■
	Cauchy			■			■			■
	Uniform		■				■			■
	SkewNormal		■				■			■
	Exponential		■				■			■
	Shell		■				■			■
	Bimodal		■				■			■
	Multimodal		■				■			■

(2020) for the complete study) and are outperformed already before reaching 100 random directions. Further inspection shows that the improvement of simulated annealing (SA) is very weak, and minor improvement can be expected for even higher number of directions.

A possible explanation would be that the parameters of SA should be tuned separately for each setting.

Comparison with exact depth

Apart from knowing which of the discussed approximation methods gives the best approximation, it is also of interest to have information on the approximation error. To calculate the approximation error, the exact values of the depths have to be known. From the considered depths only the Mahalanobis depth and the zonoid depth can be exactly computed in high dimensions in reasonable time. For the halfspace depth, a family of exact algorithms has been developed in Section 1.2 (for the complete study see Dyckerhoff and M., 2016) to compute the depth in arbitrary dimension which possesses best complexity of $O(n^{d-1} \log n)$. For the considered sample size of $n = 1000$ the exact computation of the halfspace depth is (in reasonable time) only possible when $d \leq 5$. Although there is an exact algorithm for the projection depth (Liu and Zuo, 2014b), the considered sample size of $n = 1000$ in dimension $d = 5$ is already too large to have the value of the depth computed in reasonable time. For the asymmetric projection depth no exact algorithm exists. Therefore, a closer look at two situations is taken. First, let us examine the approximation of the halfspace depth since it probably is the most prominent depth. Because of the high computational cost approximation errors for the halfspace depth were computed only in dimension $d = 5$. Second, to get some intuition on approximation errors in high dimensions, the mean absolute error (MAE) and the mean relative error (MRE) are analyzed, which were exactly computed for the zonoid (and also the Mahalanobis) depth in all dimension $d = 5, 10, 15, 20$. The zonoid depth was chosen since apart from the trivial case of the Mahalanobis depth, it is the only depth considered in this study for which exact computation is possible when $d = 20$. The time for computing the zonoid depth of a single point w.r.t. a sample of size $n = 1000$ in dimension $d = 20$ is still under one second. For both setups (halfspace depth, $d = 5$, and zonoid depth, $d \leq 20$) the same simulated data sets were taken that were already used in the simulation comparison. For the halfspace depth, only the first fifteen simulated data sets were used to compute the approximation errors. The approximated depth values as well as the exact depth values for the fifteen data sets simulated from the normal distribution are shown for the halfspace depth in Table 1.11. The last two lines of the table show the mean absolute error (MAE) and the mean relative error (MRE) of the considered approximation methods. The respective tables for all nine distributions and $N \approx 100, 1000, 10000$ projections are given in Section 3 of Supplementary Materials of Dyckerhoff et al. (2021).

Because of its good computability, for the zonoid depth all 1000 simulated data sets were used for each combination of dimension and distribution. In Table 1.12 the mean relative errors (MRE) are shown for the case where 1000 projections were used. The respective tables for both the mean relative errors (MRE) and the mean absolute errors (MAE) for all nine distributions and $N \approx 100, 1000, 10000$ projections are shown in Section 2.2.2 of Supplementary Materials of Dyckerhoff et al. (2021).

Table 1.11: *Exact and approximate values of the halfspace depth for* 15 *points together with the corresponding MAE and MRE: normal distribution,* $n = 1000$, $d = 5$, $N \approx 1000$ *projections.*

	RS	GS	RRS	RGS	RaSi	SA	CD	NM	Exact
1	0.279	0.286	0.267	0.273	0.279	0.269	0.268	0.269	0.265
2	0.143	0.160	0.132	0.139	0.148	0.136	0.132	0.130	0.128
3	0.244	0.259	0.238	0.245	0.247	0.241	0.238	0.238	0.236
4	0.329	0.340	0.310	0.336	0.324	0.316	0.311	0.311	0.309
5	0.220	0.235	0.200	0.208	0.225	0.204	0.199	0.200	0.197
6	0.236	0.259	0.214	0.229	0.227	0.219	0.216	0.217	0.213
7	0.238	0.242	0.230	0.235	0.244	0.233	0.227	0.229	0.226
8	0.228	0.229	0.218	0.225	0.229	0.223	0.219	0.218	0.215
9	0.171	0.169	0.152	0.153	0.164	0.154	0.151	0.152	0.149
10	0.241	0.248	0.224	0.230	0.228	0.225	0.223	0.223	0.221
11	0.187	0.205	0.170	0.182	0.185	0.169	0.169	0.169	0.166
12	0.284	0.280	0.269	0.273	0.288	0.274	0.272	0.271	0.268
13	0.221	0.228	0.201	0.218	0.212	0.206	0.206	0.202	0.200
14	0.171	0.182	0.161	0.161	0.176	0.160	0.158	0.157	0.154
15	0.168	0.188	0.157	0.158	0.168	0.161	0.161	0.157	0.157
MAE	0.017	0.027	0.003	0.011	0.016	0.006	0.003	0.003	0.000
MRE	0.105	0.169	0.017	0.063	0.099	0.035	0.019	0.016	0.000

For the halfspace depth and normally distributed data in dimension $d = 5$, when $N \approx 1000$ projections are used, the best methods are NM and RRS, followed by CD, whereas the worst methods are RS, GS and RaSi. Even though there is one case (data set 15) where RRS and NM found the exact halfspace depth, MAE suggests that on average the best halfspace found by RRS or NM contains three points more than the optimal halfspace. Furthermore, relative depth approximation error remains (again on an average) below 2% of the exact depth value.

For the zonoid depth, when the approximation was done using $N \approx 1000$ projections, RRS gets the first place (when $d > 5$) followed by CD, NM and SA. An important point to note is that the very basic methods like RS and RaSi are unusable when dimension is high with relative error rates beyond 50% ($d = 15$) or even beyond 100% ($d = 20$). It is noteworthy that the same holds for RRS when the number of projections is low (see Supplementary Materials of Dyckerhoff et al., 2021), which suggests that RRS needs a substantial number of projections to work well. However, the more elaborate methods like CD, NM and SA perform well regardless of the number of projections. With these methods, relative errors can be kept reasonably low, even below 1.5% (except the Cauchy distribution) of the exact value.

1.4.4 Comments on quality of approximation

It is important to provide additional insights in the quality of approximation of the discussed above methods. Mahalonobis and zonoid depths have been included for comparison

Table 1.12: *Mean relative error (MRE) for the approximation of the zonoid depth,* $n = 1000$ *data points,* $N \approx 1000$ *projections.*

		Approximation algorithm							
d	Distribution	RS	GS	RRS	RGS	RaSi	SA	CD	NM
5	Normal	0.018782	0.040681	0.000002	0.002805	0.019107	0.000535	0.000004	0.000001
	t-Dist	0.021093	0.047018	0.000002	0.003281	0.021418	0.000597	0.000008	0.000001
	Cauchy	0.043137	0.080894	0.000465	0.016957	0.045022	0.010639	0.017534	0.000056
	Uniform	0.018799	0.041453	0.000002	0.003367	0.018021	0.000537	0.000004	0.000001
	SkewNormal	0.021533	0.046762	0.000002	0.005977	0.029959	0.000643	0.000005	0.000001
	Exponential	0.021569	0.047682	0.000002	0.003068	0.020915	0.000653	0.000007	0.000001
	Shell	0.022559	0.052451	0.000002	0.003401	0.032908	0.000665	0.000005	0.000001
	Bimodal	0.023904	0.052215	0.000002	0.003452	0.037974	0.000727	0.000005	0.000001
	Multimodal	0.022954	0.048468	0.000002	0.006329	0.033007	0.000691	0.000006	0.000001
10	Normal	0.217197	1.118694	0.000025	0.942766	0.219397	0.007122	0.000057	0.000094
	t-Dist	0.222892	1.289776	0.000036	1.094230	0.228138	0.008009	0.000094	0.000168
	Cauchy	0.303627	2.212219	0.006813	1.661869	0.307660	0.064152	0.052861	0.034096
	Uniform	0.215653	1.090465	0.000023	0.937243	0.219798	0.006999	0.000054	0.000098
	SkewNormal	0.218591	1.081675	0.000028	0.923719	0.237190	0.008551	0.000062	0.000228
	Exponential	0.229525	1.201967	0.000037	1.001438	0.232001	0.009666	0.000090	0.000489
	Shell	0.222987	1.076789	0.000028	0.916450	0.244420	0.007856	0.000122	0.000214
	Bimodal	0.234134	1.111908	0.000030	0.940334	0.301895	0.008984	0.000063	0.002062
	Multimodal	0.219745	1.078995	0.000027	0.906989	0.225948	0.009091	0.000072	0.000280
15	Normal	0.585529	—	0.000182	—	0.609475	0.022943	0.000392	0.001717
	t-Dist	0.575809	—	0.000277	—	0.596498	0.027329	0.000737	0.002582
	Cauchy	0.676690	—	0.020609	—	0.692426	0.166416	0.084469	0.097217
	Uniform	0.579684	—	0.000160	—	0.588454	0.022043	0.000365	0.001557
	SkewNormal	0.581892	—	0.000200	—	0.619134	0.027594	0.000437	0.003568
	Exponential	0.607952	—	0.000343	—	0.622455	0.033453	0.000909	0.006960
	Shell	0.581782	—	0.000186	—	0.600450	0.025505	0.000951	0.008322
	Bimodal	0.600136	—	0.000206	—	0.719552	0.030843	0.000546	0.014352
	Multimodal	0.576272	—	0.000180	—	0.594009	0.026946	0.000436	0.002727
20	Normal	1.143799	—	0.001079	—	1.177983	0.052060	0.002086	0.006032
	t-Dist	1.024786	—	0.001421	—	1.052585	0.060773	0.003030	0.007248
	Cauchy	1.110807	—	0.049412	—	1.125693	0.369281	0.158725	0.099501
	Uniform	1.131364	—	0.001026	—	1.148654	0.050696	0.002042	0.005657
	SkewNormal	1.141836	—	0.001293	—	1.178756	0.070894	0.002138	0.008604
	Exponential	1.151532	—	0.002333	—	1.180201	0.126961	0.004631	0.014133
	Shell	1.195071	—	0.001521	—	1.233843	0.076760	0.004516	0.006101
	Bimodal	1.189500	—	0.001256	—	1.376561	0.119274	0.003015	0.014454
	Multimodal	1.117096	—	0.001127	—	1.157293	0.066462	0.002162	0.006479

purposes and for wider representation of the class of depths satisfying the projection property. Based on experimental data, for these two depths very good approximation quality is expected for optimization techniques (RRS, CD, NM, SA). This can be explained by "well-behaving" function to be minimized $\phi_{\boldsymbol{z},\boldsymbol{X}}(\boldsymbol{u})$ while Mahalanobis and zonoid depth are not robust. For halfspace, projection, and asymmetric projection depths, $\phi_{\boldsymbol{z},\boldsymbol{X}}(\boldsymbol{u})$ is presumably a non-convex functional in most of the cases and can possess multiple local minima. While worse optimization results are expected for these three depths, exact value is achievable theoretically for the halfspace depth due to step-wise shape of $\phi_{\boldsymbol{z},\boldsymbol{X}}(\boldsymbol{u})$; *i.e.* even purely random

algorithms (like RS or RaSi) can "guess" exact value with positive probability thanks to plateaus in $\phi_{\boldsymbol{z},\boldsymbol{X}}(\boldsymbol{u})$.

Apart heuristic procedures, to ensure the significance of the results of developed algorithms and gain trust of practitioners applying them, (statistical) guarantees on their numerical convergence are desired. Following Section 1.5 delivers very first answer on this question by providing uniform convergence rates for approximation of the halfspace depth $D(\boldsymbol{x}|X)$ w.r.t. a random vector for the RS method, which is mainly possible due to the independence of directions $\boldsymbol{u}$s. Obtaining the same result for a data sample, *i.e.* $D(\boldsymbol{x}|\boldsymbol{X})$, does not appear to be possible uniformly, see Section 1.5.3 for an example. The situation is even more involved when an optimization technique is in place and each subsequent direction depends on the evaluation of the previous one(s). Finally, presence of multiple local mimina further complicates the task of derivation of guarantees on approximation.

Generally speaking, depth statistics with "good" properties are computationally expensive. There seems to to be a conflict between maintaining such attractive statistical properties as affine invariance (respectively affine equivariance for depth regions) and robustness on one side and computational tractability (*i.e.* non-exponential complexity in (n, d)) on the other side; see Johnson and Preparata (1978) and Bernholt (2006) for theoretical results on this topic. Some insights on choice of practically suitable depth notion—also from computational point of view—can be found in Mosler and M. (2022). In order to smooth this search for compromise, two ways can be considered, both (most probably) requiring substantial theoretical developments and additional statistical assumptions. One such possibility is to connect approximation error with computational time and/or complexity. Another possibility is to clarify a trade-off between statistical properties the depth function satisfies and its computational complexity. More on these directions can be found in the description of ideas for future research in Section 5.1.

1.5 First theoretical guarantees on depth approximation

Since all the 8 algorithms introduced in the previous Section 1.4 are heuristics, the questions of their reliability immediately arises. In this section, let us try to answer it by uniform bounds on the approximation error. In view of the complexity of the algorithms, with most efficient being optimization techniques run on a non-convex functional (over the unit sphere), the answer demands further effort and is planned for future research; see Section 5.1. Here, on an example of one depth (namely halfspace depth) and one algorithm (namely random search, while being the simplest one because purely random), uniform approximation bounds are derived. As a further simplification, these bounds are derived for computation of depth w.r.t. the random vector X (and not data set as it is obviously desirable); impossibility of such bounds (under most general assumptions) is discussed in Example 1 below. Denote this quantity $D_{\mathrm{H},N}(\cdot|X)$.

First, in Section 1.5.1, the task is formalized and the notation is set up. Further, in Section 1.5.2, the uniform approximation bound on the halfspace depth is derived, in two steps. In the first step, the main result is derived, under the assumption of uniformly Lipschitz continuous functions $\varphi_{\boldsymbol{x}}$. In the second step this result is extended to the general class of equicontinuous functions (1.14). Finally, in Section 1.5.3, the negative answer on uniform approximation for a data set—under most general assumptions—is given.

1.5.1 Approximation of the halfspace depth

Let $N = 1, 2, \ldots$, and consider $U_1, \ldots, U_N$ a random sample from the uniform distribution on $\mathbb{S}^{d-1}$. One is interested in the quality of the approximation of the halfspace depth (1.3) by its randomized counterpart (which is closely related to Definition 1.8):

$$D_{\mathrm{H},N}(\boldsymbol{x}|X) = \min_{i=1,\ldots,N} \mathbb{P}\left[\langle U_i, X\rangle \leq \langle U_i, \boldsymbol{x}\rangle\right]. \tag{1.12}$$

In what follows, $D_{\mathrm{H},N}(\boldsymbol{x}|X)$ will be shortened to $D_{\mathrm{H},N}(\boldsymbol{x})$. It is evident that

$$D_{\mathrm{H},N}(\boldsymbol{x}) \geq D_{H,N+1}(\boldsymbol{x}) \geq D_{\mathrm{H}}(\boldsymbol{x}) \quad \text{for all } \boldsymbol{x} \in \mathbb{R}^d \text{ and } N = 1, 2 \ldots,$$

and that for $d = 1$ the random depth $D_{\mathrm{H},N}$ with high probability reduces to the true depth D_{H}

$$\mathbb{P}\left[\sup_{x\in\mathbb{R}} |D_{\mathrm{H},N}(\boldsymbol{x}) - D_{\mathrm{H}}(\boldsymbol{x})| = 0\right] = 1 - 2\left(\frac{1}{2}\right)^N \quad \text{for all } N = 1, 2, \ldots$$

In the interesting case $d > 1$, for any $x \in \mathbb{R}^d$ the random $D_{\mathrm{H},N}(\boldsymbol{x})$ approximates $D_{\mathrm{H}}(\boldsymbol{x})$. In (Dyckerhoff, 2004, Proposition 11) it was shown that for any $x \in \mathbb{R}^d$ and $P \in \mathcal{P}\left(\mathbb{R}^d\right)$ the convergence $D_{\mathrm{H},N}(\boldsymbol{x}) \xrightarrow[N\to\infty]{\text{a.s.}} D_{\mathrm{H}}(\boldsymbol{x})$ holds true. The goal of the current section is to establish a uniform extension of that convergence result, and to derive the corresponding rates of convergence; for an array of applications of these results to the computation of the depth see Section 2.3 of M. et al. (2015).

Define the *halfspace function* of $X \sim P$, given for $\boldsymbol{x} \in \mathbb{R}^d$ by

$$\varphi_{\boldsymbol{x}} \colon \mathbb{S}^{d-1} \to [0, 1] \colon \boldsymbol{u} \mapsto \mathbb{P}\left[\langle \boldsymbol{u}, X\rangle \leq \langle \boldsymbol{u}, \boldsymbol{x}\rangle\right]. \tag{1.13}$$

Both $D_{\mathrm{H}}(\boldsymbol{x})$ and its approximation $D_{\mathrm{H},N}(\boldsymbol{x})$ can be expressed in terms of $\varphi_{\boldsymbol{x}}$ as

$$D_{\mathrm{H}}(\boldsymbol{x}) = \inf_{\boldsymbol{u}\in\mathbb{S}^{d-1}} \varphi_{\boldsymbol{x}}(\boldsymbol{u}), \quad \text{and} \quad D_{\mathrm{H},N}(\boldsymbol{x}) = \min_{i=1,\ldots,N} \varphi_{\boldsymbol{x}}(U_i).$$

If the minimum value of $\varphi_{\boldsymbol{x}}$ is attained in a single direction in $\mathbb{S}^{d-1}$, denote this direction by $\widetilde{\boldsymbol{u}}(\boldsymbol{x}) \in \mathbb{S}^{d-1}$, that is $D_{\mathrm{H}}(\boldsymbol{x}) = \varphi_{\boldsymbol{x}}\left(\widetilde{\boldsymbol{u}}(\boldsymbol{x})\right)$.

When deriving the rates of convergence of the depth approximations that are uniform on a set $C \subset \mathbb{R}^d$, it is necessary to assume a certain form of the equicontinuity of the halfspace functions

$$\{\varphi_{\boldsymbol{x}} \colon \boldsymbol{x} \in C\}. \tag{1.14}$$

1.5.2 Uniform convergence rates for approximation of the halfspace depth

One is interested in the rate of convergence of the random sequence

$$\Delta_N(C) = \sup_{\boldsymbol{x}\in C} |D_{\mathrm{H},N}(\boldsymbol{x}) - D_{\mathrm{H}}(\boldsymbol{x})|$$

as $N \to \infty$. Let us start by two technical results that will be of great importance in the sequel. For $\Gamma(\cdot)$ the gamma function, denote by

$$a_d(\varphi) = \frac{\Gamma(d/2)}{\Gamma((d-1)/2)\sqrt{\pi}} \int_0^{\varphi} (\sin(\theta))^{d-2} \,\mathrm{d}\,\theta, \quad \text{for } \varphi \in [0,\pi]$$

the $(d-1)$-dimensional Hausdorff measure of a cap of a polar angle[1.2] φ of a sphere in $\mathbb{R}^d$ with unit surface (hyper-)area. In other words, $a_d(\varphi)$ is the ratio of the surface area of the spherical cap of polar angle φ in $\mathbb{S}^{d-1}$, and the surface area of the whole $(d-1)$-sphere $\mathbb{S}^{d-1}$. The function a_d takes particularly simple forms for $d = 2$ and $d = 3$

$$a_2(\varphi) = \frac{\varphi}{\pi} \quad \text{and} \quad a_3(\varphi) = \left(\sin\left(\frac{\varphi}{2}\right)\right)^2.$$

For higher dimensions d, it is a trigonometric polynomial. Note that all functions a_d are strictly increasing and continuous on their domain. Denote the inverse of a_d by $a_d^{-1}\colon [0,1] \to [0,\pi]$.

Lipschitz continuous halfspace functions

Assume that the set of functions (1.14) is uniformly Lipschitz continuous with constant $L \geq 0$, *i.e.* that

$$\sup_{\boldsymbol{x}\in C} |\varphi_{\boldsymbol{x}}(\boldsymbol{u}) - \varphi_{\boldsymbol{x}}(\boldsymbol{v})| \leq L \|\boldsymbol{u} - \boldsymbol{v}\|_g \quad \text{for all } \boldsymbol{u}, \boldsymbol{v} \in \mathbb{S}^{d-1}, \tag{1.15}$$

where $\|\boldsymbol{u} - \boldsymbol{v}\|_g = \arccos(\langle \boldsymbol{u}, \boldsymbol{v}\rangle)$ is the great-circle distance, *i.e.* the geodesic distance between $\boldsymbol{u}$ and $\boldsymbol{v}$. A condition similar to (1.15) was used in Burr and Fabrizio (2017) when deriving the uniform rates of convergence for the ordinary empirical halfspace depth process. Examples of distributions that satisfy property (1.15) are listed in Section 4 of Nagy et al. (2020).

Theorem 1.3 *(Nagy et al., 2020) Let $P \in \mathcal{P}(\mathbb{R}^d)$ be such that (1.15) holds true for a set $C \subset \mathbb{R}^d$. Then*

$$\limsup_{N\to\infty} \frac{N\, a_d(\Delta_n(C)/L) - \log N}{\log\log N} \leq d \quad \text{a.s.}$$

[1.2] The angle between the rays from the center of the hypersphere to the apex of the cap, and the edge of the $(d-1)$-dimensional disk that forms the base of the cap.

General uniformly continuous projections

Now, assume that the class of functions (1.14) is uniformly equicontinuous, but not necessarily uniformly Lipschitz. Consider the minimal modulus of continuity of this class of functions

$$\delta\colon [0,\pi] \to [0,1]\colon t \mapsto \sup_{\boldsymbol{x}\in C} \sup_{\boldsymbol{u},\boldsymbol{v}\in\mathbb{S}^{d-1},\, \|\boldsymbol{u}-\boldsymbol{v}\|_g \le t} \left|\varphi_{\boldsymbol{x}}(\boldsymbol{u}) - \varphi_{\boldsymbol{x}}(\boldsymbol{v})\right|. \tag{1.16}$$

Under the condition of uniform equicontinuity of (1.14), the function $\delta(t)$ is continuous, non-negative and non-decreasing, with $\delta(0) = 0$, see Chapter 2, § 6 of DeVore and Lorentz (1993).

Theorem 1.4 *(Nagy et al., 2020) Let $P \in \mathcal{P}\left(\mathbb{R}^d\right)$ and $C \subset \mathbb{R}^d$ be such that the function δ from (1.16) is continuous at 0 and strictly increasing with an inverse function δ^{-1}. Then*

$$\limsup_{N\to\infty} \frac{N\, a_d\left(\delta^{-1}\left(\Delta_N(C)\right)\right) - \log N}{\log\log N} \le d \quad \text{a.s.}$$

If (1.14) is uniformly Hölder continuous, *i.e.* for some $K > 0$ and $\alpha \in (0,1]$ one may choose $\delta(t) \le Kt^{\alpha}$, the last formula gives

$$\limsup_{N\to\infty} \frac{N\, a_d\left(\left(\Delta_N(C)/K\right)^{1/\alpha}\right) - \log N}{\log\log N} \le d \quad \text{a.s.}$$

1.5.3 Non-uniform approximation

If the measure P is not regular enough, uniform approximations of the halfspace depth may not be attainable. Let us illustrate this claim on an example, where an atomic distribution is constructed for which approximated depth fails to converge uniformly to its population counterpart.

Example 1 *If P does not admit a density, the halfspace functions $\varphi_{\boldsymbol{x}}$ may contain discontinuities. In particular, this occurs when the depth is computed with respect to the empirical measure of a random sample of observations, i.e. for the sample halfspace depth. In that case, it can be inferred that the convergence of the approximations is not uniform, even on bounded sets C.*

Consider the example of P atomic, supported in a set $\{\boldsymbol{x}_i\}_{i=1}^m$ in $\mathbb{R}^d$ such that $P(\{\boldsymbol{x}_i\}) = p_i$ with $\sum_{i=1}^m p_i = 1$. Let $0 < p_1 \le p_2 \le \cdots \le p_m$. Without loss of generality, assume that the convex hull $K \subset \mathbb{R}^d$ of these points is d-dimensional, i.e. that K is not fully contained in an affine subspace of $\mathbb{R}^d$ of dimension lower than d. Let $\boldsymbol{x}$ be a point on a facet of K, and let $\boldsymbol{x}_\varepsilon = \boldsymbol{x} + \varepsilon\,\boldsymbol{v}$ for $\varepsilon > 0$, where $\boldsymbol{v} \in \mathbb{S}^{d-1}$ is the outward normal of the facet of K on which $\boldsymbol{x}$ lies. Since $\boldsymbol{x}_\varepsilon \notin K$ by definition, $D_{\mathrm{H},N}(\boldsymbol{x}_\varepsilon) = 0$ for any $\varepsilon > 0$. For its approximation clearly $D_{\mathrm{H},N}(\boldsymbol{x}_\varepsilon) \ge p_1$ if and only if for some halfspace

$$H(\boldsymbol{x}_\varepsilon, U_i) = \left\{\boldsymbol{y} \in \mathbb{R}^d\colon\ \langle U_i, \boldsymbol{y}\rangle \le \langle U_i, \boldsymbol{x}_\varepsilon\rangle\right\}$$

whose boundary passes through $\boldsymbol{x}_\varepsilon$ *with outer normal* $U_i \in \mathbb{S}^{d-1}$ *intersects* K*. For* ε *small, this will occur with high probability if the directions* U_i *are sampled uniformly on* $\mathbb{S}^{d-1}$ *— as* $\varepsilon \to 0$ *from the right, the condition* $H(\boldsymbol{x}_\varepsilon, U_i) \cap K = \varnothing$ *effectively reduces to* $U_i = -\boldsymbol{v}$*, an event of null probability. Thus, the convergence of the approximations cannot be uniform on the line segment* $\boldsymbol{x} + \varepsilon\,\boldsymbol{v}$ *with* $\varepsilon \in (0, 1)$*, and*

$$\limsup_{N\to\infty} \sup_{\varepsilon\in(0,1)} \left(D_{\mathrm{H},N}(\boldsymbol{x}_\varepsilon) - D_{\mathrm{H}}(\boldsymbol{x}_\varepsilon)\right) \geq p_1 > 0 \quad \text{a.s.}$$

Further, for uniform convergence rates for the family of projection depths, the reader is invited to consult Section 5 of Nagy et al. (2020).

Chapter 2

Application: nonparametric imputation using data depth

Missing data is a ubiquitous problem in statistics. Non-responses to surveys, machines that break and stop reporting, and data that have not been recorded, impede analysis and threaten the validity of inference. A common strategy (Little and Rubin, 2014) for dealing with missing values is single imputation, replacing missing entries with plausible values to obtain a completed data set, which can then be analyzed.

There are two main families of parametric imputation methods: "joint" and "conditional" modeling, see *e.g.*, Josse and Reiter (2018) for a literature overview. Joint modeling specifies a joint distribution for the data, the most popular being the normal multivariate distribution. The parameters of the distribution, here the mean and the covariance matrix, are then estimated from the incomplete data using an algorithm such as expectation maximization (EM) (Dempster et al., 1977). The missing entries are then imputed with the conditional mean, *i.e.*, the conditional expectation of the missing values, given observed values and the estimated parameters. An alternative is to impute missing values using a principal component analysis (PCA) model which assumes data are generated as a low rank structure corrupted by Gaussian noise. This method is closely connected to the literature on matrix completion Josse and Husson (2012), Hastie et al. (2015), and has shown good imputation capacity due to the plausibility of the low rank assumption (Udell and Townsend, 2018). The conditional modeling approach (Van Buuren, 2012) consists in specifying one model for each variable to be imputed, and considers the others variables as explanatory. This procedure is iterated until predictions stabilize. Nonparametric imputation methods have also been developed such as imputation by k-nearest neighbors (kNN) (see Troyanskaya et al., 2001, and references therein) or random forest (Stekhoven and Bühlmann, 2012).

Most imputation methods are defined under the missing (completely) at random (M(C)AR) assumption, which means that the probability of having missing values does not depend on missing data (nor on observed data). Gaussian and PCA imputations are sensitive to outliers and deviations from distributional assumptions, whereas nonparametric methods such as kNN and random forest cannot extrapolate.

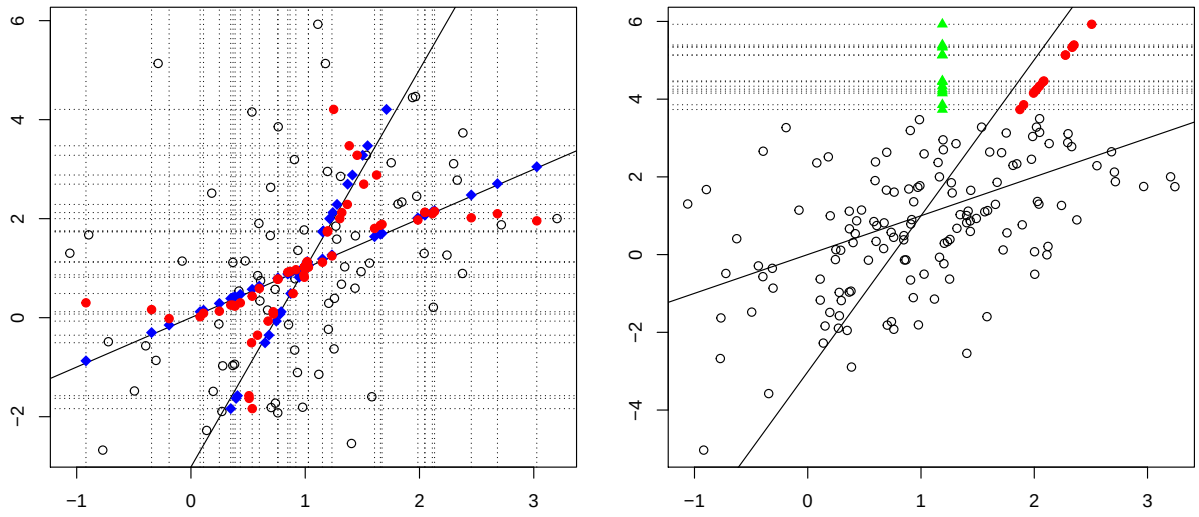


Figure 2.1: *Bivariate normal distribution with 30% MCAR (left) and with MAR in the second coordinate for values* > 3.5 *(right); imputation using maximum zonoid depth (filled circles), conditional mean imputation using EM estimates (rhombi), and random forest imputation (triangles).*

2.1 Motivation

In this chapter, a family of nonparametric imputation methods is proposed, based on the notion of a statistical depth function (Tukey, 1975). Data depth is a data-driven multivariate measure of centrality that describes data with respect to location, scale, and shape based on a multivariate ordering. It has been applied in multivariate data analysis (Liu et al., 1999), classification (Jörnsten, 2004, Lange et al., 2014b), multivariate risk measurement (Cascos and Molchanov, 2007), and robust linear programming (Bazovkin and Mosler, 2015), but has never been applied in the context of missing data. Depth based imputation provides excellent predictive properties and has the advantages of both global and local imputation methods. It imputes close to the data geometry, while still accounting for global features. In addition, it allows robust imputation in both outliers and heavy-tailed distributions.

Figures 2.1 and 2.2 motivate the proposed depth-based imputation by contrasting it to classical methods. First, 150 points are drawn from a bivariate normal distribution with mean $\boldsymbol{\mu}_1 = (1,1)^\top$ and covariance $\boldsymbol{S}_1 = \left((1,1)^\top, (1,4)^\top\right)$ and 30% of the entries are removed completely at random in both variables; points with one missing entry are indicated by dotted lines while solid lines provide (oracle) imputation using distribution parameters. The imputation assuming a joint Gaussian distribution using EM estimates is shown by rhombi (Figure 2.1, left). Zonoid depth-based imputation, represented by filled circles, shows that the sample is not necessarily normal, and that this uncertainty increases as one moves to the fringes of the data cloud, where imputed points deviate from the conditional mean towards the unconditional one. Second, the missing values are generated as follows: the first coordinate is removed when the second coordinate > 3.5 (Figure 2.1, right). Here,

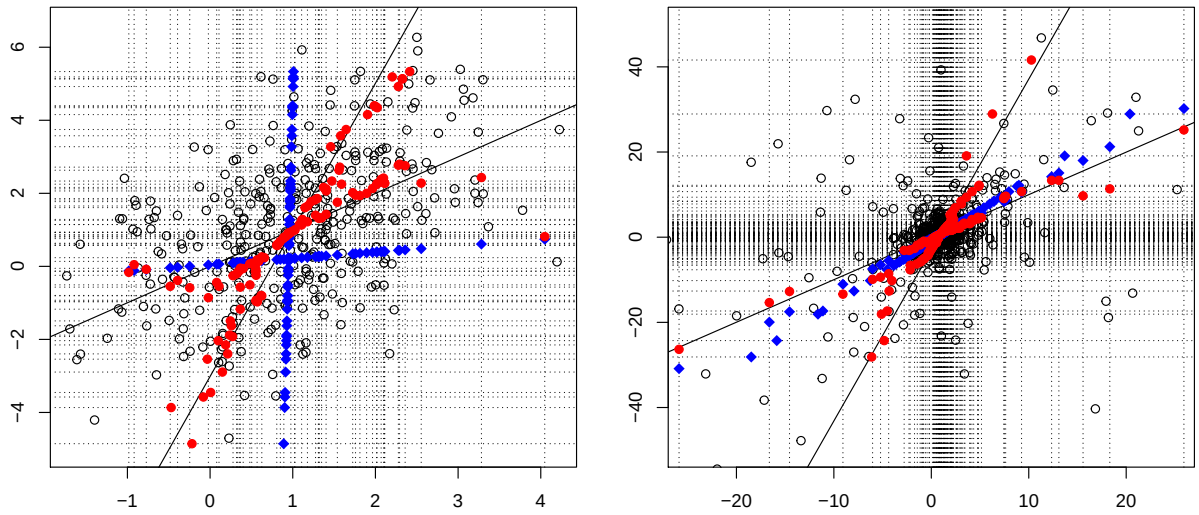


Figure 2.2: *Left: Mixture of normal (425 points, 15% MCAR) and Cauchy (75 points) samples. Right: 1000 Cauchy distributed points with 15% MCAR. Imputation with halfspace depth (filled circles) and conditional mean imputation using EM estimates (rhombi).*

the depth-based imputation allows extrapolation when predicting missing values, while the random forest imputation (triangles) gives, as expected, rather poor results.

In Figure 2.2 (left), 500 points are drawn, 425 from the same normal distribution as above, with 15% of MCAR values and 75 outliers from the Cauchy distribution with the same center and shape matrix and without missing values. In Figure 2.2 (right), 1000 points are depicted drawn from Cauchy distribution with 15% MCAR. As expected, imputation with conditional mean based on EM estimates (rhombi) is rather random. Depth-based imputation with halfspace depth (filled circles) has robust imputed values that are close to the (distribution's) regression lines reflecting data geometry.

The rest of the chapter is structured as follows. In Section 2.2, the general depth-based imputation method is proposed: stated formally in Section 2.2.2 with an algorithm and theoretical guarantees for the elliptical family, it is preceded by iterative-regression motivation in Section 2.2.1. Further, in Section 2.3, by-depth analysis is performed, for Mahalanobis (Section 2.3.1), zonoid (Section 2.3.2), and halfspace (Section 2.3.3) depths. After this, imputation of missing data for distributions with non-convex support is discussed in Section 2.3.4, as well as the question of "outsiders", *i.e.*, points to be imputed lying beyond the convex hull of the complete data (these can cause problems if data depth in these regions systematically equals zero), which is tackled in Section 2.3.5. Finally, Section 2.4 gathers information about simulation (and real-data) studies. Additionally, Section 5 of M. et al. (2020) addresses the question of multiple imputation, for the family of elliptical distributions.

2.2 Proposed imputation scheme

2.2.1 Imputation by iterative regression

Let X be a random vector in $\mathbb{R}^d$ and denote $\boldsymbol{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$ a sample. For a point $\boldsymbol{x}_i \in \boldsymbol{X}$, denote $miss(i)$ and $obs(i)$ the sets of its coordinates containing missing and observed values, $|miss(i)|$ and $|obs(i)|$ their corresponding cardinalities.

Let the rows $\boldsymbol{x}_i$ be i.i.d. draws from $\mathcal{N}(\boldsymbol{\mu_X}, \boldsymbol{\Sigma_X})$. One of the simplest conditional methods for imputing missing values consists in the following iterative regression imputation: (1) initialize missing values arbitrary, using unconditional mean imputation; (2) impute missing values in one variable by the values predicted by the regression model of this variable with the remaining variables taken as explanatory ones; (3) iterate through variables containing missing values until convergence. Here, at each step, each point $\boldsymbol{x}_i$ with missing values at a coordinate j is imputed with the univariate conditional mean $\mathbb{E}[X|X_{\{1,\ldots,d\}\setminus\{j\}} = \boldsymbol{x}_{i,\{1,\ldots,d\}\setminus\{j\}}, \boldsymbol{\mu}_X = \boldsymbol{\mu_X}, \boldsymbol{\Sigma}_X = \boldsymbol{\Sigma_X}]$ with the moment estimates $\boldsymbol{\mu_X} = \frac{1}{n}\sum_{i=1}^n \boldsymbol{x}_i$ and $\boldsymbol{\Sigma_X} = \frac{1}{n-1}\sum_{i=1}^n (\boldsymbol{x}_i - \boldsymbol{\mu_X})(\boldsymbol{x}_i - \boldsymbol{\mu_X})^\top$. After convergence, each point $\boldsymbol{x}_i$ with missing values in $miss(i)$ is imputed with the multivariate conditional mean

$$\begin{aligned} &\mathbb{E}[X|X_{obs(i)} = \boldsymbol{x}_{i,obs(i)}, \boldsymbol{\mu}_X = \boldsymbol{\mu_X}, \boldsymbol{\Sigma}_X = \boldsymbol{\Sigma_X}] \\ =&\boldsymbol{\mu}_{\boldsymbol{X}\, miss(i)} + \boldsymbol{\Sigma}_{\boldsymbol{X}\, miss(i),obs(i)} \boldsymbol{\Sigma}^{-1}_{\boldsymbol{X}\, obs(i),obs(i)} \big(\boldsymbol{x}_{i,obs(i)} - \boldsymbol{\mu}_{\boldsymbol{X}\, obs(i)}\big). \end{aligned} \tag{2.1}$$

The last expression is the closed-form solution to

$$\min_{\boldsymbol{z}_{miss(i)} \in \mathbb{R}^{|miss(i)|},\, \boldsymbol{z}_{obs(i)} = \boldsymbol{x}_{obs(i)}} d_{\text{Mah}}(\boldsymbol{z}, \boldsymbol{\mu_X} | \boldsymbol{\Sigma_X})$$

with $d^2_{\text{Mah}}(\boldsymbol{z}, \boldsymbol{\mu_X}|\boldsymbol{\Sigma_X}) = (\boldsymbol{z} - \boldsymbol{\mu_X})^\top \boldsymbol{\Sigma}^{-1}_{\boldsymbol{X}} (\boldsymbol{z} - \boldsymbol{\mu_X})$ being the squared Mahalanobis distance from $\boldsymbol{z}$ to $\boldsymbol{\mu_X}$. Minimizing the Mahalanobis distance can be seen as maximizing a centrality measure—the Mahalanobis depth:

$$\max_{\boldsymbol{z}_{miss(i)} \in \mathbb{R}^{|miss(i)|},\, \boldsymbol{z}_{obs(i)} = \boldsymbol{x}_{obs(i)}} D_{\text{Mah}}(\boldsymbol{z}|\boldsymbol{X})$$

where the Manahalobis depth of $\boldsymbol{x} \in \mathbb{R}^d$ w.r.t. X is defined by (1.2) (see Definition 1.1 in Section 1.1.2).

The Manahalobis depth is the simplest instance of a statistical depth function. Now, let us generalize the iterative imputation algorithm to other depths.

2.2.2 Imputation by depth maximization

Let us start by strengthening the requirements on data depth necessary for its application in imputation of missing values. In the rest of this chapter, the depth invariance will be restricted to the class of affine transformations, *i.e.* $\mathcal{T}$ consists of translations and linear mappings (see Section 1.1.1). Further, not just monotonicity on rays, but quasiconcavity of the depth function (*i.e.*, convexity of central regions) will be assumed.

A unified framework is suggested to impute missing values by depth maximization, which extends iterative regression imputation. More precisely, consider the following iterative scheme: (1) initialize missing values arbitrarily using unconditional mean imputation; (2) impute a point $\boldsymbol{x}$ containing missing coordinates with the point $\boldsymbol{y}$ maximizing data depth conditioned on observed values $\boldsymbol{x}_{obs}$:

$$\boldsymbol{y} = \underset{\boldsymbol{z}_{miss}\in\mathbb{R}^{|miss|}\,,\,\boldsymbol{z}_{obs}=\boldsymbol{x}_{obs}}{\arg\max} D(\boldsymbol{z}|\boldsymbol{X})\,; \tag{2.2}$$

(3) iterate until convergence.

The solution of (2.2) can be non-unique (see Figure 1 in Supplementary Materials of M. et al. (2020) for an illustration) and the depth value may become zero immediately beyond the convex hull of the support of the distribution. To avoid these problems, *imputation by depth* (ID) of an $\boldsymbol{x}$ which has missing values with $\boldsymbol{y} = ID\big(\boldsymbol{x}, D(\cdot|\boldsymbol{X})\big)$ is suggested as follows:

$$\begin{aligned} ID\big(\boldsymbol{x}, D(\cdot|\boldsymbol{X})\big) &= ave\Big(\underset{\boldsymbol{u}\in\mathbb{R}^d\,,\,\boldsymbol{u}_{obs}=\boldsymbol{x}_{obs}}{\arg\min} \{\|\boldsymbol{u}-\boldsymbol{v}\|\,|\,\boldsymbol{v}\in D^{\alpha^*}(\boldsymbol{X})\}\Big), \\ \text{with}\quad \alpha^* &= \inf_{\alpha\in(0;1)}\big\{\alpha\,|\,D^{\alpha}(\boldsymbol{X})\cap\{\boldsymbol{z}\,|\,\boldsymbol{z}\in\mathbb{R}^d\,,\,\boldsymbol{z}_{obs}=\boldsymbol{x}_{obs}\}=\varnothing\big\}, \end{aligned} \tag{2.3}$$

where ave is the averaging operator. The imputation by iterative maximization of depth is summarized in Algorithm 2.1. The complexity of Algorithm 2.1 is $O\big(N_\epsilon n_{miss}\Omega(D)\big)$. It depends on the data geometry and on the missing values (through the number of outer-loop iterations N_ϵ necessary to achieve ϵ-convergence), the number of points containing missing values n_{miss}, and the depth-specific complexities for solving (2.3) $\Omega(D)$.

Algorithm 2.1 Single imputation

function IMPUTE.DEPTH.SINGLE($\boldsymbol{X}$)
  $\boldsymbol{Y} \leftarrow \boldsymbol{X}$
  $\boldsymbol{\mu} \leftarrow \hat{\boldsymbol{\mu}}^{(obs)}(\boldsymbol{X})$ ▷ Calculate mean, ignoring missing values
  for $i = 1 : n$ **do**
    if $miss(i) \neq \varnothing$ **then**
      $\boldsymbol{y}_{i,miss(i)} \leftarrow \boldsymbol{\mu}_{miss(i)}$ ▷ Impute with unconditional mean
  $I \leftarrow 0$
  repeat ▷ Iterate until convergence or maximal iteration
    $I \leftarrow I + 1$
    $\boldsymbol{Z} \leftarrow \boldsymbol{Y}$
    for $i = 1 : n$ **do**
      if $miss(i) \neq \varnothing$ **then**
        $\boldsymbol{y}_i \leftarrow ID\big(\boldsymbol{y}_i, D(\cdot|\boldsymbol{Z})\big)$ ▷ Impute with maximum depth
  until $\max_{i\in\{1,\ldots,n\},j\in\{1,\ldots,d\}} |\boldsymbol{y}_{i,j} - \boldsymbol{z}_{i,j}| < \epsilon$ **or** $I = I_{max}$
  return $\boldsymbol{Y}$

Theoretical properties for elliptical distributions

An elliptical distribution is defined as follows (see Fang et al. (1990), and Liu and Singh (1993) in the data depth context).

Definition 2.1 *A random vector X in $\mathbb{R}^d$ is elliptical if and only if there exists a vector $\boldsymbol{\mu}_X \in \mathbb{R}^d$ and $d \times d$ symmetric and positive semi-definite invertible matrix $\boldsymbol{S}_X = \boldsymbol{\Lambda}\boldsymbol{\Lambda}^\top$ such that for a random vector U uniformly distributed on the unit sphere $\mathcal{S}^{d-1}$ and a non-negative random variable R, it holds that $X \overset{\mathrm{d}}{=} \boldsymbol{\mu}_X + R\boldsymbol{\Lambda}U$. One writes then $X \sim \mathcal{E}_d(\boldsymbol{\mu}_X, \boldsymbol{S}_X, F_R)$, where F_R is the cumulative distribution function of the generating variate R.*

Theorem 2.1 shows that for an elliptical distribution, imputation of one point with a quasiconcave uniformly consistent depth converges to the center of the conditional distribution when conditioning on the observed values. Theorem 2.1 is illustrated in Figure 2 in Supplementary Materials of M. et al. (2020).

Theorem 2.1 (One row consistency) *(M. et al., 2020) Let $\boldsymbol{X} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)^\top$ be a data set in $\mathbb{R}^d$ drawn i.i.d. from $X \sim \mathcal{E}_d(\boldsymbol{\mu}_X, \boldsymbol{\Sigma}_X, F_R)$ with $d \geq 2$, F_R absolutely continuous with strictly decreasing density, and let $\boldsymbol{x} = (\boldsymbol{x}_{obs}, \boldsymbol{x}_{miss}) \in \mathbb{R}^d$ with $|obs(\boldsymbol{x})| \geq 1$. Further, let $D(\cdot|X)$ satisfy (P1)–(P5) and $D^\alpha(\boldsymbol{X}) \xrightarrow[n\to\infty]{a.s.} D^\alpha(X)$. Then for $\boldsymbol{y} = ID\big(\boldsymbol{x}, D(\cdot|\boldsymbol{X})\big)$,*

$$\left|\boldsymbol{y}_{miss} - \boldsymbol{\mu}_{X\,miss} - \boldsymbol{\Sigma}_{X\,miss,obs}\boldsymbol{\Sigma}^{-1}_{X\,obs,obs}(\boldsymbol{x}_{obs} - \boldsymbol{\mu}_{X\,obs})\right| \xrightarrow[n\to\infty]{a.s.} 0\,.$$

Theorem 2.2 states that if missing values constitute a portion of the sample but are in a single variable, the imputed values converge to the center of the conditional distribution when conditioning on the observed values.

Theorem 2.2 (One column consistency) *(M. et al., 2020) Let $\boldsymbol{X} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)^\top$ be a data set in $\mathbb{R}^d$ drawn i.i.d. from $X \sim \mathcal{E}_d(\boldsymbol{\mu}_X, \boldsymbol{\Sigma}_X, F_R)$ with $d \geq 2$, F_R absolutely continuous with strictly decreasing density, and let $miss(i) = \{j\}$ with probability $p \in (0,1)$ for a fixed $j \in \{1, \ldots, d\}$. Let $D(\cdot|Z)$ satisfy (P1)–(P5) and $D^\alpha(\boldsymbol{Z}) \xrightarrow[n\to\infty]{a.s.} D^\alpha(Z)$ for $Z = (1-p)X + pZ'$ with $Z' = \boldsymbol{\mu}_{X\,j} - \boldsymbol{\Sigma}_{X\,j,-j}\boldsymbol{\Sigma}^{-1}_{X\,-j,-j}(X_{-j} - \boldsymbol{\mu}_{X\,-j})$. Further, let $\boldsymbol{Y}$ exist such that $\boldsymbol{y}_i = ID\big(\boldsymbol{x}_i, D(\cdot|\boldsymbol{Y})\big)$ if $miss(i) = \{j\}$ and $\boldsymbol{y}_i = \boldsymbol{x}_i$ otherwise. Then, for all i with $miss(i) = \{j\}$ and denoting $-j$ for $\{1, ..., d\} \setminus \{j\}$,*

$$\left|\boldsymbol{y}_{i,j} - \boldsymbol{\mu}_{X\,j} - \boldsymbol{\Sigma}_{X\,j,-j}\boldsymbol{\Sigma}^{-1}_{X\,-j,-j}(\boldsymbol{x}_{i,-j} - \boldsymbol{\mu}_{X\,-j})\right| \xrightarrow[n\to\infty]{a.s.} 0\,.$$

2.3 By-depth analysis

The generality of the proposed methodology lies in the possibility of using any notion of depth which defines imputation properties. Let us focus here on imputation with Mahalobis, zonoid, and halfspace depths. These are of particular interest because they are quasiconcave and require two, one, and zero first moments of the underlying probability measure, respectively.

Corollary 2.1 *(M. et al., 2020) Theorems 2.1 and 2.2 hold for the halfspace depth, for the zonoid depth if* $\mathbb{E}[\|X\|] < \infty$*, and for the Mahalanobis depth if* $\mathbb{E}[\|X\|^2] < \infty$*.*

In addition, the function $f(\boldsymbol{z}_{miss}) = D(\boldsymbol{z}|\boldsymbol{X})$ subject to $\boldsymbol{z}_{obs} = \boldsymbol{x}_{obs}$ in equation (2.2), iteratively optimized in Algorithm 2.1, is quadratic for the Mahalanobis depth, continuous inside $conv(\boldsymbol{X})$ (the smallest convex set containing $\boldsymbol{X}$) for the zonoid depth, and stepwise discrete for the halfspace depth, which in all cases leads to efficient implementations. For a trivariate Gaussian sample, $f(\boldsymbol{z}_{miss})$ is depicted in Figure 1 in Supplementary Materials of M. et al. (2020).

The use of a non-quasiconcave depth (*e.g.*, simplicial or spatial depth) results in non-convex optimization when maximizing depth, and this non-stability impedes numerical convergence of the algorithm.

2.3.1 Mahalanobis depth

Imputation with the Mahalanobis depth is related to existing methods. First, let us show the link with the minimization of the covariance determinant.

Proposition 2.1 (Covariance determinant is quadratic in a point's missing entries) *(M. et al., 2020) Let* $\boldsymbol{X}(\boldsymbol{y}) = \left(\boldsymbol{x}_1, \ldots, (\boldsymbol{x}_{i,1}, \ldots, \boldsymbol{x}_{i,|obs(i)|}, \boldsymbol{y}^\top)^\top, \ldots, \boldsymbol{x}_n\right)^\top$ *be a* $n \times d$ *matrix with* $\boldsymbol{\Sigma}_{\boldsymbol{X}}(\boldsymbol{y})$ *invertible for all* $\boldsymbol{y} \in \mathbb{R}^{|miss(i)|}$*. Then* $|\boldsymbol{\Sigma}_{\boldsymbol{X}}(\boldsymbol{y})|$ *is quadratic and globally minimized in* $\boldsymbol{y} = \boldsymbol{\mu}_{\boldsymbol{X}\, miss(i)}(\boldsymbol{y}) + \boldsymbol{\Sigma}_{\boldsymbol{X}\, miss(i),obs(i)}(\boldsymbol{y})\boldsymbol{\Sigma}^{-1}_{\boldsymbol{X}\, obs(i),obs(i)}(\boldsymbol{y})\big((\boldsymbol{x}_{i,1}, \ldots, \boldsymbol{x}_{i,|obs(i)|}) - \boldsymbol{\mu}_{\boldsymbol{X}\, obs(i)}\big)$*.*

From Proposition 2.1 it follows that the minimum of the covariance determinant is unique and the determinant itself decreases at each iteration. Thus, to impute points with missing coordinates one-by-one and iterate until convergence constitutes the block coordinate descent method, which can be proved to numerically converge due to Proposition 2.7.1 from Bertsekas (1999) (as long as $\boldsymbol{\Sigma}_{\boldsymbol{X}}$ is invertible).

Further, Theorem 2.3 states that imputation using the maximum Mahalanobis depth, iterative (multiple-output) regression, and regularized PCA (Josse and Husson, 2012) with $S = d - 1$ dimensions, all converge to the same imputed sample.

Theorem 2.3 *(M. et al., 2020) Suppose imputation of* $\boldsymbol{X} = (\boldsymbol{X}_{miss}, \boldsymbol{X}_{obs})$ *in* $\mathbb{R}^d$ *with* $\boldsymbol{Y} = (\boldsymbol{y}_1, ..., \boldsymbol{y}_n)^\top$ *so that* $\boldsymbol{y}_i = \arg\max_{\boldsymbol{z}_{obs(i)} = \boldsymbol{y}_{obs(i)}} D_{\text{Mah}}(\boldsymbol{z}|\boldsymbol{Y})$ *for each* i *with* $|miss(i)| > 0$ *and* $\boldsymbol{y}_i = \boldsymbol{x}_i$ *otherwise. Then for each such* $\boldsymbol{y}_i$*, it also holds that:*

- $\boldsymbol{x}_i$ *is imputed with the* **conditional mean***:*

$$\boldsymbol{y}_{i,miss(i)} = \boldsymbol{\mu}_{\boldsymbol{Y}\, miss(i)} + \boldsymbol{\Sigma}_{\boldsymbol{Y}\, miss(i),obs(i)}\boldsymbol{\Sigma}^{-1}_{\boldsymbol{Y}\, obs(i),obs(i)}(\boldsymbol{x}_{obs(i)} - \boldsymbol{\mu}_{\boldsymbol{Y}\, obs(i)})$$

  which is equivalent to **single**- *and* **multiple**-*output* **regression**,

- $\boldsymbol{Y}$ *is a* **stationary point** *of* $|\boldsymbol{S}_{\boldsymbol{X}}(\boldsymbol{X}_{miss})|$*:* $\frac{\partial|\boldsymbol{S}_{\boldsymbol{X}}|}{\partial \boldsymbol{X}_{miss}}(\boldsymbol{Y}_{miss}) = \boldsymbol{0}$*, and*

- *each missing coordinate j of $\boldsymbol{x}_i$ is imputed with* **regularized PCA** *as in Josse & Husson (2012) with any $0 < \sigma^2 \leq \lambda_d$ and with $\boldsymbol{X} - \boldsymbol{\mu}_{\boldsymbol{X}} = \boldsymbol{U}\boldsymbol{\Lambda}^{\frac{1}{2}}\boldsymbol{V}^\top$ the singular value decomposition (SVD): $\boldsymbol{y}_{i,j} = \sum_{s=1}^{d} \boldsymbol{U}_{i,s}\sqrt{\frac{\lambda_s - \sigma^2}{\lambda_s}}\boldsymbol{V}_{j,s} + \boldsymbol{\mu}_{\boldsymbol{Y}\,j}$.*

The first point of the theorem sheds light on the connection between imputation by Mahalanobis depth and the iterative regression imputation of Section 2.2.1. When the Mahalanobis depth is used in Algorithm 2.1, each $\boldsymbol{x}_i$ with missingness in $miss(i)$ is imputed by the multivariate conditional mean as in equation (2.1), and thus lies in the $\big(d-|miss(i)|\big)$-dimensional multiple-output regression subspace of $\boldsymbol{X}_{\cdot,miss(i)}$ on $\boldsymbol{X}_{\cdot,obs(i)}$. This subspace is obtained as the intersection of the single-output regression hyperplanes $\boldsymbol{X}_{\cdot,j}$ on $\boldsymbol{X}_{\cdot,\{1,\dots,d\}\setminus\{j\}}$ for all $j \in miss(i)$ corresponding to missing coordinates. The third point strengthens the method as imputation with regularized PCA has proved to be highly efficient in practice due to its sticking to low-rank structure of importance and ignoring noise.

The complexity of imputing a single point with the Mahalanobis depth is $O(nd^2 + d^3)$. Despite its good properties, it is not robust to outliers. However, robust estimates for $\boldsymbol{\mu}_{\boldsymbol{X}}$ and $\boldsymbol{\Sigma}_{\boldsymbol{X}}$ can be used, *e.g.*, the minimum covariance determinant ones (MCD, see Rousseeuw and Van Driessen, 1999).

2.3.2 Zonoid depth

For the recall of the definition of zonoid depth and its trimmed regions (see also empirical version), the reader is referred to Definition 1.6 in Section 1.1.2.

Imputation of a point $\boldsymbol{x}_i$ in Algorithm 1 is then performed by a slight modification of the linear programming for computation of zonoid depth with variables γ and $\boldsymbol{\lambda} = (\lambda_1, ..., \lambda_n)^\top$:

$$\min \gamma \quad \text{s.t.} \quad \boldsymbol{X}_{\cdot,obs(i)}^\top \boldsymbol{\lambda} = \boldsymbol{x}_{i,obs(i)}\,, \boldsymbol{\lambda}^\top \mathbf{1}_n = 1\,, \gamma \mathbf{1}_n - \boldsymbol{\lambda} \geq \mathbf{0}_n\,, \boldsymbol{\lambda} \geq \mathbf{0}_n\,.$$

Here $\boldsymbol{X}_{\cdot,obs(i)}$ stands for the completed $n \times |obs(i)|$ data matrix containing columns corresponding only to non-missing coordinates of $\boldsymbol{x}_i$, and $\mathbf{1}_n$ (respectively $\mathbf{0}_n$) is a vector of ones (respectively zeros) of length n. In the implementation, the simplex method is used, which is known for being fast despite its exponential complexity. This implies that, for each point $\boldsymbol{x}_i$, imputation is performed by the weighted mean:

$$\boldsymbol{y}_{i,miss(i)} = \boldsymbol{X}_{\cdot,miss(i)}^\top \boldsymbol{\lambda}\,,$$

the average of the maximum number of equally weighted points. Additional insight on the position of imputed points with respect to the sample can be gained by inspecting the optimal weights λ_i. Zonoid imputation is related to local methods such as as kNN imputation, as only some of the weights are positive.

2.3.3 Halfspace depth

Halfspace depth has been introduced to this manuscript in Definition 1.2 of Section 1.1.2, with its empirical version defined by (1.10) in Section 1.2.1.

With nonparametric imputation by halfspace depth, one can expect that after convergence of Algorithm 2.1, for each point initially containing missing values, it holds that $\boldsymbol{y}_i = \arg\max_{\boldsymbol{z}_{obs}=\boldsymbol{x}_{obs}} \min_{\boldsymbol{u}\in\mathcal{S}^{d-1}} \left|\left\{k \,:\, \boldsymbol{y}_k^\top \boldsymbol{u} \geq \boldsymbol{z}^\top \boldsymbol{u},\, k \in \{1,...,n\}\right\}\right|$. Thus, imputation is performed according to the maximin principle based on criteria involving indicator functions, which implies robustness of the solution. Note that as the halfspace depth is not continuous, the searched-for maximum (2.2) may be non-unique (see Figure 1 (top right) in Supplementary Materials of M. et al. (2020)); imputation is then performed with the barycenter of the maximizing arguments (2.3). Due to the combinatorial nature of the halfspace depth, to speed up implementation, the Nelder-Mead downhill-simplex algorithm is run $2d$ times, and the average over the solutions is taken. The imputation is illustrated in Figure 3 in Supplementary Materials of M. et al. (2020).

The halfspace depth can be computed exactly (Dyckerhoff and M., 2016) with complexity $O(n^{d-1}\log n)$, although to avoid computational burden its approximation with random directions is implemented(Dyckerhoff, 2004) having complexity $O(kn)$, with k denoting the number of random directions. All of the experiments are performed with exactly computed halfspace depth, unless stated otherwise.

2.3.4 Beyond ellipticity: local depth

Imputation with the so-called "global depth" may be appropriate in applications even if the data moderately deviate from ellipticity. However, it can fail when the distribution has non-convex support or several modes. A solution is to use the local depth in Algorithm 2.1.

Definition 2.2 (Paindaveine and Van Bever, 2013) *For a depth $D(\cdot|X)$, the β-local depth is defined as $LD^\beta(\cdot, X) : \mathbb{R}^d \to \mathbb{R}^+ : \boldsymbol{x} \mapsto LD^\beta(\boldsymbol{x}, X) = D(\boldsymbol{x}|X^{\beta,\boldsymbol{x}})$ with $X^{\beta,\boldsymbol{x}}$ the conditional distribution of X conditioned on $\bigcap_{\alpha\geq 0,\, P_Y(D^\alpha(Y))\geq\beta} D^\alpha(Y)$, where Y has the distribution $P_Y = \frac{1}{2}P_X + \frac{1}{2}P_{2\boldsymbol{x}-X}$.*

The locality level β should be chosen in a data-driven way, for instance by cross-validation. An important advantage of this approach is that any "global depth" (for exact set of properties the depth notion should satisfy, see Paindaveine and Van Bever, 2013) can be plugged in to the local depth. Here, it is suggested to use the Nelder-Mead algorithm to enable imputation with maximum local depth regardless of the chosen depth notion.

2.3.5 Dealing with outsiders

A number of depths that exploit the geometry of the data are equal to zero beyond $conv(\boldsymbol{X})$, including the zonoid and halfspace depths. Although (2.3) deals with this situation, for a finite sample it means that points with missing values having the maximal value in at least one of the observed coordinates will never move from the initial imputation because they will become vertices of the $conv(\boldsymbol{X})$. For the same reason, other points to be imputed and lying exactly on the $conv(\boldsymbol{X})$ will not move much during imputation iterations. As such points are not numerous and would need to move quite substantially to influence imputation

quality, they are imputed—during the initial iterations—using the spatial depth function (Vardi and Zhang, 2000), which is everywhere non-negative. This resembles the so-called "outsider treatment" introduced by Lange et al. (2014b). Another possibility is to extend the depth beyond $conv(\boldsymbol{X})$, see *e.g.*, Einmahl et al. (2015) for the halfspace depth.

2.4 Assessing the quality of imputation

Quality measure for the single imputation is chosen using the following logic. Producing a single data set without missing entries is crucial from application point of view. In practice, a data analyst wishes to employ a statistical software of choice, while very little of those incorporate the ability to accept missing values in the input arguments. Imputation can be required due to abstention in surveys, where the institution has to provide a complete data set. Such missing entries could also reveal identity, or were (deliberately) removed for anonymity-preserving purposes.

Since the goal of the future statistical analysis is not clear at the imputation stage, logically, single imputation is assessed by its ability to impute (*i.e.* predict the absent cells of data points) close to the (unknown) missing values. Thus, the root mean square error (RMSE) appears a natural measure for the quality of imputation.

In this spirit, the prediction abilities are assessed of halfspace, zonoid, and Mahalanobis depth imputation, and the robust Mahalanobis depth imputation using MCD mean and covariance estimates, with the robustness parameter chosen in an optimal way due to knowledge of the simulation setting. Their performance is measured against the competitors: conditional mean imputation based on EM estimates (Dempster et al., 1977) of the mean and covariance matrix; regularized PCA imputation with rank 1 and 2 (Josse and Husson, 2012); two nonparametric imputation methods: random forest (Stekhoven and Bühlmann, 2012, using the default implementation in the `R`-package `missForest`), and kNN imputation (Troyanskaya et al., 2001) choosing k from $\{1, \ldots, 15\}$ minimizing the imputation error over 10 validation sets as in Stekhoven and Bühlmann (2012). Mean and oracle (if possible) imputations are used to benchmark the results.

In Section 4, M. et al. (2020) (see additionally Supplementary Materials of the article) provide an extensive simulation study that considers elliptical distributions (see Definition 2.1) with the Student-t generator and with outlier contamination, the missing at random (MAR) mechanism, the low-rank model, contamination in higher dimensions, skewed distribution and a distribution with non-convex support, as well as real-data study on four data sets: Banknotes, Glass, and Blood Transfusion data downloaded from the UCI Machine Learning Repository (Dua and Graff, 2017) and the Cow data set. Below, experiments for the contaminated elliptical distribution (also for higher-dimensional data, see Section 2.4.1), MAR-values (Section 2.4.2), skewed distribution and a distribution with non-convex support (Section 2.4.3) are shown.

More precisely, each single experiment (for one method) is conducted as follows: (a) a data set without missing values is generated (provided for real data sets); (b) missing values

are introduced due to MCAR (or MAR where mentioned) mechanism; (c) imputation (of all missing values) is performed; (d) RMSE is computed comparing to the original data set from (a). For each imputation method, this sequence of actions is performed multiple times, and medians and MADs of the corresponding RMSEs are reported (or the RMSE boxplots).

2.4.1 Contaminated elliptical setting

Here, in each run, 100 points are generated according to an elliptical distribution (Definition 2.1) with $\boldsymbol{\mu}_2 = (1,1,1)^\top$ and the shape $\boldsymbol{S}_2 = \big((1,1,1)^\top, (1,4,4)^\top, (1,4,8)^\top\big)$, where F_R is the univariate Student-t distribution ranging in number of degrees of freedom (d.f.) from the Gaussian to the Cauchy: $t = \infty, 10, 5, 3, 2, 1$. Further, 15% of outliers are added (which do not contain missing values) that stem from the Cauchy distribution with the same parameters $\boldsymbol{\mu}_2$ and $\boldsymbol{S}_2$. For each of the 1000 simulations, 25% of values are removed completely at random (MCAR), and median and MAD of RMSE of each imputation method are computed.

Dist.	D^{Tuk}	D^{zon}	D^{Mah}	$D^{Mah}_{MCD,75}$	EM	regPCA1	regPCA2	kNN	RF	mean	oracle
$t\,\infty$	**1.751**	1.86	1.945	*1.81*	1.896	1.958	1.945	1.859	1.86	2.23	1.563
	(**0.2317**)	(0.3181)	(0.4299)	(*0.239*)	(0.3987)	(0.4495)	(0.4328)	(0.2602)	(0.2332)	(0.3304)	(0.1849)
$t\,10$	**1.942**	2.087	2.165	*2.022*	2.112	2.196	2.165	2.051	2.047	2.48	1.733
	(**0.2976**)	(0.4295)	(0.5473)	(*0.3128*)	(0.5226)	(0.5729)	(0.5479)	(0.3143)	(0.3043)	(0.4163)	(0.2266)
$t\,5$	**2.178**	2.333	2.421	*2.231*	2.376	2.398	2.421	2.315	2.325	2.766	1.939
	(**0.3556**)	(0.4924)	(0.6026)	(*0.381*)	(0.5715)	(0.6035)	(0.5985)	(0.3809)	(0.3946)	(0.528)	(0.2979)
$t\,3$	**2.635**	2.864	2.935	*2.664*	2.828	2.916	2.93	2.797	2.838	3.34	2.356
	(**0.6029**)	(0.7819)	(0.8393)	(*0.5877*)	(0.7773)	(0.8221)	(0.8384)	(0.6045)	(0.6228)	(0.7721)	(0.4946)
$t\,2$	**3.763**	4.082	4.136	*3.783*	4.036	4.09	4.14	3.955	4.026	4.623	3.323
	(**1.17**)	(1.535)	(1.501)	(*1.224*)	(1.518)	(1.585)	(1.503)	(1.265)	(1.354)	(1.561)	(1.04)
$t\,1$	*17.17*	20.43	20.27	**16.46**	19.01	19.81	20.53	18.96	19.04	21.04	14.44
	(*13.27*)	(15.99)	(15.91)	(**12.94**)	(15.21)	(16.15)	(16.28)	(14.73)	(14.62)	(15.56)	(11.33)

Table 2.1: *Median and MAD of the RMSEs of the imputation for* 100 *points drawn from elliptically symmetric Student-t distributions, with* $\boldsymbol{\mu}_2$ *and* $\boldsymbol{S}_2$ *contaminated with* 15% *of outliers, and* 25% *of MCAR values on non-contaminated data, repeated 1000 times.*

As expected, Table 2.1 shows that the best RMSEs are obtained by the robust depth-based imputation methods: Tukey depth and Mahalanobis depth with MCD estimates (for 5% and 15% of missing values see Supplementary Materials of M. et al., 2020). Being restricted to a neighborhood, nonparametric methods (only) sometimes impute based on non-outlying points, and thus perform less well as the preceding group. The rest of the included imputation methods cannot deal with the contaminated data and perform rather poorly.

To check the resistance to outliers in higher dimensions, consider a simulation setting similar to this from above, in dimension 6, with a normal multivariate distribution with $\boldsymbol{\mu}_3 = (0,\ldots,0)^\top$ and a Toeplitz covariance matrix $\boldsymbol{S}_3$ (having $\sigma_{i,j} = 2^{-|i-j|}$ as entries). The data are contaminated with 15% of outliers and have 15% of MCAR values on non-contaminated data. The Tukey depth is approximated using 1000 random directions. Figure 2.3 (left) shows that the Tukey depth imputation has high predictive quality, comparable to that of the random forest imputation even with only 1000 random directions; zonoid and robust Mahalanobis depths follow them.

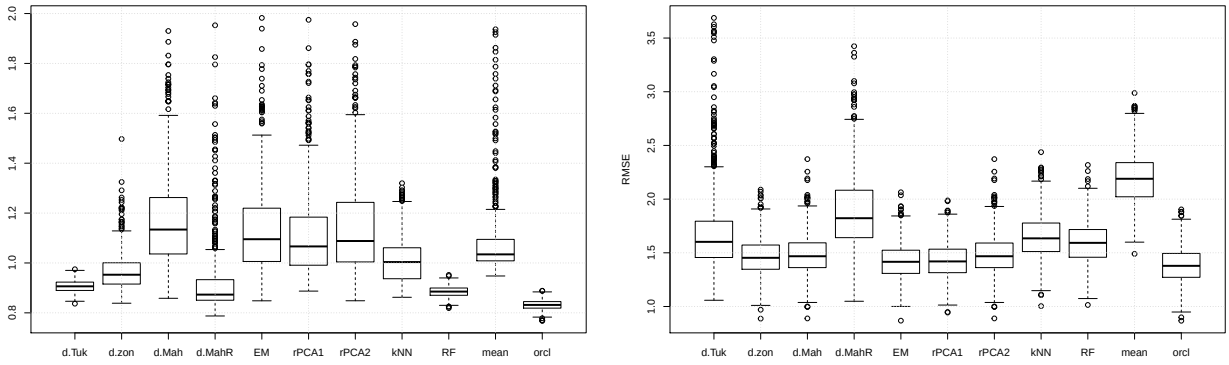


Figure 2.3: *Left: RMSE boxplots for different imputation methods for* 1000 *points drawn from a* 6*-dimensional Gaussian distribution with* $\boldsymbol{\mu}_3$ *and* $\boldsymbol{S}_3$ *contaminated with* 15% *of outliers, and* 15% *of MCAR values on non-contaminated data, over* 500 *repetitions. Right: RMSE boxplots for* 100 *points drawn from a correlated* 3*-dimensional Gaussian distribution with* $\boldsymbol{\mu}_4$ *and* $\boldsymbol{S}_4$ *with MAR values, over* 1000 *repetitions.*

2.4.2 The MAR setting

Next, highly correlated Gaussian data are generated by setting $\boldsymbol{\mu}_4 = (1, 1, 1)$ and the covariance matrix to $\boldsymbol{S}_4 = \big((1, 1.75, 2)^\top, (1.75, 4, 4)^\top, (2, 4, 8)^\top\big)$. Then, missing values are inserted according to the MAR mechanism: the first and third variables are missing depending on the value of the second variable. Figure 2.3 (right) shows the boxplots of the RMSEs. As expected, semiparametric methods (EM, regularized PCA and Mahalanobis depth) perform close to the oracle imputation. The good performance of the rank 1 regularized PCA can be explained by the high correlation between variables. The zonoid depth imputes well despite having no parametric knowledge. Nonparametric methods are unable to capture the correlation, while robust methods "throw away" points possibly containing valuable information.

2.4.3 Skewed and non-convexly supported distributions

First, consider only a slight deviation from ellipticity, for which (also theoretically) depth-based imputation provides good results. Let us simulate 150 points from a skewed normal distribution (Azzalini and Capitanio, 1999), insert 15% MCAR values, and impute them with global (halfspace, zonoid and Mahalanobis) depths and their local versions (see Section 2.3.4). This is shown in Figure 2.4. In this setting, both global and local imputation perform similarly.

Further, let us consider an extreme departure from elliplicity with the moon-shaped example from Paindaveine and Van Bever (2013). For this, 150 bivariate observations from $(X_1, X_2)^\top$ with $X_1 \sim U(-1, 1)$ and $X_2|X_1 = x_1 \sim U\big(1.5(1 - x_1^2), 2(1 - x_1^2)\big)$ are generated, and 15% of MCAR values on X_2 are introduced, see Figure 2.5 (left). Figure 2.5 (right) shows boxplots of the RMSE for single imputation using local halfspace, zonoid and Mahalanobis depths. If the depth and value of β are properly chosen (this can be achieved by cross-

validation), the local-depth imputation considerably outperforms the classical methods as well as the global depth.

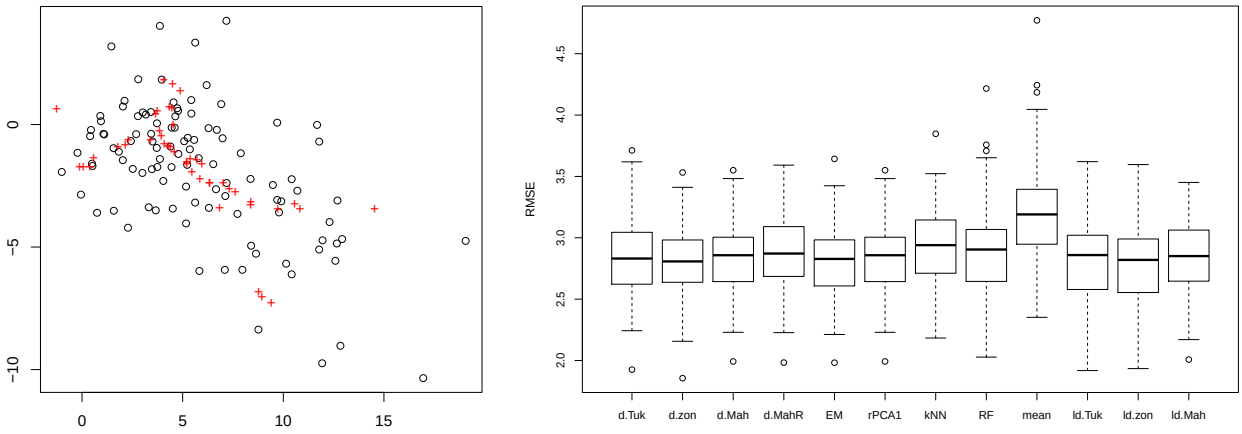


Figure 2.4: *Left: An example of halfspace depth imputation (pluses). Right: boxplots of RMSEs of the prediction for 150 points drawn from a skewed distribution with* 15% *MCAR, over 100 repetitions; ld.* stands for the local depth with* $\beta = 0.8$.

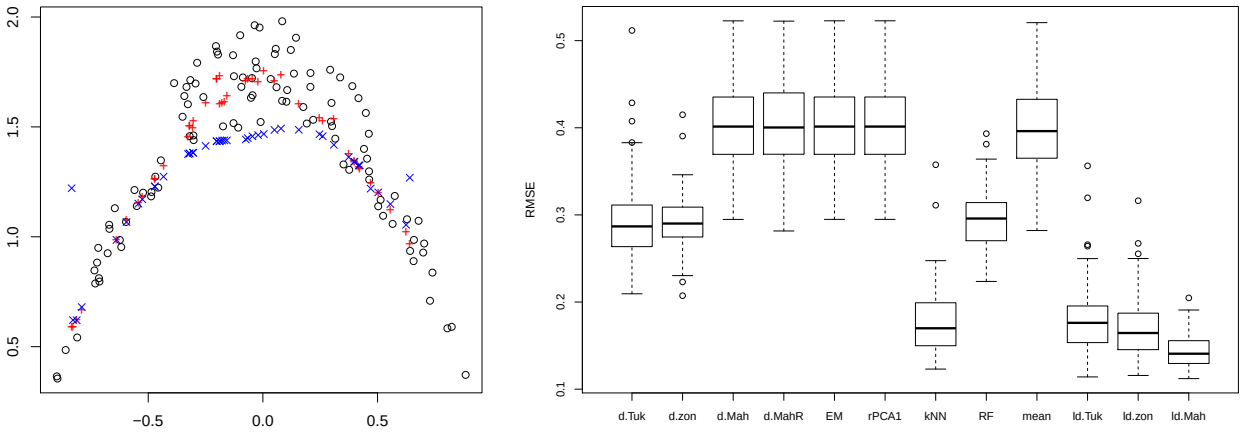


Figure 2.5: *Left: Comparison of global (crosses) and local (pluses) halfspace depth imputation. Right: boxplots of RMSEs of predictions for 150 points drawn from the moon-shaped distribution with* 15% *MCAR values in the second coordinate, over 100 repetitions; ld.* stands for the local depth with* $\beta = 0.2$.

Chapter 3

Novel notion: data depth for curves

3.1 Introduction to functional depth

For clarity, let us start with recalling the concept of *statistical depth* in the functional framework. After a brief review of recent advances in this field, notions of functional depth functions relevant for understanding the subsequent material are listed in Section 3.1.1.

Let $d \geq 1$ be an integer. Let $(\mathbb{R}^d, |\cdot|_2)$ be the d-dimensional Euclidean space, $\mathscr{C}([0,1], \mathbb{R}^d)$ be the space of continuous functions defined on the interval $[0,1]$ and taking values in $\mathbb{R}^d$. In this chapter and the following Chapter 4 (where mostly the case $d = 1$ is addressed), the random function F is considered which takes its values in $\mathscr{C}([0,1], \mathbb{R}^d)$.

Where applicable, and without loss of generality, restrict to functions defined on $[0,1]$. In practice, only a finite dimensional marginal $\big(F(t_1), \ \ldots, \ F(t_p)\big)$, $t_1 < \ldots < t_p$, $p \geq 1$ and $(t_1, \ \ldots, \ t_p) \in [0,1]^p$ can be observed, while dimension marginals of F will be mentioned as F_j for $j = 1, ..., d$. Considering $\big(F(t_1), \ \ldots, \ F(t_p)\big)$ as a discretized curve rather than a simple random vector of dimension p permits to take into account the dependence structure between the measurements over the functional argument, especially when the argument points t_i are not equispaced. To come back to a function from discrete values, interpolation procedures or approximation schemes based on appropriate dictionaries can be used, combined with a preliminary smoothing step when the observations are noisy. From a statistical perspective, the analysis is based on a functional data set $\mathcal{F} = \{\mathbf{f}_1, \ \ldots, \ \mathbf{f}_n\}$ composed of $n \geq 1$ independent realizations of the stochastic process F. One may refer to, *e.g.*, Ramsay and Silverman (2005) for a deep view on functional data analysis.

Depths in a functional framework have been first considered in Fraiman and Muniz (2001), where it is proposed to define functional depths as simple integrals over the argument of a multivariate depth function D. Due to the averaging effect, local changes for the curve $\mathbf{f}$ only induce slight modifications of the depth value. Later, alternative functional depths have been introduced, see López-Pintado and Romo (2009, 2011) for depths based on the geometry of the set of curves, Chakraborty and Chaudhuri (2014) for a notion of depth based on the L_2 distance or Dutta et al. (2011) for a functional version of the halfspace depth. Since the axiomatic frameworks introduced in Zuo and Serfling (2000) or Mosler (2013) for multivariate depths are no longer adapted to the richness of the topological structure of

functional spaces, new sets of desirable properties were suggested in 2012 by Mosler and Polyakova (2018) and in 2016 by Nieto-Reyes and Battey (2016) (for their refinements, see also Gijbels and Nagy, 2017).

3.1.1 Relevant notions

Let us review the notions of functional data depth relevant for this manuscript. First, regard the functional projection depth.

Definition 3.1 ***Functional projection depth*** *(Hubert et al., 2015) of a univariate function $f \in \mathscr{C}([0,1],\mathbb{R})$ w.r.t. $F : \Omega \to \mathscr{C}([0,1],\mathbb{R})$ is defined as the integral (Fraiman and Muniz, 2001) of the projection (Zuo and Serfling, 2000) depth:*

$$FD_{\text{Proj}}(f|F) = \int_0^1 D_{\text{Proj}}\big(f(t)|F(t)\big)dt\,, \tag{3.1}$$

where $D_{\text{Proj}}(\cdot|\cdot)$ is the multivariate projection depth defined in (1.4).

Below, let us consider functional depths defined for multiple-output functions $\mathbf{f} = (f_1, ..., f_d) : [0,1] \to \mathbb{R}^d$.

Definition 3.2 ***Modified multivariate band depth*** *(Ieva and Paganoni, 2013) of $\mathbf{f} \in \mathscr{C}([0,1],\mathbb{R}^d)$ w.r.t. $F : \Omega \to \mathscr{C}([0,1],\mathbb{R}^d)$ is defined as*

$$FD^J_{\text{mMBD}}(\mathbf{f}|F) = \sum_{k=1}^{d} p_k FD^J_{\text{mBD}}(f_k|F_k)\,, \quad p_k > 0\ \forall\ k = 1, ..., d, \quad \sum_{k=1}^{d} p_k = 1\,, \tag{3.2}$$

with F_k being the kth marginal and univariate ***modified band depth*** *(López-Pintado and Romo, 2009) defined as*

$$FD^J_{\text{mBD}}(f|F) = \sum_{j=2}^{J} \mathbb{E}[\tilde{\lambda}\big(E(f; F_{i_1}, ..., F_{i_j})\big)]\,, \tag{3.3}$$

where $F_{i_1}, ..., F_{i_j}$ are independent copies of the random function following the same distribution as F, $E(f; F_{i_1}, ..., F_{i_j}) = \{t \in [0,1]\,, \min_{r=i_1,...,i_j} F_r(t) \le f(t) \le \max_{r=i_1,...,i_j} F_r(t)\}$, $\tilde{\lambda}(f) = \lambda\big(E(f; F_{i_1}, ..., F_{i_j})\big)/\lambda([0,1])$, and λ is the Lebesgue measure on $[0,1]$.

J is usually chosen $= 2$ to reduce the computational cost (see, *e.g.*, Section 4.2.3 for a similar evidence) and will be omitted in the sequel. $p_k = 1/d$ to avoid tuning it, other choices are possible though.

Indeed, modified band depth corresponds to the application of the integtated depth idea of Fraiman and Muniz (2001) to the simplicial depth (1.6) defined by Liu (1990) with $d = 1$, and the modified multivariate band depth of (Ieva and Paganoni, 2013) generalizes it to d-dimensional functional data by weighted average aggregation of dimension-marginal depths. Another angle consists in aggregation by argument marginals, and is described right below.

Definition 3.3 ***Simplicial band depth*** *(López-Pintado et al., 2014) is defined as a probability that the trajectory of a function* $\mathbf{f}$ *is inside a random region in* $[0,1]\times\mathbb{R}^d$ *determined by random simplices at each argument value* $t\in[0,1]$*:*

$$FD^J_{\mathrm{SBD}}(\mathbf{f}|F)=\mathbb{P}_F\big[\mathbf{f}(t)\ \in\ simplex\big(F_{i_1}(t),...,F_{i_{d+1}}(t)\big)\,,\forall\, t\in[0,1]\big]\,, \tag{3.4}$$

where $F_{i_1},...,F_{i_{d+1}}$ *are independent copies of the random function following the distribution of* F *as before, and* $simplex(\cdot)$ *designates a simplex with arguments being its vertices.*

Definition 3.4 ***Modified simplicial band depth*** *(López-Pintado et al., 2014) is obtained from the simplicial band depth by replacing the indicator function with the Lebesgue measure (*λ *as before):*

$$FD^J_{\mathrm{mSBD}}(\mathbf{f}|F)=\mathbb{E}_F\big[\lambda\big(t\in[0,1]\ :\ \mathbf{f}(t)\ \in\ simplex(F_{i_1}(t),...,F_{i_{d+1}}(t))\big)\big]\,. \tag{3.5}$$

Definition 3.5 ***Multivariate functional halfspace depth*** *(Claeskens et al., 2014) is defined as weighted integral over the argument-marginal halfspace depth:*

$$FD^J_{\mathrm{MFHD}}(\mathbf{f}|F)=\int_0^1 D_H\big(\mathbf{f}(t)|F(t)\big)\cdot w(t)dt\,, \tag{3.6}$$

with the weights being either constant over the functional argument or reflecting the variability with the argument:

$$w(t)=\frac{vol\big(D^\alpha_{\mathrm{H}}\big(F(t)\big)\big)}{\int_0^1 vol\big(D^\alpha_H\big(F(u)\big)\big)du}\,, \tag{3.7}$$

i.e., proportional to the volume of the halfspace α*-region at the given value of the functional argument.*

Definition 3.6 ***Skew-adjusted functional projection depth*** *(Hubert et al., 2015) is defined as follows:*

$$FD^J_{\mathrm{saPRJ}}(\mathbf{f}|F)=\int_0^1 D_{\mathrm{SPD}}\big(\mathbf{f}(t)|F(t)\big)dt\,,$$

where the (multivariate) skew-adjusted projection depth of $\boldsymbol{x}\in\mathbb{R}^d$ *w.r.t. a random vector* X *in* $\mathbb{R}^d$ *is defined as*

$$D_{\mathrm{SPD}}(\boldsymbol{x}|X)=\frac{1}{1+O_{\mathrm{A}}(\boldsymbol{x}|X)}\,,$$

while the adjusted outlyingness $O_A(\cdot|\cdot)$ *is:*

$$O_{\mathrm{A}}(\boldsymbol{x}|X)=\sup_{\boldsymbol{u}\in\mathbb{S}^{d-1}}\begin{cases}\frac{\boldsymbol{u}^\top\boldsymbol{x}-med(\boldsymbol{u}^\top X)}{w_2(\boldsymbol{u}^\top X)-med(\boldsymbol{u}^\top X)} & \text{if } \boldsymbol{u}^\top X>med(\boldsymbol{u}^\top X)\\ \frac{med(\boldsymbol{u}^\top X)-\boldsymbol{u}^\top\boldsymbol{x}}{med(\boldsymbol{u}^\top X)-w_1(\boldsymbol{u}^\top X)} & \text{if } \boldsymbol{u}^\top X<med(\boldsymbol{u}^\top X)\end{cases}$$

with

$$w_1(Y)=Q_1(Y)-1.5\mathbf{e}^{-4MC(Y)}IRQ(Y)\,,$$
$$w_2(Y)=Q_3(Y)+1.5\mathbf{e}^{+3MC(Y)}IRQ(Y)\,,$$

Q_1 *and* Q_3 *being univariate* 1^{st} *and* 3^{rd} *quartiles of the random variable* Y*, IQR being its interquantile range, and MC denoting the* medcouple *of Brys et al. (2004).*

These last five multivariate functional depths will be further shortly denoted as mMBD, SBD, mSBD, MFHD, and saPRJ, respectively.

3.2 Necessity of a depth for unparametrized curves

In recent years, statisticians have been facing complex types of data that they analyze using a functional depth (Fraiman and Muniz, 2001, López-Pintado and Romo, 2009, Narisetty and Nair, 2016) or even a multivariate functional depth approach (Claeskens et al., 2014). These new techniques have proven to be very useful for data visualization, to estimate a measure of location or spread, to detect outliers (see also Hubert et al., 2015), for clustering, or to detect if two groups of functions come from the same population.

However, functional depths are sensitive to parametrization of curves. Figures 3.1 and 3.2 illustrate the impact of two different parametrizations on depths rankings of curves provided by the *multivariate functional halfspace depth* (MFHD) developped by Claeskens et al. (2014) (with weight function set to a constant) and by the *modified simplicial band depth* (mSBD) developped by López-Pintado et al. (2014).

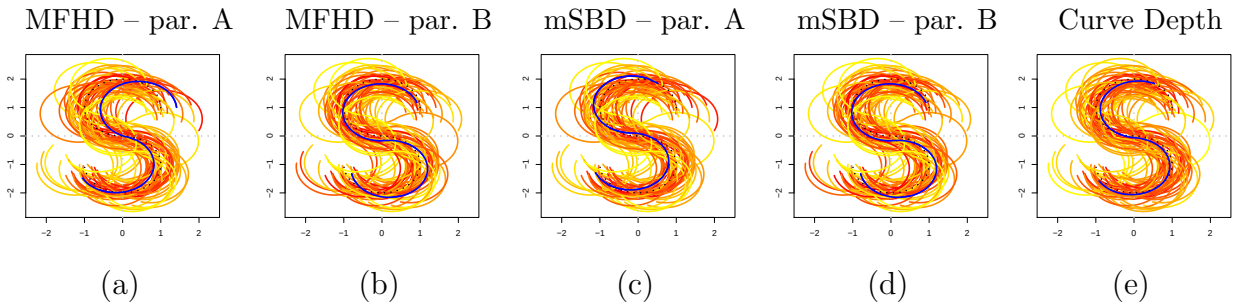


Figure 3.1: *Comparison of depth based ordering for two parametrizations A and B provided respectively by MFHD (a)–(b), mSBD (c)–(d), and by the new depth for unparameterized curves (e). The depth increases from yellow to red. Each deepest curve is plotted in blue. The center of symmetry of the distribution is plotted using black dots. Source: an ensemble of 50 simulated S letters; see Section A.1 in Supplementary Materials of Lafaye De Micheaux et al. (2022).*

In Figure 3.1 (a)-(d), it is seen that the choice of a parametrization (A or B) has a clear impact on which curve is identified as the deepest (in blue). Moreover, unlike MFHD and mSBD, the unparameterized approach finds a deepest curve which is very close to the center of symmetry (the dotted curve). Also, one observes that some curves with high depth (in red) seem to be outliers (Figure 3.1 (a) and (c), upper right) and some curves with low depth (in yellow) are close to the deepest curve (Figure 3.1 (b) and (d)). This problem is even more striking on Figure 3.2. There, many simulated hurricane tracks are identified as outliers (in red, on panels (a)–(d)) by MFHD and mSBD (with two different parametrizations) even if they are close to the center of distribution of the curves (in dark blue). This is in agreement with (Mirzargar et al., 2014, Section 5) who note that "*the time-parameterization is more sensitive to the velocity outlier as a parameterization-dependent feature, the arc-length and life-time percentage parameterization are more sensitive to shape and positional outliers.*" Here again the unparameterized approach correctly identifies outliers (panel (e)).

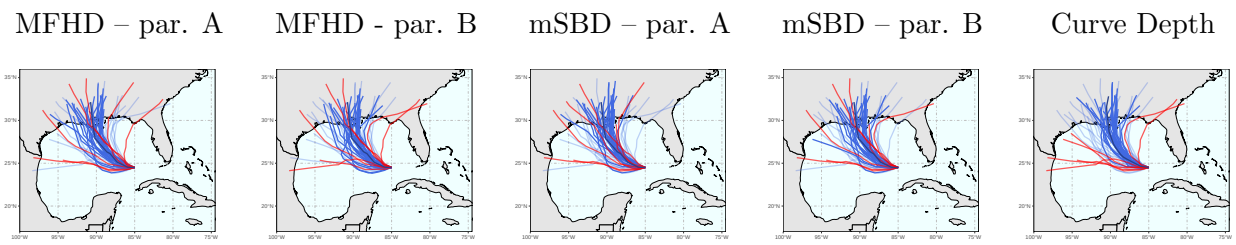


Figure 3.2: *Comparison of the depth based ordering for two parametrizations A (time) and B (arc-length) provided respectively by MFHD (a)–(b), mSBD (c)–(d), and by the new depth for unparameterized curves (e). Curves with low value of depth are plotted in red, the others in blue. Each deepest curve is plotted in dark blue. Source: an ensemble of 50 simulated hurricane tracks (Mirzargar et al., 2014).*

Note that MFHD and mSBD depths are computed by comparing each point on a given curve only to points (from the other curves) that "occur at the same time". Curves are thus compared pointwisely and not globally (this is a direct consequence of parametrization), which is the cause of the aforementioned artefacts.

Of course, depending on the context, working with a proper parametrization of curves can be relevant. For instance, if available, one could use speed of writing as a meaningful parametrization in a handwriting recognition problem; see Section A.2 in Supplementary Materials of Lafaye De Micheaux et al. (2022). For further discussion on the importance and possible choices of a proper parametrization when employing functional data depth, see, *e.g.*, López-Pintado et al. (2014), Mirzargar et al. (2014) and references therein.

In this chapter, the aim is to define a depth which is invariant to the choice of a parametrization of the curves. This was originally motivated by the need to analyze a very large number of bundles of white matter fibers obtained through diffusion tensor imaging (an MRI-based neuroimaging technique) among a population of elderly twins. These neuronal fibers, also called axons, are nerve cell extensions that transmit electrical information between different regions of the brain. The aim of the study was to investigate if genetics plays a role in the spatial organization of these fibers.

With this motivation in mind, a new concept of *depth for curves* is proposed in this chapter, that is invariant to the choice of the parametrization. It will be broadly applicable, thanks to the freely available `R/C++` package `curveDepth` (M. et al., 2019), to many other similar types of data. On can mention a few examples such as textile fibers (Xu et al., 2001), blood clot fibers (Collet et al., 2005), blood vessels centrelines (Sangalli et al., 2009), moving objects such as birds migrating (Su et al., 2014, Yuan et al., 2017), multidimensional data sets obtained by constructing principal curves (Hastie and Stuetzle, 1989). The results of the study with application of the developed depth notion for curves to neuroimaging are provided in Section 3.5, right after the theoretical exposition of the used statistical model (Section 3.3) and the novel depth notion (Section 3.4) with its properties (Section 3.4.2). The reader is further referred to Lafaye De Micheaux et al. (2022) for implementation details

(Section 4), simulated-data experiments (Section 5) and application of the proposed curve depth to classification and clustering (Section 6.2).

3.3 Statistical setting

In what follows, the space of unparameterized curves is introduced and a statistical model on it is defined. For a comprehensive reference the reader is referred to Kemppainen and Smirnov (2017, Section 2) which borrowed material from Aizenman and Burchard (1999, Section 2.1) and Burago et al. (2001, Section 2.5). For additional details see Section B in Supplementary Materials of Lafaye De Micheaux et al. (2022).

3.3.1 The space of unparameterized curves

Let Γ be the set of increasing continuous functions $\gamma : [0,1] \to [0,1]$ such that $\gamma(0) = 0$ and $\gamma(1) = 1$. A *parameterized curve* $\boldsymbol{\beta}$, also called a *path*, is an element of $\mathscr{C}([0,1], \mathbb{R}^d)$. The image of $\boldsymbol{\beta}$, denoted as $S_{\boldsymbol{\beta}} = \boldsymbol{\beta}([0,1])$, is called the *locus* of $\boldsymbol{\beta}$. Informally if $\boldsymbol{\beta}(t)$ describes the position of a moving particle at time t, then $S_{\boldsymbol{\beta}}$ describes the physical route taken by this particle with no consideration being given to stops or goings backward occuring on its trajectory. The function $\boldsymbol{\beta} : [0,1] \mapsto \mathbb{R}^d$, a parametrization of $S_{\boldsymbol{\beta}}$ with parameter t, provides an ordering along $S_{\boldsymbol{\beta}}$. Note that there might exists an infinite number of different parametrizations describing the same locus.

Remark 3.1 *The start point of $S_{\boldsymbol{\beta}}$ is the image of 0 by $\boldsymbol{\beta}$. The end point is the image of 1. The locus of a trivial curve coincides with a singleton, i.e., a single point of $\mathbb{R}^d$.*

Formally, unparameterized curves are usually defined via an equivalence relation on the set of parameterized curves in $\mathbb{R}^d$ up to the set of monotonic functions from $[0,1]$ to $[0,1]$. Roughly speaking, two curves $\boldsymbol{\beta}_1$ and $\boldsymbol{\beta}_2$ are said equivalent if they share the same locus and visit its points continuously and in the same order, possibly at a different speed. Hereafter, let us restrict to the set of all curves equivalent to $\boldsymbol{\beta}$ that start at $\boldsymbol{\beta}(0)$ and stop at $\boldsymbol{\beta}(1)$. More precisely, two parameterized curves $\boldsymbol{\beta}_1$ and $\boldsymbol{\beta}_2$ are equivalent whenever there exist two reparametrizations $\gamma_1, \gamma_2 \in \Gamma$ such that $\boldsymbol{\beta}_1 \circ \gamma_1 = \boldsymbol{\beta}_2 \circ \gamma_2$. Let us then define the *unparameterized curve* $\mathcal{C}_{\boldsymbol{\beta}}$ as the set of all paths equivalent to $\boldsymbol{\beta}$, that is the equivalence class of $\boldsymbol{\beta}$ up to this equivalence relation. Informally, $\mathcal{C}_{\boldsymbol{\beta}}$ describes the trajectory from $\boldsymbol{\beta}(0)$ to $\boldsymbol{\beta}(1)$, with no information about the location at any time. Note that in this context, it would be possible to consider the general definition, *i.e.*, to walk a path $\boldsymbol{\beta}$ from $\boldsymbol{\beta}(1)$ to $\boldsymbol{\beta}(0)$, or the other way around. But restricting all definitions by considering the set of parameterized curves in $\mathbb{R}^d$ only up to the set of reparametrizations Γ greatly simplifies exposition; see Remark B.2 in Supplementary Materials of Lafaye De Micheaux et al. (2022). In the sequel, an unparameterized curve will be generically denoted $\mathcal{C}$. Notice that all parameterized curves in the same equivalence class $\mathcal{C}$ share the same locus, which enables one to talk about the locus of $\mathcal{C}$, denoted thereafter as $S_{\mathcal{C}}$.

The *space of unparameterized curves* is then defined as

$$\mathfrak{C} = \{\mathcal{C}_{\boldsymbol{\beta}} : \boldsymbol{\beta} \in \mathscr{C}([0,1], \mathbb{R}^d)\}.$$

In other words, $\mathfrak{C}$ is the quotient space of $\mathscr{C}([0,1], \mathbb{R}^d)$ by the equivalence relation on the set of parameterized curves

Following Kemppainen and Smirnov (2017), endow the space of curves $\mathfrak{C}$ with the Fréchet metric $d_{\mathfrak{C}}$ defined as

$$d_{\mathfrak{C}}(\mathcal{C}_1, \mathcal{C}_2) = \inf\{\|\boldsymbol{\beta}_1 - \boldsymbol{\beta}_2\|_\infty;\ \boldsymbol{\beta}_1 \in \mathcal{C}_1,\ \boldsymbol{\beta}_2 \in \mathcal{C}_2\}, \quad \mathcal{C}_1, \mathcal{C}_2 \in \mathfrak{C}, \tag{3.8}$$

where $\|\boldsymbol{\beta}\|_\infty = \sup_{t\in[0,1]} |\boldsymbol{\beta}(t)|_2$ for $\boldsymbol{\beta} \in \mathscr{C}([0,1], \mathbb{R}^d)$. The resulting metric space $(\mathfrak{C}, d_{\mathfrak{C}})$ is non linear. It inherits the properties of separability and completeness from $\mathscr{C}([0,1], \mathbb{R}^d)$; see Section B.2 in Supplementary Materials of Lafaye De Micheaux et al. (2022). This guarantees the existence of non-atomic probability measures on $(\mathfrak{C}, d_{\mathfrak{C}})$. Moreover, according to Parthasarathy (1967, Theorems 1.2, 3.2 and 8.1), every probability measure defined on $\mathfrak{C}$ is regular and tight.

3.3.2 The arc-length probability measure of a curve

The length $L(\boldsymbol{\beta})$ of a parameterized curve $\boldsymbol{\beta} \in \mathcal{C}$ is defined as

$$L(\boldsymbol{\beta}) = \sup_{\tau} \{L_\tau(\boldsymbol{\beta});\ \tau \text{ a partition of } [0,1]\}, \tag{3.9}$$

where $L_\tau(\boldsymbol{\beta}) = \sum_{j=1}^{J} |\boldsymbol{\beta}(\tau_j) - \boldsymbol{\beta}(\tau_{j-1})|_2$ is the *chordal length* of $\boldsymbol{\beta}$ associated with the partition $\tau = \{\tau_0, \ldots, \tau_J;\ 0 = \tau_0 < \cdots < \tau_J = 1,\ J \in \mathbb{N}^*\}$. Informally $L(\boldsymbol{\beta})$ is the total distance traveled by a particle moving from $\boldsymbol{\beta}(0)$ to $\boldsymbol{\beta}(1)$ along the support $S_{\boldsymbol{\beta}}$ of the curve $\boldsymbol{\beta}$ (taking into account any backward steps). Then all parameterized curves in $\mathcal{C}$ have the same length. Consequently, the length of $\mathcal{C}$, denoted $L(\mathcal{C})$, is defined by $L(\mathcal{C}) = L(\boldsymbol{\beta})$, for any $\boldsymbol{\beta} \in \mathcal{C}$. Note that the function $L : \mathfrak{C} \to [0, +\infty]$ is not continuous, but it is measurable (Lemma B.3 in Supplementary Materials of Lafaye De Micheaux et al., 2022). In the following assume that all unparameterized curves belong to the measurable set $\mathfrak{C}_L = \{\mathcal{C} \in \mathfrak{C};\ 0 < L(\mathcal{C}) < \infty\} \subset \mathfrak{C}$, the subset of *rectifiable* (*i.e.*, of finite length) unparameterized curves with a positive length.

According to Väisälä (2006, Theorem 2.4), each curve $\mathcal{C} \in \mathfrak{C}_L$ contains a unique parametrization $\boldsymbol{\beta}_{\mathcal{C}} : [0,1] \to \mathbb{R}^d$, called *the arc-length parametrization*, whose restrictions to the intervals $[0,t]$, noted $\boldsymbol{\beta}_{\mathcal{C}}^t$, satisfy $L(\boldsymbol{\beta}_{\mathcal{C}}^t) = tL(\mathcal{C})$, for all $t \in [0,1]$. Informally, with $\boldsymbol{\beta}_{\mathcal{C}}$, the locus $S_{\mathcal{C}}$ is visited at a constant speed. Then any rectifiable curve $\mathcal{C}$ may be expressed as

$$\mathcal{C} = \{\boldsymbol{\beta}_{\mathcal{C}} \circ \gamma;\ \gamma \in \Gamma\}.$$

Using the arc-length parametrization $\boldsymbol{\beta}_{\mathcal{C}}$ of an unparameterized curve $\mathcal{C}$, one can thus define the *line integral* of a non-negative Borel function $f : \mathbb{R}^d \to \mathbb{R}$ over $\mathcal{C}$ as

$$\int_{\mathcal{C}} f(\boldsymbol{s}) d\boldsymbol{s} := \int_0^1 f\left(\boldsymbol{\beta}_{\mathcal{C}}(t)\right) L(\mathcal{C}) dt, \tag{3.10}$$

where the integral on the right is a Riemann integral. Furthermore, define the *arc-length probability measure* of $\mathcal{C}$ as the probability distribution $\mu_{\mathcal{C}}$ on the Borel sets of $\mathbb{R}^d$:

$$\text{for any borel set } A \text{ of } \mathbb{R}^d, \qquad \mu_{\mathcal{C}}(A) = \frac{1}{L(\mathcal{C})} \int_{\mathcal{C}} \mathbb{1}_A(\boldsymbol{s}) d\boldsymbol{s}\,, \tag{3.11}$$

where the indicator function $\mathbb{1}_A(\boldsymbol{x})$ takes the value 1 if $\boldsymbol{x} \in A$ and 0 otherwise.

From (3.10) and (3.11), it immediately follows

$$\int_{\mathcal{C}} f(\boldsymbol{s}) d\mu_{\mathcal{C}}(\boldsymbol{s}) = \int_0^1 f(\boldsymbol{\beta}_{\mathcal{C}}(t)) dt. \tag{3.12}$$

Also, note that $\mu_{\mathcal{C}}$ only contains information about the support $S_{\mathcal{C}}$ of $\mathcal{C}$ and the frequency at which its points are visited. Roughly speaking, $\mu_{\mathcal{C}}(A)$ can be interpreted as a ratio: the distance travelled by a particle on the subset $S_{\mathcal{C}} \cap A$ divided by the total distance it travels on $S_{\mathcal{C}}$. (Note that $L(\mathcal{C})$ can be different from the length of $S_{\mathcal{C}}$.) It is somehow a normalised measure of how much of curve $\mathcal{C}$ intersects with A.

3.3.3 Describing a sample of curves

Denote by $\mathcal{P}$ the set of all probability measures defined on the Borel σ-algebra of the Borel sets of $(\mathfrak{C}, d_{\mathfrak{C}})$ whose support is a subset of rectifiable curves of positive length (to exclude singletons):

$$\mathcal{P} = \Big\{ P, \text{ a probability measure on } (\mathfrak{C}, d_{\mathfrak{C}}) \ ; \ P\left(\mathfrak{C}_L\right) = 1 \Big\}.$$

Consider a random unparameterized curve $\mathcal{X}$, namely a random element taking "values" in the space of unparameterized curves $\mathfrak{C}$, whose probability distribution $P \in \mathcal{P}$ is unknown. Define the probability distribution Q_P as follows:

$$\text{for all borel sets } A \text{ of } \mathbb{R}^d, \qquad Q_P(A) = \int_{\mathfrak{C}} \mu_{\mathcal{C}}(A) dP(\mathcal{C}) = \mathbb{E}_P[\mu_{\mathcal{X}}(A)], \tag{3.13}$$

a measure of how much (on average) a curve generated by $\mathcal{X}$ intersect with A.

Remark 3.2 *In Section B.3 in Supplementary Materials of Lafaye De Micheaux et al. (2022), it is shown that for any Borel bounded function $f : \mathbb{R}^d \to \mathbb{R}$, the function $\mathcal{C} \in \mathfrak{C}_L \mapsto \int_{\mathcal{C}} f d\mu_{\mathcal{C}} \in \mathbb{R}$ is measurable. Consequently, Q_P is well-defined.*

The statistical model considered in this chapter is to assume that the data to be observed are n random unparameterized curves $\mathcal{X}_1, \ldots, \mathcal{X}_n$, which are independent copies of the random element $\mathcal{X}$, that is to say

$$\mathcal{X}_1, \ldots, \mathcal{X}_n \text{ are i.i.d. from } P \in \mathcal{P}. \tag{3.14}$$

In the next section, a population data depth for unparameterized curves is defined, as well as its sample version.

3.4 Data depth for unparametrized curves

For a pair $(\boldsymbol{u}, \boldsymbol{x}) \in \mathbb{S}^{d-1} \times \mathbb{R}^d$, let $H_{\boldsymbol{u},\boldsymbol{x}}$ denote the closed halfspace $\{\boldsymbol{y} \in \mathbb{R}^d : \boldsymbol{y}^\top \boldsymbol{u} \geq \boldsymbol{x}^\top \boldsymbol{u}\}$ whose frontier is orthogonal to the vector u and goes through the point x. Notice that if $d = 1$, the unit-sphere is $\{-1, 1\}$.

3.4.1 Population and sample versions

Definition 3.7 (Curve depth, population version) *(Lafaye De Micheaux et al., 2022) Let $\mathcal{C} \in \mathfrak{C}_L$ be an unparameterized curve and let $P \in \mathcal{P}$ be a probability measure. Define the* curve depth *of $\mathcal{C}$ w.r.t. P, denoted $D(\mathcal{C}|P)$, by the mapping*

$$\begin{aligned} D: \quad \mathfrak{C}_L \times \mathcal{P} &\to \mathbb{R} \\ (\mathcal{C}, P) &\mapsto D(\mathcal{C}|P) = \int_{\mathcal{C}} D(\boldsymbol{s}|Q_P, \mu_{\mathcal{C}}) d\mu_{\mathcal{C}}(\boldsymbol{s}), \end{aligned} \tag{3.15}$$

where the above line integral is computed via (3.10) using, for any $d \geq 1$ and any $\boldsymbol{x} \in S_{\mathcal{C}}$,

$$D(\boldsymbol{x}|Q_P, \mu_{\mathcal{C}}) = \inf_{\boldsymbol{u} \in \mathbb{S}^{d-1}} \frac{Q_P(H_{\boldsymbol{u},\boldsymbol{x}})}{\mu_{\mathcal{C}}(H_{\boldsymbol{u},\boldsymbol{x}})}, \tag{3.16}$$

with the convention that $a/0 = +\infty$ for all $a > 0$ and $0/0 = 0$ in the above ratio.

The term $D(\boldsymbol{x}|Q_P, \mu_{\mathcal{C}})$ aims to compare the two distributions Q_P and $\mu_{\mathcal{C}}$ around $\boldsymbol{x} \in S_{\mathcal{C}}$. For $\boldsymbol{u}$ and $\boldsymbol{x}$ fixed, recall from (3.11) and from (3.13) that $\mu_{\mathcal{C}}(H_{\boldsymbol{u},\boldsymbol{x}})$ measures (the fraction of length of) how much the curve $\mathcal{C}$ delves into the halfspace $H_{\boldsymbol{u},\boldsymbol{x}}$, whereas $Q_P(H_{\boldsymbol{u},\boldsymbol{x}})$ measures (the expected fraction of length of) how much a random curve $\mathcal{X}$ (with distribution P) delves into $H_{\boldsymbol{u},\boldsymbol{x}}$. Consequently, the ratio $Q_P(H_{\boldsymbol{u},\boldsymbol{x}})/\mu_{\mathcal{C}}(H_{\boldsymbol{u},\boldsymbol{x}})$ is small when curves generated according to P enter less into $H_{\boldsymbol{u},\boldsymbol{x}}$ than the curve $\mathcal{C}$. Getting a value $r > 1$ (resp. $r < 1$) for this ratio, indicates that $\mathcal{X}$ generates curves that enter into $H_{\boldsymbol{u},\boldsymbol{x}}$, on average, r times more (resp. $1/r$ times less) than $\mathcal{C}$ does; see Figure 3.3 for a visual aid.

Then, similarly to the original halfspace depth, to obtain $D(\boldsymbol{x}|Q_P, \mu_{\mathcal{C}})$, consider all possible rotations of the halfspace $H_{\boldsymbol{u},\boldsymbol{x}}$ around $\boldsymbol{x}$ to find the one that discriminates the most the curve $\mathcal{C}$ from a curve generated according to P. Let us call $D(\boldsymbol{x}|Q_P, \mu_{\mathcal{C}})$ as the *point curve depth* at $\boldsymbol{x} \in S_{\mathcal{C}}$. Then (3.15) defines the depth of $\mathcal{C}$ w.r.t. P as the mean of the point curve depths at all $\boldsymbol{x}$ in its locus $S_{\mathcal{C}}$.

Notice that if there exists $\boldsymbol{u} \in \mathbb{S}^{d-1}$ such that $Q_P(H_{\boldsymbol{u},\boldsymbol{x}}) = 0$, then $\boldsymbol{x}$ is an outlier w.r.t. Q_P, and thus the contribution of $\boldsymbol{x} \in S_{\mathcal{C}}$ to the depth of $\mathcal{C}$ w.r.t. P is set to zero, that is $D(\boldsymbol{x}|Q_P, \mu_{\mathcal{C}}) = 0$.

If $Q_P(H_{\boldsymbol{u},\boldsymbol{x}}) > 0$ for all $\boldsymbol{u} \in \mathbb{S}^{d-1}$, that means x lies in the convex hull of the support of Q_P. The aim is to calculate the depth of $\boldsymbol{x} \in S_{\mathcal{C}}$ w.r.t. Q_P relatively to the measure $\mu_{\mathcal{C}}$, that is why one considers the ratio $Q_P(H_{\boldsymbol{u},\boldsymbol{x}})/\mu_{\mathcal{C}}(H_{\boldsymbol{u},\boldsymbol{x}})$ in the definition of $D(\boldsymbol{x}|Q_P, \mu_{\mathcal{C}})$. In this case, one can show that there exists $\boldsymbol{u}$ such that $\mu_{\mathcal{C}}(H_{\boldsymbol{u},\boldsymbol{x}}) \geq Q_P(H_{\boldsymbol{u},\boldsymbol{x}}) > 0$ (Lemma C.1 in Supplementary Materials of Lafaye De Micheaux et al., 2022), so that $\boldsymbol{x} \mapsto D(\boldsymbol{x}|Q_P, \mu_{\mathcal{C}})$ is bounded by 1. Moreover, $\boldsymbol{x} \mapsto D(\boldsymbol{x}|Q_P, \mu_{\mathcal{C}})$ is measurable as a limit of measurable functions (see Lemma C.4 in Supplementary Materials of Lafaye De Micheaux et al., 2022).

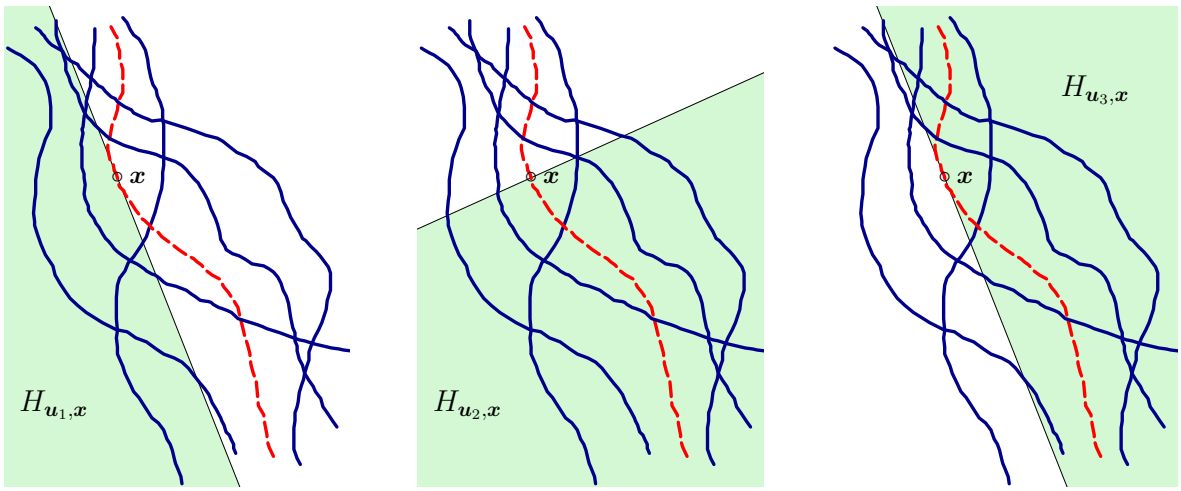


Figure 3.3: *Illustrations of the statistical model and depth calculation (3.16) for three halfspaces with a sample of five curves generated by $\mathcal{X}$ in blue and the curve $\mathcal{C}$ in red. Consider all halfspaces whose frontier contains the point $\boldsymbol{x}$ and pick up the smallest ratio of the probability measures between Q_P and $\mu_{\mathcal{C}}$: (left) $\frac{Q_P(H_{\boldsymbol{u}_1,\boldsymbol{x}})}{\mu_{\mathcal{C}}(H_{\boldsymbol{u}_1,\boldsymbol{x}})} = 9.714$, (middle) $\frac{Q_P(H_{\boldsymbol{u}_2,\boldsymbol{x}})}{\mu_{\mathcal{C}}(H_{\boldsymbol{u}_2,\boldsymbol{x}})} = 0.999$, (right) $\frac{Q_P(H_{\boldsymbol{u}_3,\boldsymbol{x}})}{\mu_{\mathcal{C}}(H_{\boldsymbol{u}_3,\boldsymbol{x}})} = 0.618$*

Definition 3.8 (Curve depth, sample version) *(Lafaye De Micheaux et al., 2022) Let $\mathcal{X}_1, \ldots, \mathcal{X}_n$ be a random sample of unparameterized curves belonging to $\mathfrak{C}_L$ a.s. and let $\mathcal{C} \in \mathfrak{C}_L$ be a rectifiable unparameterized curve. With a slight abuse of notation, and thanks to (3.10), define the* curve depth *of $\mathcal{C}$ w.r.t. $\mathcal{X}_1, \ldots, \mathcal{X}_n$ by the mapping*

$$\begin{aligned} D: \quad \mathfrak{C}_L \times \{\mathfrak{C}_L\}^n \quad &\to \mathbb{R} \\ (\mathcal{C}, \mathcal{X}_1, \ldots, \mathcal{X}_n) \quad &\mapsto D(\mathcal{C}|\mathcal{X}_1, \ldots, \mathcal{X}_n) = \int_{\mathcal{C}} D(\boldsymbol{s}|Q_n, \mu_{\mathcal{C}}) d\mu_{\mathcal{C}}(\boldsymbol{s}), \end{aligned} \tag{3.17}$$

where $Q_n = (\mu_{\mathcal{X}_1} + \cdots + \mu_{\mathcal{X}_n})/n$ and $\boldsymbol{\beta}_{\mathcal{C}}$ is the arc-length parametrization of $\mathcal{C}$.

Remark 3.3 *In a sense, the proposed depth may be seen as a generalization of the Tukey's halfspace depth in $\mathbb{R}^d$. If $\mathcal{C}$ is a trivial curve, that is $L(\mathcal{C}) = 0$ and $S_{\mathcal{C}} = \{\boldsymbol{y}\}$ for some $\boldsymbol{y} \in \mathbb{R}^d$, define $\mu_{\mathcal{C}}$ as the Dirac measure $\delta_{\boldsymbol{y}}$ at $\boldsymbol{y}$. Then, if $\mathcal{X}_1, \ldots, \mathcal{X}_n$ are also trivial curves, that is $S_{\mathcal{X}_i} = \{\boldsymbol{x}_i\}$, $i = 1, \ldots, n$, one gets*

$$\begin{aligned} D(\mathcal{C}|\mathcal{X}_1, \ldots, \mathcal{X}_n) &= D(\boldsymbol{y}|Q_n, \delta_{\boldsymbol{y}}) \\ &= \inf_{\boldsymbol{u} \in \mathbb{S}^{d-1}} \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}_{\boldsymbol{x}_i \in H_{\boldsymbol{u},\boldsymbol{y}}}. \end{aligned}$$

Theorem 3.1 below states that the sample version of the curve depth (3.17) converges in probability to the population version (3.15) as $n \to \infty$.

Theorem 3.1 *(Lafaye De Micheaux et al., 2022) Let $\mathcal{C} \in \mathfrak{C}_L$ be an unparameterized curve such that $\mu_{\mathcal{C}}$ is non-atomic. Let P be a probability measure in the space of unparameterized curves such that $P \in \mathcal{P}$ and Q_P is non-atomic. Then the sample curve depth $D(\mathcal{C}|\mathcal{X}_1, \ldots, \mathcal{X}_n)$ converges in probability to $D(\mathcal{C}|P)$ as $n \to \infty$.*

3.4.2 Properties

The main aim of the proposed curve depth is to provide a meaningful statistical ordering of the observed curve data, which is experimentally studied and illustrated on real-data examples in Section 3.5. Theorem 3.1 states the consistency of the sample curve depth under mild assumptions and in this section its properties are discussed.

Following the suggestion of Liu (1990) for simplicial depths, Zuo and Serfling (2000) have defined four properties to be satisfied by a proper multivariate depth function: affine invariance, maximality at the center of symmetry, monotonicity relative to the deepest point and vanishing at infinity. (For a slightly different version of the postulates see also Dyckerhoff (2004) and Mosler (2013).) For a functional depth, Nieto-Reyes and Battey (2016) suggest that six properties need to be satisfied, but Gijbels and Nagy (2017) argue that some of them could be demanding.

The situation appears to be even more challenging for the space of unparameterized curves. Indeed, (loci of) unparameterized curves can be seen as subsets of $\mathbb{R}^d$ which are parameterized by paths up to the same order of visit of their points. These mathematical objects can thus be thought of as being "between" functional data and set data. Moreover, since no canonical mandatory postulates for a functional depth have been established yet, and since the existing postulates are mainly inherited from those for the multivariate depth function, the following analysis is based on the latter.

Since the length is an important characteristic of an unparameterized curve, similarity invariance, which is associated with a similarity group preserving orientation and ratio of lengths, seems to be more appropriate than affine invariance in this context. Moreover, the space of unparameterized curves is not a vector space. For instance the surjection $\boldsymbol{\beta} \mapsto \mathcal{C}_{\boldsymbol{\beta}}$ is not linear (there is no natural way to define the addition of two unparameterized curves and thus no line segment between two unparameterized curves, a crucial point for the monotonicity property). It is thus not possible to extend the classical formulation of a depth using results from Dutta et al. (2011) or Mosler and Polyakova (2018), say. Similarly, there is no universal way to define a notion of symmetry for unparameterized curves, no symmetry center can be defined either. The vanishing at infinity property can be directly extended to the space of curves. Below let us state the properties satisfied by the curve depth function and summarize them in Theorem 3.2.

Boundness. Calculating the curve depth (3.15) consists in integrating a non-negative function bounded by one w.r.t. a probability measure. This fulfills one of the basic requirements of a depth function: to take values on the unit interval.

Similarity invariance. For a multivariate depth, affine invariance is required for changelessness w.r.t. an affine change of the coordinate system. For the space of unparameterized curves, consider affine transformations that also preserve ratios of the lengths of curves, *i.e.*, similarities. (Note that the length of an unparameterized curve is a property of the equivalence class.) A similarity $f : \mathbb{R}^d \to \mathbb{R}^d$ is an affine transform, $f(\boldsymbol{x}) = r\boldsymbol{A}\boldsymbol{x} + \boldsymbol{b}$ such

that $\boldsymbol{A}$ is an orthogonal matrix, r is a positive factor and $\boldsymbol{b} \in \mathbb{R}^d$ is a vector. In particular, for all $\boldsymbol{x}$ and $\boldsymbol{y}$ in $\mathbb{R}^d$, one has $|f(\boldsymbol{x}) - f(\boldsymbol{y})|_2 = r|\boldsymbol{x} - \boldsymbol{y}|_2$. Denote by P_f the distribution of the image under f of a stochastic process having a distribution P. A map D satisfies the property of similarity invariance if for every rectifiable curve $\mathcal{C}$ and every similarity map $f : \mathbb{R}^d \to \mathbb{R}^d$, it holds $D(\mathcal{C}, P) = D(f \circ \mathcal{C}, P_f)$.

Vanishing at infinity. The farther away an unparametrized curve is from a data cloud of curves, the smaller its depth should be. To formulate the vanishing at infinity property of the curve depth$D$, consider any sequence $(\mathcal{C}_n)_n$ of curves in $\mathfrak{C}_L$ such that $\mu_{\mathcal{C}_n}$ is a non-atomic measure for all n and $\lim_{n\to\infty} d_{\mathfrak{C}}(\mathcal{C}_n, 0) = \infty$, where 0 denotes the set of parametrized curves equivalent to the constant curve $t \mapsto \boldsymbol{\beta}(t) = 0$ for all $t \in [0, 1]$. However, such a formulation involves sequences of curves whose length tends to infinity. To exclude these cases, assume that there exists some $\ell > 0$ such that $L(\mathcal{C}_n) < \ell$ for all n. This guarantees that only the location of these curves tends to infinity. Then one can prove that

$$\lim_{n\to\infty} D(\mathcal{C}_n, P) = 0.$$

Theorem 3.2 *(Lafaye De Micheaux et al., 2022) Under the assumptions of Theorem 3.1, the curve depth is a depth function in $\mathfrak{C}_L$, i.e., it takes values in $[0, 1]$, is similarity-invariant and is vanishing at infinity.*

3.5 Real-data illustration on brain imaging

Numerical computation of the newly proposed curve depth and of the above-mentioned distance are implemented in the `R` package `CurveDepth` (M. et al., 2019) which is available on the CRAN (R Core Team, 2022).

3.5.1 Application to the Older Australian Twins Study data

White matter (WM) in the brain is made up of long myelinated axonal fibers generally regarded as passive routes connecting several gray matter regions (the ones containing neurons) to permit flow of information across them. In such tissue, water tends to diffuse mostly along the direction of the fibers. The ratio of axial and radial movement is called fractional anisotropy. Diffusion Tensor Magnetic Resonance Imaging (DTI) measures the motion of hydrogen atoms within water in all three dimensions.

Here the use is made of access to DTI scans from the Older Australian Twins Study (OATS), an ongoing longitudinal study investigating genetic and environmental factors and their associations and interactions in healthy brain ageing and ageing-related neurocognitive disorders for people aged 65+ years (Sachdev et al., 2009).

The DTI data considered in the current section were drawn from 34 twin pairs, aged between 67.3 and 84.2 years. Eleven of the 34 pairs were dizygotic (DZ) twin pairs (*i.e.*, non-identical twins sharing 50% of their genes) and 23 monozygotic (MZ) twin pairs (*i.e.*,

identical twins sharing 100% of their genes). Using `MRtrix` software (Tournier et al., 2012) to extract corticospinal fiber tracts from the DTI scans (an operation called tractography), the resulting data sets were two bundles of around 1,000 fibers each per subject (see Figure 3.4; left). Other pre-processing steps are described in Supplementary Materials of Lafaye De Micheaux et al. (2022), Section H.

It is quite a challenging task to visualize brain fibers. Consequently this information is difficult to use in a clinical environment (*e.g.*, for surgery planning). New tools are thus needed for efficiently representing these *tractograms*. An interesting approach by Mercier et al. (2018) consists in progressively simplifying tractograms by grouping similar fibers into a specific geometric representation.

The depth for curves developed here can also help neuroscientists to visualize a 3D bundle of fibers. One can follow the approach adopted by Mercier et al. (2018) by grouping curves according to their depths. It is also possible to assign a transparency value to each curve equal or proportional to its depth value (see Figure 3.4; left) to inspect the whole bundle at once. Similarly, one can instead assign a low transparency value to the least deep curves in order to visualize outliers (see Figure 3.4; right). Outliers can eventually be removed before further statistical analyses are conducted.

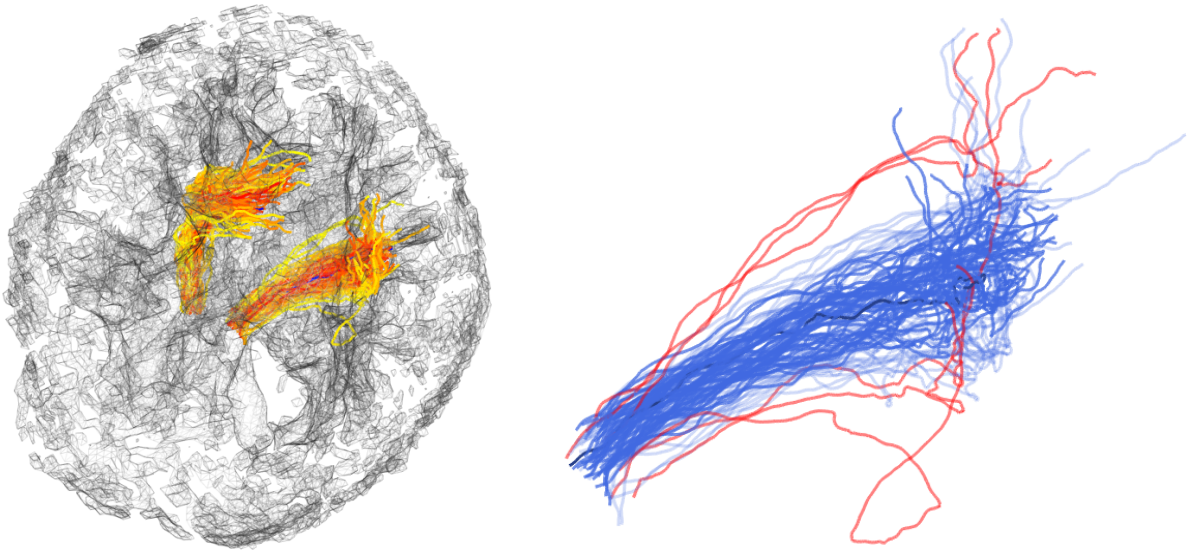


Figure 3.4: *Illustrations of the ordering of the white matter fibers for one subject using the curve depth. (Left) Whole brain fiber data set for one twin; see* `http://biostatisticien.eu/DataDepthFig8` *for an interactive 3D applet. (Right) Result of bundle ordering for the right side of the brain only. Only the first* 100 *fibers in the data set are displayed, among which 6 are identified as outliers and colored in red (their depth is less than 0.075).*

It is demonstrated on Figure 3.5 that the curve depth approach gives better results in terms of outlier detection than four other existing depth measures that can be applied to three-dimensional curves. These multivariate functional depth-based competitors (with an arc-length parametrization) are the *modified multivariate band depth* (mMBD), *simplicial*

band depth (SBD), mSBD, *skew-adjusted functional projection depth* (saPRJ), and the curve depth. Observe that the 15 fibers having the lowest depth as computed by the curve depth are located outside of the bundle, while there are fibers with a low depth value inside this bundle for the competitors. Furthermore the range of depth values associated to the curve depth is the widest among the 5 methods considered here. And there is a clearer separation between the depths of outliers and the other fibers. Notice that it is hard to distinguish outliers using SBD and mSBD and that the bottom fiber which is clearly outside the bundle is not detected as an outlier by SBD. Finally, mMBD and saPRJ detect fewer outliers than the curve depth, some of them being the same as the ones detected by the proposed approach.

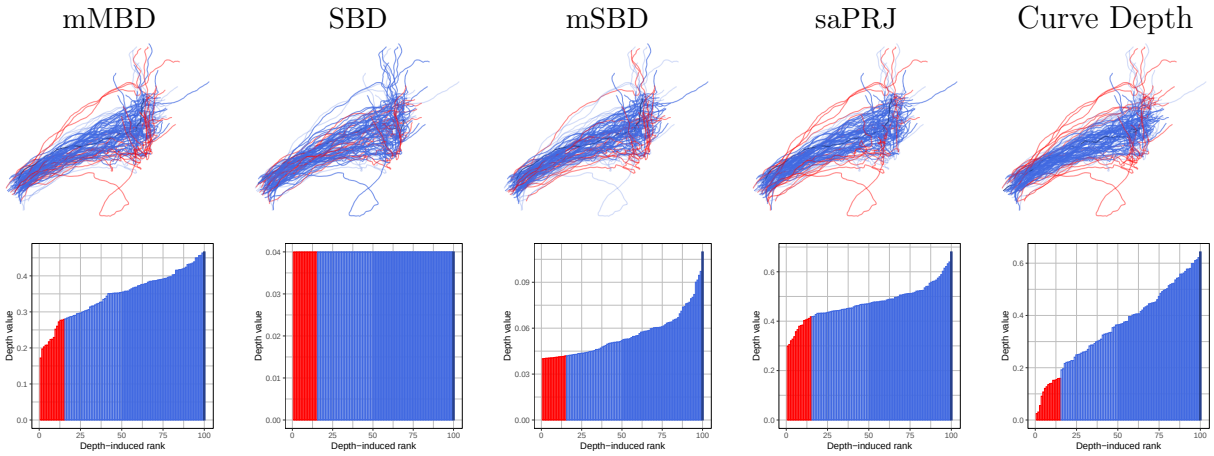


Figure 3.5: *Top: Curve boxplots for a sample of* 100 *WM-fibers from the right side of the brain. The set of* 100 *curves is partitioned into* 15/35/49/1 *curves:* 15 *with the smallest value of depth, considered as outliers (red),* 35 *with a larger value of depth, considered as outer curves (light blue),* 49 *with the largest value of depth, considered as the more central curves (blue), and finally the deepest curve of all (dark blue). Depth methods are (from left to right) mMBD, SBD, mSBD, saPRJ, and the* curve depth. *Bottom: Corresponding depth-ranked histograms.*

3.5.2 Curve registration

Image registration is one of the main pre-processing steps in any statistical analysis of brain imaging data. Its aim is to geometrically match up image volumes of brain structures, for example for structure localization or difference detection. Broadly, this consists in finding rotation and translation parameters that will minimize a certain cost function (*e.g.*, least squares or mutual information) which quantifies how well aligned two images are. Image registration is an active field of research since existing algorithms still have defects, for example they might suffer from directionality bias (Modat et al., 2014). All standard libraries dedicated to the analysis of fMRI, MRI and DTI brain imaging data contain an image registration procedure; see, *e.g.*, `RNiftyReg` (Clayden et al., 2020) in the R software (R Core Team, 2022).

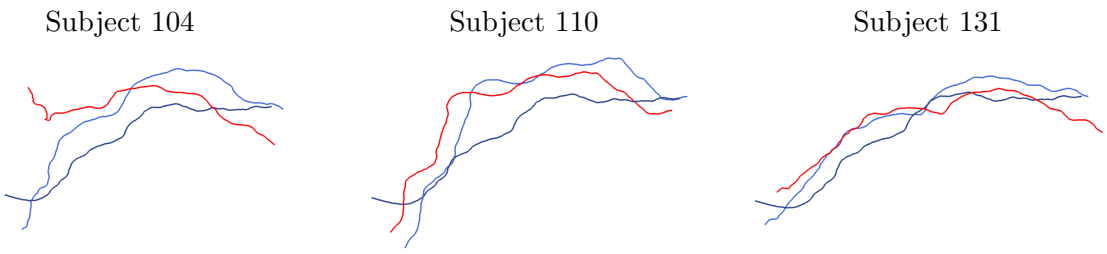


Figure 3.6: *Illustration of the registration process. Subject 235 is the reference subject (i.e., the subject whose deepest curve is D, the* deepest of all*). The red and the dark blue curves are the deepest curves (before registration) of the given subject and of subject* 235*, respectively. The red curve are brought as close as possible, in terms of distance* (3.8)*, to the dark blue curve. The transformed curve (i.e, after registration) is the light blue curve. Distances from each red curve (i.e., before registration) and from each light blue curve (i.e., after registration) to the deepest of all are* 10.271 *and* 3.245 *(for subject* 104*),* 4.539 *and* 3.395 *(for subject* 110*), and* 3.329 *and* 2.084 *(for subject* 131*), respectively.*

Here, the proposed approach to register the bundles at hand is to first extract one single best representative curve for each bundle (namely the deepest one; see the dark blue fiber in Figure 3.4) and then to match these representatives as best as possible. In the twin DTI data set considered here, the aim was to register 68 bundles, of about 1,000 fibers each, located in the left hemisphere say. To reach this goal, first the deepest fiber within each bundle was computed, noted thereafter d_j, $j = 1, \ldots, 68$. Then the deepest fiber among $d_1, ..., d_{68}$ was computed, which is denoted D. Finally, for each bundle j, the rigid transformation was found (in terms of rotation, translation and centering) that minimizes the distance (3.8) between the curves d_j and D. Registration is then achieved by applying each one of these rigid transformations to all the fibers within the corresponding bundle. This process is illustrated in Figure 3.6.

3.5.3 A statistical comparison between MZ and DZ twins

After having performed curve registration, comparison of the empirical distributions is possible. Given two distributions $P_0, P_1 \in \mathcal{P}$ on the space of curves $\mathfrak{C}$, consider the mapping that yields the DD-plot (Liu et al., 1999):

$$\mathfrak{C} \to [0,1]^2 \,,\, \mathcal{C} \mapsto (D(\mathcal{C}|P_0), D(\mathcal{C}|P_1)) \,. \tag{3.18}$$

For two random samples of curves $\{\mathcal{X}_1^{(0)}, \ldots, \mathcal{X}_{n_0}^{(0)}\}$ and $\{\mathcal{X}_1^{(1)}, \ldots, \mathcal{X}_{n_1}^{(1)}\}$ from P_0 and P_1 respectively, the empirical DD-plot can be constructed as:

$$\bigcup_{k=0,1} \left\{ \left(D(\mathcal{X}_i^{(k)}|\mathcal{X}_1^{(0)}, \ldots, \mathcal{X}_{n_0}^{(0)}), D(\mathcal{X}_i^{(k)}|\mathcal{X}_1^{(1)}, \ldots, \mathcal{X}_{n_1}^{(1)}) \right) , i = 1, \ldots, n_k \right\} .$$

For six pairs of twins, DD-plots are presented in Figure 3.7, whose contribution is twofold. First, as a proof of concept, the empirical distributions of two MZ twins are very

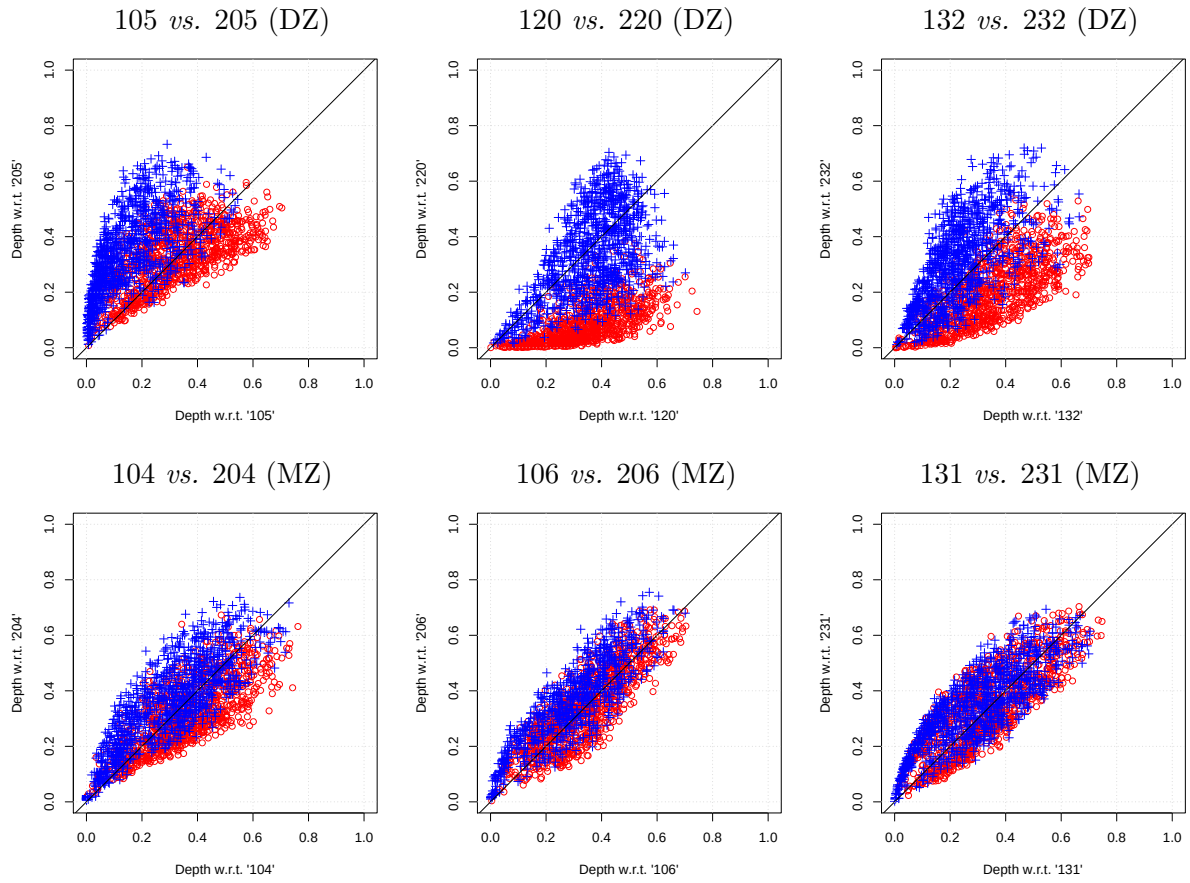


Figure 3.7: *DD-plots of six pairs of twins (red circles for 1xx; blue "+" signs for 2xx) with associated p-values in parenthesis. (Top) three DZ, namely:* 105 *and* 205 *(*$1 - e9$*),* 120 *and* 220 *(*0.017*),* 132 *and* 232 *(*0.003*). (Bottom) three MZ: namely* 104 *and* 204 *(*0.733*),* 106 *and* 206 *(*0.366*),* 131 *and* 231 *(*0.366*).*

similar since the points are concentrated around the diagonal of the DD-plot while those of DZ twins differ (see also Liu et al., 1999). Second, this closeness of the MZ twins underlines the high quality of the curve registration using the the geometrical matching (Section 3.5.2) in the sense that (each of) these two bundles of curves are meant to substantially coincide.

Recently, neuroscientists have discovered that several structures in the brain are influenced by our genetics; see, *e.g.*, (Wen et al., 2016). This suggests a genetically-driven spatial organisation of corticospinal brain fibers. This biological hypothesis can be statistically confirmed by applying the depth-based Wilcoxon testing procedure introduced by Liu and Singh (1993) and further described in (López-Pintado and Romo, 2009). For each pair of twins, 500 fibers were considered selected at random from the first twin as a reference sample. Then 50 fibers from each twin were used (selected at random among the remaining fibers) to calculate the test statistic value. The p-values, computed using the normal asymptotic null distribution given by Lehmann and D'Abrera (1975), are provided in Figure 3.7. They are small for DZ twins and large for MZ twins, a statistical evidence in favor of this biological hypothesis.

Chapter 4

Functional anomaly detection

Functional anomaly detection aims at identifying the curves that significantly differ from the others among the data set available. Given the richness of spaces of functions, the major difficulty lies in the huge diversity in the nature of the observed differences, which may not only depend on the locations of the curves. Following in the footsteps of Hubert et al. (2015), one may distinguish between three types of anomalies: *shift* (the observed curve has the same shape as the majority of the sample except that it is shifted away), *amplitude* or *shape* anomalies. All these three types of anomalies can be *isolated/transient* or *persistent*, depending on their duration with respect to that of the observations. One may easily admit that certain types of anomalies are harder to detect than others: for instance, an isolated anomaly in shape compared to an isolated anomaly in amplitude.

In the context of anomaly detection, work with functional data presents two main difficulties. First, functional spaces are very rich and are difficult to handle, and in particular to identify types of anomalies not present in the training data set. The second difficulty is that the practitioner only has access to partial observations of data. A preliminary step of basis expansion is often used to represent data recorded at discrete times as a continuous function and to reduce noise in data. Even more, basis expansion allows registration into a common argument-scale if observation arguments are not the same across records. Although functional data analysis has been the subject of much attention in recent years, very few generic and flexible methods tailored to functional anomaly detection are documented in the machine-learning literature except for specific types of anomalies.

Most popular approach, usually referred to as *filtering*, consists in bringing the anomaly detection problem to the multivariate case by means of an adequate projection using functional principal component analysis (FPCA) (Ramsay and Silverman, 2005) or a preliminarily selected basis of the function space considered (*e.g.* Fourier, wavelets) and apply next an anomaly detection algorithm designed for the finite-dimensional setup to the resulting representation (for an overview of such methods, see Chandola et al., 2009). This method has an obvious drawback: In FPCA, estimation of the Kahrunen-Loève basis can be very challenging and lead to loose approximations, jeopardizing next the anomaly detection stage, while the *a priori* representation offered by the "atoms" of a predefined basis or frame may unsuccessfully capture the patterns carrying the relevant information to dis-

tinguish abnormal curves from the others. Another approach is based on the notion of *minimum volume sets* (MV-sets in shortened version), originally introduced in Einmahl and Mason (1992) and that generalizes the concept of quantiles to multivariate distributions and offers a nice nonparametric framework for anomaly detection in finite dimension, see Scott and Nowak (2006)'s work. Given the fact that no analogue of Lebesgue measure on an infinite-dimensional Banach space exists and since, considering a law λ of reference (*e.g.* the Wiener or a Poisson measure) on the function space $\mathcal{H}$ of interest, the volume $\lambda(C)$ of a measurable subset $C \subset \mathcal{H}$ can be hardly computed in general, it is far from straightforward to extend MV-set estimation to the functional setup.

The angle embraced in this chapter is quite different. The direct approach promoted here is free from any preliminary representation stage and can be straightforwardly applied to a functional data set. Precisely, in the subsequent Section 4.1, it is proposed to extend the isolation forest algorithm (Liu et al., 2008) to the functional data framework, in a very flexible way, so as to deal with a wide variety of abnormal shapes.

In statistics, although its applications are by no means restricted to anomaly detection, the concept of *functional data depth* that allows to define a notion of centrality in the path space and a center-outward ordering of the curves of the functional data set, see, *e.g.*, (Cuevas et al., 2007, Claeskens et al., 2014, Hubert et al., 2015), has been used for this purpose. However, since the vast majority of functional depth functions introduced only describe the relative location properties of the sample curves, they generally fail to detect other types of anomalies. In Section 4.2, a novel notion of functional depth is proposed, which proves efficient for a wide spectrum of anomalies and particularly the isolated ones.

4.1 Functional isolation forest

In this section, the functional anomaly detection algorithm is proposed. First, in Section 4.1.1, preliminary materials regarding multivariate anomaly detection, which are important for presentation of the further material, are recalled. Then, in Section 4.1.2, the functional isolation forest algorithm is presented and questions of parameter tuning are discussed. Further, Section 4.1.3 gives a brief synopsis of the simulation study. Finally, Section 4.1.4 concludes by indicating the connection between the method and the concept of data depth.

4.1.1 Preliminaries

Let us begin with detailing forest-based multivariate anomaly-detection approaches, which ideas are essential for their functional extension. Thus, the *isolation forest* (IF) algorithm is briefly recalled, as well as its score-predicting mechanism, and its recent data-geometry-adapting modification called *extended isolation forest*.

Isolation forest

As a first go, let us describe the isolation forest algorithm for anomaly detection in the multivariate context in a formalized manner for clarity's sake, as well as its extended isolation forest version, see (Liu et al., 2008, 2012) and (Hariri et al., 2021) respectively. These two unsupervised algorithms can be viewed as *ensemble learning* methods insofar as they build a collection of binary trees and an anomaly scoring function based on the aggregation of the latter. Let $\boldsymbol{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$ be a training sample composed of n independent realizations of a generic random variable, X, that takes its values in a finite dimensional Euclidian space, $\mathbb{R}^d$ say, and the upper indexing will be used to refer to its constituents (=variables) $X = (X^{(1)}, \ldots, X^{(d)})^\top$.

An *isolation tree* (*i*tree in abbreviated form) $\mathcal{T}$ of depth $J \geq 1$ is a proper binary tree that represents a nested sequence of partitions of the feature space $\mathbb{R}^d$. The root node corresponds to the whole space $\mathcal{C}_{0,0} = \mathbb{R}^d$, while any node of the tree, indexed by the pair (j, k) where j denotes the depth of the node with $0 \leq j < J$ and k, the node index with $0 \leq k \leq 2^j - 1$, is associated to a subset $\mathcal{C}_{j,k} \subset \mathbb{R}^d$. A non terminal node (j, k) has two children, corresponding to disjoint subsets $\mathcal{C}_{j+1,2k}$ and $\mathcal{C}_{j+1,2k+1}$ such that $\mathcal{C}_{j,k} = \mathcal{C}_{j+1,2k} \cup \mathcal{C}_{j+1,2k+1}$. A node (j, k) is said to be terminal if it has no children.

Each *i*tree is obtained by recursively filtering a subsample of training data of size ψ in a top-down fashion, by means of the following procedure. The data set composed of the training observations present at a node (j, k) is denoted by $\boldsymbol{X}_{j,k}$. At iteration $k + 2^j$ of the *i*tree growing stage, a direction m in $\{1, \ldots, d\}$, or equivalently a *split variable* $X^{(m)}$, is selected uniformly at random (and independently from the previous draws) as well as a *split value* κ in the interval $[\min_{\boldsymbol{x} \in \boldsymbol{X}_{j,k}} \boldsymbol{x}^{(m)}, \max_{\boldsymbol{x} \in \boldsymbol{X}_{j,k}} \boldsymbol{x}^{(m)}]$ corresponding to the range of the projections of the points in $\boldsymbol{X}_{j,k}$ onto the m-th axis. The children subsets are then defined by $\mathcal{C}_{j+1,2k} = \mathcal{C}_{j,k} \cap \{\boldsymbol{x} \in \mathbb{R}^d : \boldsymbol{x}^{(m)} \leq \kappa\}$ and $\mathcal{C}_{j+1,2k+1} = \mathcal{C}_{j,k} \cap \{\boldsymbol{x} \in \mathbb{R}^d : \boldsymbol{x}^{(m)} > \kappa\}$, the children training data sets being defined as $\boldsymbol{X}_{j+1,2k} = \boldsymbol{X}_{j,k} \cap \mathcal{C}_{j+1,2k}$ and $\boldsymbol{X}_{j+1,2k+1} = \boldsymbol{X}_{j,k} \cap \mathcal{C}_{j+1,2k+1}$.

An *i*tree $\mathcal{T}$ is thus built by iterating this procedure until all training data points are isolated (or the depth limit J set by the user is attained). A preliminary subsampling stage can be performed in order to avoid swamping and masking effects, when the size of the data set is too large. When it isolates any training data point, the *i*tree contains exactly $\psi - 1$ internal nodes and ψ terminal nodes. An *i*tree constructed accordingly to a training subsample allows to assign to each training datapoint $\boldsymbol{x}_i$ a path length $h_{\mathcal{T}}(\boldsymbol{x}_i)$, namely the depth at which it is isolated from the others, *i.e.* the number of edges $\boldsymbol{x}_i$ traverses from the root node to the terminal node that contains the sole training data $\boldsymbol{x}_i$. More generally, it can be used to define an anomaly score for any point $\boldsymbol{x} \in \mathbb{R}^d$.

Anomaly score prediction

As the terminal nodes of the *i*tree $\mathcal{T}$ form a partition of the feature space, one may then define the piecewise constant function $h_\tau : \mathbb{R}^d \to \mathbb{N}$ by: $\forall \boldsymbol{x} \in \mathbb{R}^d$,

$$h_\tau(\boldsymbol{x}) = j \text{ if and only if } \boldsymbol{x} \in \mathcal{C}_{j,k} \text{ and } (j, k) \text{ is a terminal node.}$$

This random path length is viewed as an indication for its degree of abnormality in a natural manner: ideally, the more abnormal the point $\boldsymbol{x}$, the higher the probability that the quantity $h_\tau(\boldsymbol{x})$ is small. Hence, the algorithm above can be repeated $N \geq 1$ times in order to produce a collection of $i$trees $\mathcal{T}_1, \ldots, \mathcal{T}_N$, referred to as an $i$forest, that defines the scoring function

$$s_n(\boldsymbol{x}) = 2^{-\frac{1}{Nc(\psi)}\sum_{l=1}^{N} h_{\tau_l}(\boldsymbol{x})}, \tag{4.1}$$

where $c(\psi)$ is the average path length of unsuccessful searches in a binary search tree, see (Liu et al., 2008) for further details.

Extended isolation forest

Observing that the geometry of the abnormal regions of the feature space is not necessarily well-described by perpendicular splits (*i.e.* by unions of hypercubes of the cartesian product $\mathbb{R}^d$), a more flexible variant of the procedure recalled above has been proposed in (Hariri et al., 2021), in the purpose of bias reduction. Rather than selecting a direction in $\{1, \ldots, d\}$, one may choose a direction $\boldsymbol{u} \in \mathbb{S}^{d-1}$, denoting by $\mathbb{S}^{d-1}$ the unit sphere of the Euclidean space $\mathbb{R}^d$. A node is then cut by choosing randomly and uniformly a threshold value in the range of the projections onto this direction of the training data points lying in the corresponding region. As the authors illustrate in their work, in certain situations (*e.g.*, bi-modal distributions) this brings substantial improvement over the existing IF algorithm when estimating the contours of normal data.

4.1.2 The FIF algorithm

Consider the problem of learning a score function $s : \mathcal{H} \to \mathbb{R}$ that reflects the degree of abnormality of the elements in an infinite dimensional space $\mathcal{H}$ w.r.t. the random function F. Let $\mathcal{H}$ be equipped with a scalar product $\langle ., . \rangle_{\mathcal{H}}$. In the following, the proposed *functional isolation forest* (FIF) algorithm is described in detail and its properties are discussed.

A functional isolation forest is a collection of *functional isolation trees* (F-$i$trees) built from $\mathcal{F} = \{f_1, \ldots, f_n\}$, a training sample composed of independent realizations of a functional random variable, F, that takes its values in $\mathcal{H}$. Given a functional observation f, the score returned by FIF is a monotone transformation of the empirical mean of the path lengths $h_{\tau_l}(f)$ computed by the F-$i$trees $\mathcal{T}_l$, for $l = 1, \ldots, N$ as defined in (4.1) in the multivariate case. While the general construction principle depicted in Section 4.1.1 remains the same for a F-$i$tree, dealing with functional values raises the issue of finding an adequate feature space to represent various properties of a function. A function may be considered as abnormal according to various criteria of location and shape, and the features should permit to measure such properties. Therefore four ingredients have been introduced to handle functional data in a general and flexible way: (i) a set of candidate *split variables* and (ii) a scalar product both devoted to function representation, (iii) a probability distribution to sample from this set and select a single *split variable*, (iv) a probability distribution to select a *split value*. The entire construction procedure of a F-$i$tree is described in detail in Algorithm 4.1 right below.

Algorithm 4.1 (Construction procedure of a F-*i*tree)

Input: *A subsample* $\{f_1, \ldots, f_\psi\}$*, a dictionary* $\mathcal{D}$*, a probability measure* $\boldsymbol{\nu}$ *and a scalar product* $\langle \cdot, \cdot \rangle_{\mathcal{H}}$*.*

1. **Initialization:** *The root node indexed by* $(0,0)$ *is associated with the whole input space* $\mathcal{C}_{0,0} = \mathcal{H}$*. The construction starts with the training data set* $\mathcal{F}_{0,0} = \{f_1, \ldots, f_\psi\}$ *composed of* n *i.i.d. realizations of the random variable* F*. Go to (2.) with* $(j = 0, k = 0)$*.*

2. **Stopping criterion:** *Test if the node* (j,k) *is terminal: a node* (j,k) *is declared as terminal if the intersection between the current set* $\mathcal{C}_{j,k}$ *and the current training set* $\mathcal{F}_{j,k}$ *is reduced to a single data point or to a set of predefined cardinal. If the node is terminal, then stop the construction for this node, otherwise go to (3.).*

3. **Children node construction:** *A non-terminal node* (j,k) *is split in three steps as follows:*

   (a) *Choose a split variable* g *according to the probability distribution* $\boldsymbol{\nu}$ *on* $\mathcal{D}$*.*

   (b) *Choose randomly and uniformly a split value* κ *in the interval*

   $$\left[\min_{f \in \mathcal{F}_{j,k}} \langle f, g \rangle_{\mathcal{H}}, \max_{f \in \mathcal{F}_{j,k}} \langle f, g \rangle_{\mathcal{H}} \right].$$

   (c) *Form the children subsets*

   $$\begin{aligned} \mathcal{C}_{j+1,2k} &= \mathcal{C}_{j,k} \cap \{f \in \mathcal{H} : \langle f, g \rangle_{\mathcal{H}} \leq \kappa\}, \\ \mathcal{C}_{j+1,2k+1} &= \mathcal{C}_{j,k} \cap \{f \in \mathcal{H} : \langle f, g \rangle_{\mathcal{H}} > \kappa\}, \end{aligned}$$

   as well as the children training data sets

   $$\mathcal{F}_{j+1,2k} = \mathcal{F}_{j,k} \cap \mathcal{C}_{j+1,2k} \text{ and } \mathcal{F}_{j+1,2k+1} = \mathcal{F}_{j,k} \cap \mathcal{C}_{j+1,2k+1}.$$

4. **Recursion:** *Apply the building procedure starting from (2.) to nodes* $(j+1, 2k)$ *and* $(j+1, 2k+1)$*.*

Output: $(\mathcal{C}_{(0,0)}, \mathcal{C}_{(1,1)}, \ldots)$*.*

Function representation

To define the set of candidate *split variables*, a direct extension of the original IF algorithm (Liu et al., 2008) would be to randomly draw an argument value (*e.g.* time), and use functional evaluations at this point to split a node, but this boils down to only rely on instantaneous observations of functional data to capture anomalies, which in practice will be usually interpolated. Drawing a direction on a unit sphere as in (Hariri et al., 2021) is no longer possible due to the potentially excessive richness of $\mathcal{H}$. To circumvent these

difficulties, the observations are projected on elements of a dictionary $\mathcal{D} \subset \mathcal{H}$ that is chosen to be rich enough to explore different properties of data and well appropriate to be sampled in a representative manner. More explicitly, given a function $g \in \mathcal{D}$, the projection of a function $f \in \mathcal{H}$ on $\mathcal{D}$, $\langle f, g \rangle_{\mathcal{H}}$ defines a feature that partially describes f. When considering all the functions of dictionary $\mathcal{D}$, one gets a set of candidate *split variables* that provides a rich representation of function F, depending on the nature of the dictionary. Dictionaries have been throughly studied in the signal processing community to achieve *sparse coding* of signals, see ,*e.g.*, Mallat and Zhang (1993). They also provide a way to incorporate *a priori* information about the nature of the data, a property very useful in an industrial context in which functional data often come from the observation of a well known device and thus can benefit from expert knowledge.

Sampling a *split variable* Once a dictionary is chosen, a probability distribution $\boldsymbol{\nu}$ on $\mathcal{D}$ is defined to draw a *split variable* g. Note that the choice of the sampling distribution $\boldsymbol{\nu}$ gives an additional flexibility to orientate the algorithm towards the search for specific properties of the functions.

Sampling a *split value* Given a chosen *split variable* g and a current training data set $\mathcal{F}_{j,k}$, a *split value* is uniformly drawn in the real interval defined by the smallest and largest values of the projections on g when considering the observations present in the node.

Choice of dictionary and scalar product

The choice of a suited dictionary plays a key role in construction of the FIF anomaly score. The dictionary can consist of deterministic functions, incorporate stochastic elements, contain the observations from $\mathcal{F}$, or be a mixture of several mentioned options. In *computational harmonic analysis*, a wide variety of bases or frames, such as wavelets, ridgelets, cosine packets, brushlets and so on, have been developed in the last decades in order to represent efficiently/parsimoniously functions, signals or images exhibiting specific form of singularities (*e.g.* located at isolated points, along hyperplanes) and may provide massive dictionaries. The following ones will be used throughout this section: *mexican hat wavelet dictionary*, *Brownian motion dictionary*, *Brownian bridge dictionary*, *cosine dictionary*, *uniform indicator dictionary*, *dyadic indicator dictionary*, and the *self-data dictionary* containing the data set itself. See Sections B and C in Supplementary Materials of Staerman et al. (2019) for detailed definitions of these dictionaries and further discussion on them, respectively.

Besides the dictionary, the *scalar product* defined on $\mathcal{H}$ brings additional flexibility to detect different types of anomalies. While L_2 scalar product allows for detection of *location anomalies*, L_2 scalar product of derivatives (or slopes) would allow to detect anomalies regarding shape. This last type of anomalies can be challenging; *e.g.* Hubert et al. (2015) mention that *shape anomalies* are more difficult to detect, and Mosler and M. (2017) argue that one should consider both location and slope simultaneously for distinguishing complex curves. Beyond these two, a wide diversity of scalar products can be used, involving a

variety of L_2-scalar products related to derivatives of certain orders, like in the definition of Banach spaces such as weighted Sobolev spaces, see Maz'ya (2011).

The FIF algorithm, together with all mentioned above dictionaries, has been implemented as `Python` software which can be downloaded from the following link: https://github.com/GuillaumeStaermanML/FIF.

4.1.3 Ability to detect variety of anomalies

As discussed above, most of the state-of-the-art methods have a focus on a certain type of anomalies and are unable to detect various deviations from the normal behavior. The flexibility of the FIF algorithm allows for choosing the scope of the detection by selecting both the scalar product and the dictionary. By choosing appropriate scalar product and dictionary, FIF is able to detect a great diversity of deviations from normal data. To account for both location and shape anomalies, the following scalar product will be employed that provides a compromise between the both

$$\langle f, g\rangle := \alpha \times \frac{\langle f, g\rangle_{L_2}}{||f||\,||g||} + (1-\alpha) \times \frac{\langle f', g'\rangle_{L_2}}{||f'||\,||g'||}, \quad \alpha \in [0,1]\,,$$

with its use being illustrated right below. Thus, setting $\alpha = 1$ yields the classical L_2 scalar product, $\alpha = 0$ corresponds to the L_2 scalar product of derivative, and $\alpha = 0.5$ is the Sobolev $W_{1,2}$ scalar product. To illustrate the FIF's ability to detect a wide variety of anomalies at a time, let us calculate the FIF anomaly scores with the Sobolev scalar product and the *gaussian wavelets dictionary* for a sample consisting of 105 curves defined as follows (inspired by Cuevas et al., 2007, see Figure 4.1):

- 100 curves defined by $f(t) = 30(1-t)^q t^q$ with q equispaced in $[1, 1.4]$,
- 5 *abnormal* curves composed by one isolated anomaly $x_0(t) = 30(1-t)^{1.2}t^{1.2}$ with a jump at $t = 0.7$, one magnitude anomaly $x_1(t) = 30(1-t)^{1.6}t^{1.6}$ and three kinds of shape anomalies $x_2(t) = 30(1-t)^{1.2}t^{1.2} + \sin(2\pi t)$, $x_3(t) = 30(1-t)^{1.2}t^{1.2}$ noised by $\varepsilon \sim \mathcal{N}(0, 0.3^2)$ on the interval $[0.2, 0.8]$ and $x_4(t) = 30(1-t)^{1.2}t^{1.2} + \frac{1}{2}\sin(10\pi t)$.

One can see that the five anomalies, although very different, are all detected by FIF which delivers a significantly differing score.

For an extensive simulation and real-data study the reader is referred to Section 4 of Staerman et al. (2019) and to it accompanying Supplementary Materials.

4.1.4 Connection to data depth

FIF can be easily extended to the multivariate functional data, *i.e.* when the quantity of interest lies in $\mathbb{R}^d$ for each moment of time:

$$\begin{aligned} F : \Omega &\longrightarrow (\mathcal{H}([0,1]))^{\otimes d} \\ \omega &\longmapsto \big((F_1(\omega))_{t\in[0,1]},\ \ldots,\ (F_d(\omega))_{t\in[0,1]}\big) \end{aligned}$$

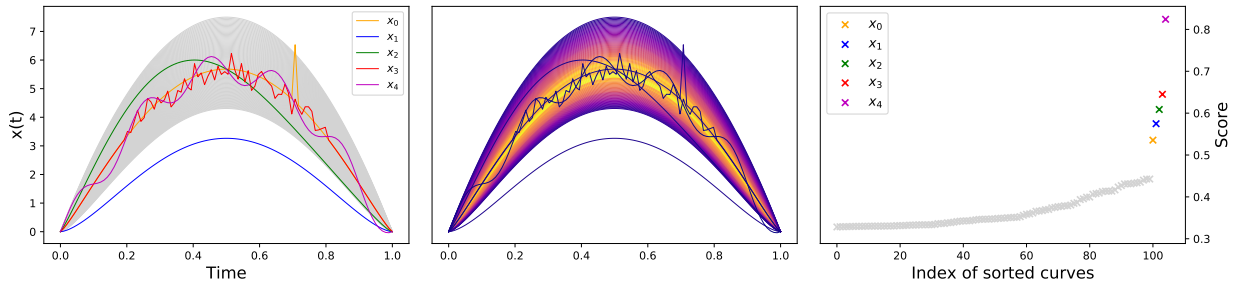


Figure 4.1: *The simulated data set with the five introduced anomalies (left). The scored data set (middle), the darker the color, the more the curves are considered anomalies. The sorted anomaly score of the data set (right).*

For this, the coordinate-wise sum of the d corresponding scalar products is used to project the data onto a chosen dictionary element:

$$\langle \mathbf{f}, \mathbf{g} \rangle_{\mathcal{H}^{\otimes d}} := \sum_{i=1}^{d} \langle f_i, g_i \rangle_{\mathcal{H}}.$$

Further, a dictionary should be defined in $(\mathcal{H}([0,1]))^{\otimes d}$. This can be done, *e.g.*, by either componentwise application of one or several univariate dictionaries from Section 4.1.2 above and Section B in Supplementary Materials of Staerman et al. (2019), or by construction of special d-variate ones. For illustration purposes, regard the following example constructed

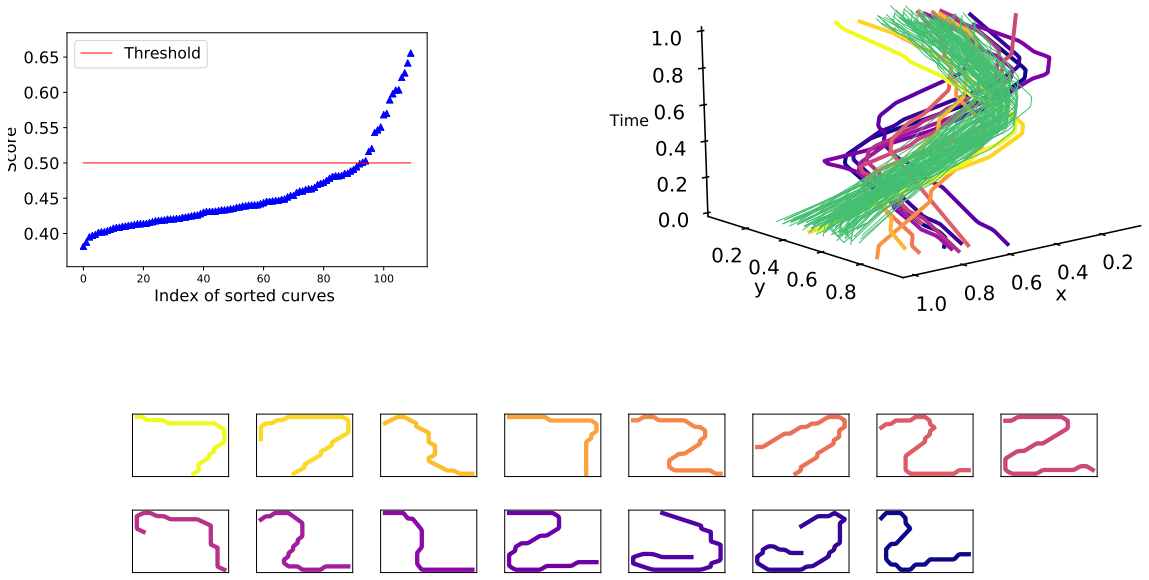


Figure 4.2: *FIF anomaly scores for a sample of* 110 *digits (*100 *"7"s and* 10 *"2"s). Left plot corresponds to the sorted score of these curves. Right plot represents the digits in three dimensions, green ones correspond to normal data, anomaly score increases from yellow to dark blue. Bottom plot shows the fifteen detected anomalies.*

based on the MNIST (LeCun et al., 1998) data set. First, the digits' contours (skeletons) are extracted using `skimage Python` library (Van Der Walt et al., 2014). Then, each observation is transformed into a curve in $(L_2([0,1]) \times L_2([0,1]))$ (one vertical and one horizontal coordinates) using length parametrization on $[0,1]$. An anomaly detection problem is constructed by taking 100 curves from class "7" and adding 10 observations from class "2". FIF is applied with the two-dimensional *sinecosine* dictionary and the following scalar product: $\langle \mathbf{f}, \mathbf{g} \rangle_{(L_2)^{\otimes d}} = \sum_{i=1}^{d} \langle f_i, g_i \rangle_{L_2}$. *sinecosine* is constructed as a direct extension of *cosine* dictionary introduced for FIF by selecting randomly cosine or sine function on each coordinates. Figure 4.2 shows anomaly detection using the visual elbow rule to define the threshold. Among those detected, five digits are indeed "7"s, but do not resemble them and thus are identified as anomalies.

Regarding FIF score as an anomaly ranking yields a connection to the notion of the statistical depth function. Letting $s_n(\mathbf{f}; \mathcal{F})$ denote FIF score of function $\mathbf{f}$ w.r.t. a functional data set $\mathcal{F}$, a data depth measure based on FIF score can be defined for (multivariate) functional data as: $FD_{FIF}(\mathbf{f}|\mathcal{F}) = 1 - s_n(\mathbf{f}; \mathcal{F})$.

Data depth proves to be a useful tool for a low-dimensional data representation called DD-plot. Using this property, Li et al. (2012) and Lange et al. (2014b) define a DD-plot classifier which consists in applying a multivariate classifier to the depth-based map. Low-dimensional representation is of particular interest for functional data and a DD-plot classifier can be defined using the FIF-based data depth. Let $\mathcal{S}^{trn} = \mathcal{S}^1 \cup ... \cup \mathcal{S}^q$ be a training set for supervised classification containing q classes, each subset $\mathcal{S}^j$ standing for class j. The depth map is defined as follows:

$$\mathbf{f} \mapsto \xi(\mathbf{f}) = \left(FD_{FIF}(\mathbf{f}|\mathcal{S}^1), ..., FD_{FIF}(\mathbf{f}|\mathcal{S}^q) \right) \in [0,1]^q \,.$$

As an illustration, let us apply the depth map to 3 digits ("1", "5" and "7", 100 observations per digit for training and 100 for testing) of the MNIST dataset after their transformation to bivariate functions as bove (see Figure 4.3). One observes appealing geometrical interpretation (observe, *e.g.*, the location of the abnormally distant—from their corresponding classes—observations) and a clear separation of the classes. To illustrate separability,

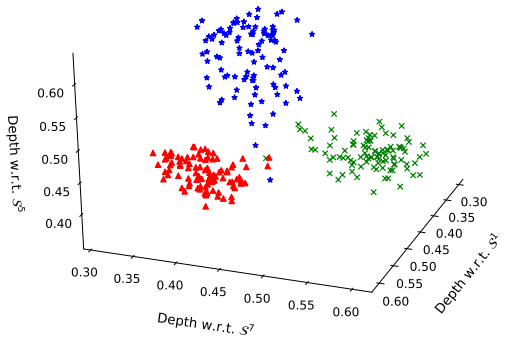


Figure 4.3: *Depth space embedding of the three digits ("1", "5" and "7") of the MNIST dataset.*

linear multiclass (one-against-all) SVM was applied in the depth space, which delivers the accuracy of 99% on the test data.

4.2 Geometric depth approach

In this section, a novel notion of depth for functional data is proposed: the area-of-the-convex-hull (ACH) depth. As seen from the simulation and real-data study of Section 4.2.4, this notion is well suited for detection of functional anomalies, and in particular the isolated anomalies. The rest of the section is organized as follows. In Section 4.2.1, the proposed depth notion is directly introduced. Next, Section 4.2.2 discusses its statistical and computational properties. Further, in Section 4.2.3 the question of tuning the hyperparameters is tackled. Finally, Section 4.2.4 benchmarks the proposed functional depth against the state-of-the-art methods. Implementation of the ACH depth can be found under the following link: https://github.com/GuillaumeStaermanML/ACHD.

4.2.1 The area of the convex hull of functions

The purpose here is to present at length the statistical depth function proposed for path-valued random variables. As shall be seen below, its definition is based on very simple geometrical ideas and various desirable properties can be easily checked from it. Statistical and computational issues are also discussed at length. By $\mathcal{K}_2$ is meant the collection of all compact subsets of $\mathbb{R}^2$ and λ denotes Lebesgue measure on the plane $\mathbb{R}^2$. Consider an i.i.d. sample $F_1, \ldots, F_n$ drawn from the same distribution as a random function F belonging to a class of random measures $\mathcal{P}(\mathcal{C}([0,1]))$. The graph of any function f in $\mathcal{C}([0,1])$ is denoted by

$$graph(f) = \{(t, y) : y = f(t), t \in [0,1]\},$$

while $graph(\{f_1, \ldots, f_n\})$ denotes the set

$$\bigcup_{i=1}^{n} graph(\{f_i\})$$

defined by a collection of $n \geq 1$ functions $\{f_1, \ldots, f_n\}$ in $\mathcal{C}([0,1])$. Now, let us give a precise definition of the proposed statistical depth measure for random variables valued in $\mathcal{C}([0,1])$.

Definition 4.1 *(Staerman et al., 2020) Let $J \geq 1$ be a fixed integer. The ACH depth of degree J is the function $FD_{\text{ACH(J)}} : \mathcal{C}([0,1]) \times \mathcal{P}(\mathcal{C}([0,1])) \to [0,1]$ defined by: $\forall f \in \mathcal{C}([0,1])$,*

$$FD_{\text{ACH(J)}}(f|F) = \mathbb{E}\left[\frac{\lambda\left(conv\left(graph\left(\{F_1, \ldots, F_J\}\right)\right)\right)}{\lambda\left(conv\left(graph\left(\{F_1, \ldots, F_J\} \cup \{f\}\right)\right)\right)}\right],$$

where $F_1, \ldots, F_J$ are i.i.d. random functions drawn as F. Its average version $\overline{FD}_{\text{ACH(J)}}$ is defined by: $\forall f \in \mathcal{C}([0,1])$,

$$\overline{FD}_{\text{ACH(J)}}(f|F) = \frac{1}{J}\sum_{j=1}^{J} D_{\text{ACH(j)}}(f|F).$$

The choice of J leads to various views on the random function F, the average variant permitting to combine all of the realisations (up to degree J). When the size of the functional sample $\mathcal{F} = \{f_1, \ldots, f_n\}$, $n \geq J$, an unbiased statistical estimation of $FD_{\text{ACH(J)}}(\cdot|F)$ can be obtained by computing the symmetric U-statistic of degree J, see Lee (1990): $\forall f \in \mathcal{C}([0,1])$,

$$FD_{\text{ACH(J)}}(f|\mathcal{F}) = \frac{1}{\binom{n}{J}} \sum_{1 \leq i_1 < \ldots < i_J \leq n} \frac{\lambda\left(conv\left(graph\left(\{f_{i_1}, \ldots, f_{i_J}\}\right)\right)\right)}{\lambda\left(conv\left(graph\left(\{f_{i_1}, \ldots, f_{i_J}, f\}\right)\right)\right)}. \tag{4.2}$$

Considering the empirical average version given by

$$\forall f \in \mathcal{C}([0,1]), \quad \overline{FD}_{\text{ACH(J)}}(f|\mathcal{F}) = \frac{1}{J} \sum_{j=1}^{J} FD_{\text{ACH(j)}}(f|\mathcal{F})$$

brings some "stability". However, the computational cost rapidly increasing with J, small values of J are preferred in practice. Moreover, as illustrated in Section 4.2.3, $J = 2$ already yields satisfactory results.

4.2.2 Statistical and computational properties

First, let us state the properties satisfied by the (average) ACH depth function:

- (Non-degeneracy) For any non atomic random function F belonging to $\mathcal{P}(\mathcal{C}([0,1]))$, one has
$$\inf_{f \in \mathcal{C}([0,1])} FD(f|F) < \sup_{f \in \mathcal{C}([0,1])} FD(f|F).$$
- (Affine invariance) The depth FD is said to be (scalar-) affine invariant if for any f in $\mathcal{C}([0,1])$ and all a, b in $\mathbb{R}$, one has
$$FD(f, F) = FD(af + b|aF + b).$$
- (Vanishing at ∞) For any non atomic random function F belonging to $\mathcal{P}(\mathcal{C}([0,1]))$,
$$FD(f|F) \underset{||f||_\infty \longrightarrow \infty}{\longrightarrow} \inf_{g \in \mathcal{C}([0,1])} FD(g|F).$$
- (Continuity in f) For any non atomic random function F belonging to $\mathcal{P}(\mathcal{C}([0,1]))$, the function $f \mapsto FD(f|F)$ is continuous w.r.t. the sup norm.
- (Continuity in F) For all f in $\mathcal{C}([0,1])$, the mapping from $\mathcal{P}(\mathcal{C}([0,1]))$ $F \mapsto FD(f|F)$ is continuous w.r.t. the Lévy-Prohorov metric.

Proposition 4.1 *(Staerman et al., 2020) For all $J \geq 1$, the depth function $FD_{\text{ACH(J)}}(f|F)$ (resp., $\overline{FD}_{\text{ACH(J)}}(f|F)$) fulfills the following properties:* 'non-degeneracy', 'affine invariance', 'vanishing at infinity', 'continuity in f' *and* 'continuity in F'.

In a functional space, not satisfying *maximality at center* is not an issue. For instance, though the constant trajectory $f(t) \equiv 0$ is a center of symmetry for the Brownian motion, it is clearly not representative of this distribution. In contrast, *scalar-affine invariance* is relevant, insofar as it allows z-normalization of the functional data and *continuity in P* is essential to derive the consistency of $FD_{\mathrm{ACH(J)}}(\cdot|\mathcal{F})$ (respectively, of $\overline{FD}_{\mathrm{ACH(J)}}(\cdot|\mathcal{F})$), as stated below.

Theorem 4.1 *(Staerman et al., 2020) Let $J \geq 1$ and $F_1, \ldots, F_n$ be $n \geq J$ independent copies of a generic random function F belonging to $\mathcal{P}(\mathcal{C}([0,1]))$. As $n \to \infty$, one has, for any $f \in \mathcal{C}([0,1])$, with probability one,*

$$\left|FD_{\mathrm{ACH(J)}}(f|\mathcal{F}) - FD_{\mathrm{ACH(J)}}(f|F)\right| \to 0$$

and

$$\left|\overline{FD}_{\mathrm{ACH(J)}}(f|\mathcal{F}) - \overline{FD}_{\mathrm{ACH(J)}}(f|F)\right| \to 0.$$

As mentioned above, only sampled curves are available in practice. Each random curve F_i being observed at fixed time points $0 = t_1^{(i)} < t_2^{(i)} < \ldots < t_{p_i}^{(i)} = 1$ (potentially different for each F_i) with $p_i \geq 1$, denote by $F_1', \ldots, F_n'$ the continuous curves reconstructed from the sampled curves $(F_i(t_1^{(i)}), \ldots, F_i(t_{p_i}^{(i)}))$, $1 \leq i \leq n$, by linear interpolation. From a practical perspective, one considers the estimator $FD'_{\mathrm{ACH(J)}}(f|\mathcal{F})$ of $FD_{\mathrm{ACH(J)}}(f|F)$ given by the approximation of $FD_{\mathrm{ACH(J)}}(f|\mathcal{F})$ obtained when replacing the F_i's by the F_i''s in (4.2). The (computationally feasible) estimator $\overline{FD}'_{\mathrm{ACH(J)}}(f|\mathcal{F})$ of $\overline{FD}_{\mathrm{ACH(J)}}(f|F)$ is constructed in a similar manner. The result stated below shows that this approximation stage preserves almost-sure consistency.

Theorem 4.2 *(Staerman et al., 2020) Let $J \leq n$. Suppose that, as $n \to \infty$,*

$$\delta = \max_{1 \leq i \leq n} \max_{2 \leq k \leq p_i} \left\{ t_{k+1}^{(i)} - t_k^{(i)} \right\} \to 0.$$

As $n \to \infty$, for any $f \in \mathcal{C}([0,1])$, with probability one it holds

$$\left|FD'_{\mathrm{ACH(J)}}(f|\mathcal{F}) - FD_{\mathrm{ACH(J)}}(f|F)\right| \to 0$$

and

$$\left|\overline{FD}'_{\mathrm{ACH(J)}}(f|\mathcal{F}) - \overline{FD}_{\mathrm{ACH(J)}}(f|F)\right| \to 0.$$

Given the batch of continuous and piecewise linear curves $F_1', \ldots, F_n'$, although the computation cost of the area of their convex hull is of order $O(p \log p)$ with $p = \max_i p_i$, that of the U-statistic $FD'_{\mathrm{ACH(J)}}(f|\mathcal{F})$ (and *a fortiori* that of $\overline{FD}'_{\mathrm{ACH(J)}}(f|\mathcal{F})$) becomes very expensive as soon as $\binom{n}{J}$ is large. As pointed out in López-Pintado and Romo (2009), the choice $J = 2$ for statistics of this type may lead to a computationally tractable procedure, while offering a reasonable representation of the distribution. Varying J permits to capture much more information in general. For this reason, an *incomplete* version of the U-statistic $FD'_{\mathrm{ACH(J)}}(f|\mathcal{F})$ is computed using a basic Monte-Carlo approximation scheme with $K \geq 1$

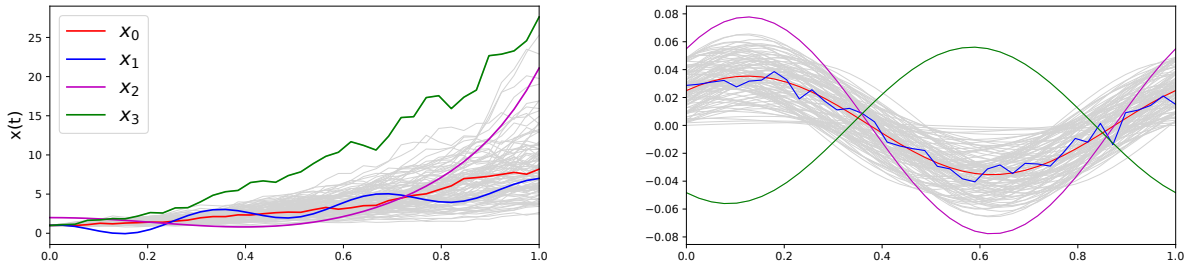


Figure 4.4: *Data sets (a) (left) and (b) (right) containing 100 paths with four selected observations. The colors are the same for the four selected observations of both data sets (a) and (b).*

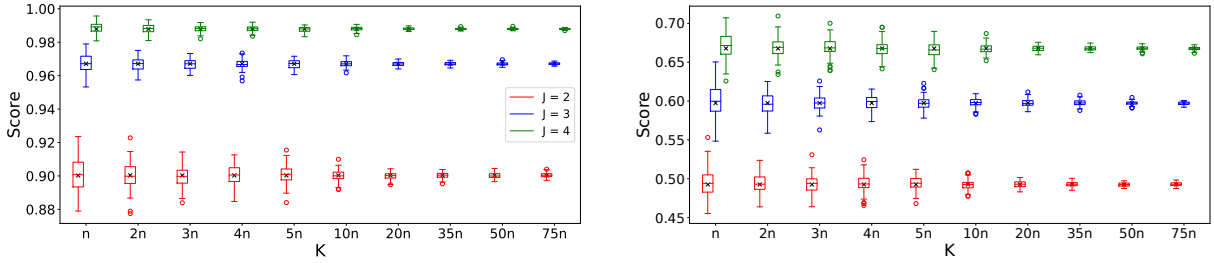


Figure 4.5: *Boxplots of the approximations of* $FD_{\mathrm{ACH(J)}}(x_0|\mathcal{F})$ *(left) and* $FD_{\mathrm{ACH(J)}}(x_3|\mathcal{F})$ *(right) over different size of* K*. The black crosses correspond to the exact depth measure* $FD_{\mathrm{ACH(J)}}(\cdot|\mathcal{F})$ *for each* J *respectively.*

replications: rather than averaging over all $\binom{n}{J}$ subsets of $\{1, \ldots, n\}$ with cardinality J to compute $FD'_{\mathrm{ACH(J)}}(f|\mathcal{F})$, one averages over $K \geq 1$ subsets drawn with replacement, forming an *incomplete U-statistic*, see Enqvist (1978). The same approximation procedure can be applied (in a randomized manner) to each of the U-statistics involved in the average $\overline{FD}'_{\mathrm{ACH(J)}}(f|\mathcal{F})$, as described in Supplementary Materials of Staerman et al. (2020).

4.2.3 Choosing tuning parameters K and J

For the sake of simplicity, the two same simulated data sets, represented in Figure 4.4, are used throughout this and the following subsections. The data set (a) corresponds to sample path segments of the *geometric Brownian motion* with mean 2 and variance 0.5, a stochastic process widely used in statistical modeling. The data set (b) consists of smooth curves given by $f(t) = a\cos(2\pi t) + b\sin(2\pi t)$, $t \in [0, 1]$, where a and b are independently and uniformly distributed on $[0, 0.05]$, as proposed by Claeskens et al. (2014). Four curves $\{x_i : i \in \{0, 1, 2, 3\}\}$ have been incorporated to each data set: a deep curve and three atypical curves (*anomalies*), with expected depth-induced ranking $FD_{\mathrm{ACH(J)}}(x_3|\mathcal{F}) < FD_{\mathrm{ACH(J)}}(x_2|\mathcal{F}) \approx FD_{\mathrm{ACH(J)}}(x_1|\mathcal{F}) < FD_{\mathrm{ACH(J)}}(x_0|\mathcal{F})$.

Parameter K reflects the trade-off between statistical performance and computational time. In order to investigate its impact on the stability of the method, let us compute depths of the deepest and most atypical curves (x_0 and x_3) for data set (b), taking $J = 2, 3, 4$. Figure 4.5 presents boxplots of the approximated ACH depth (together with the exact values of ACH depth) over 100 repetitions. Note that, as expected, depth values grow with J. The variance of the depth decreases taking sufficiently small values for $K = 5n$ and almost disappearing for $K \geq 20n$, while decreasing pattern remains the same for different values of K. For these reasons, $K = 5n$ is kept in what follows.

The choice of J is less obvious, and clearly when describing an observation in a functional space a substantial part of information is lost anyway. Nevertheless, one observes that computational burden increases exponentially with J and thus smaller values are preferable. Figure 4.6 shows the rank-rank plots of data sets (a) and (b) for small values of $J = 2, 3, 4$ and indicates, that depth-induced ranking does not change much with J. Thus, for saving computational time, value $J = 2$ is used in all subsequent experiments.

4.2.4 Robustness and anomaly detection

Under robustness of a statistical estimator one understands its ability not to be "disturbed" by atypical observations. Here, robustness of ACH depth is explored in the following simulation study: between the original data set and the same data set contaminated with

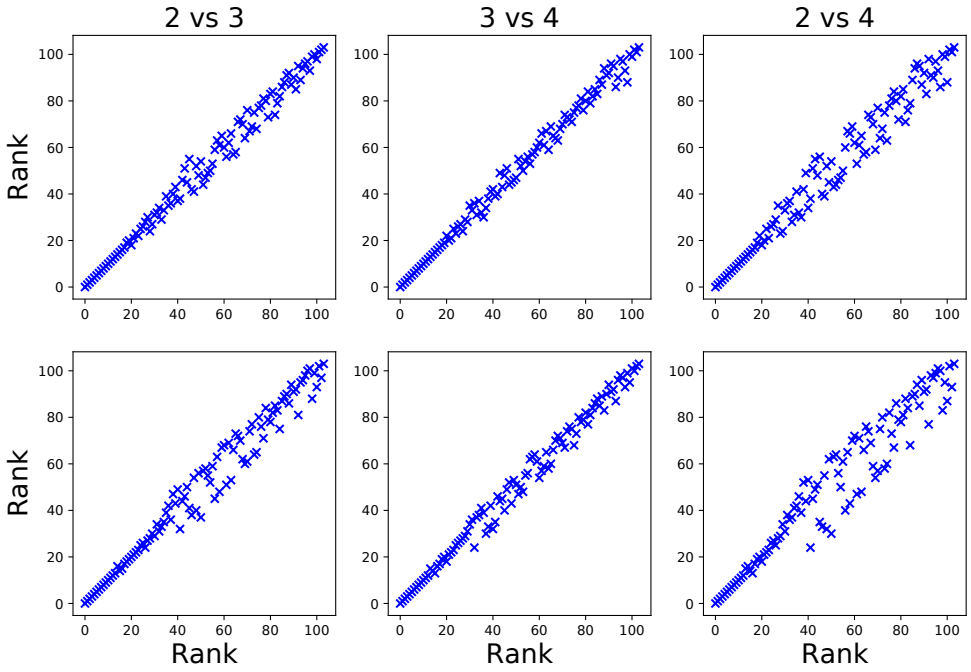


Figure 4.6: *Rank-rank plot for different values of J (2, 3 and 4). The top plots represent the ranks over the data set (a) while the bottom plots represent those over the data set (b).*

anomalies, (averaged over 10 random repetitions) Kendall's τ distance is measured, of two depth-induced rankings σ and σ', respectively, of the original data:

$$d_\tau(\sigma, \sigma') = \frac{n(n-1)}{2} \sum_{i<j} \mathbb{I}_{\{(\sigma(i)-\sigma(j))(\sigma'(i)-\sigma'(j))<0\}} \cdot$$

In their overview work, Hubert et al. (2015) introduce taxonomy for atypical observations, focusing on *location*, *isolated*, and *shape* anomalies. Here, *location* anomalies are added to data set (a) and *isolated* and *shape* anomalies to data set (b); other types of anomalies for both data sets can be found in Supplementary Materials of Staerman et al. (2020). The abnormal functions are constructed as follows. *Location* anomalies for data set (a) are $\tilde{x}(t) = x(t) + ax(t)$ with a drawn uniformly on $[0, 1]$. *Isolated* anomalies for data set (b) are constructed by adding a peak at t_0 (drawn uniformly on $[0, 1]$) of amplitude b (drawn uniformly on $[0.03, 0.06]$) such that $\tilde{y}(t_0) = y(t_0) + b$ and $\tilde{y}(t) = y(t)$ for any $t \neq t_0$. *Shape* anomalies for data set (b) are $\tilde{z}(t) = z(t) + 0.01 \times \cos(2\pi t\gamma) + 0.01 \times \sin(2\pi t\gamma)$ with γ drawn uniformly from $\{1, 2, ..., 10\}$. By varying the percentage of abnormal observations α, ACH depth is compared to several of the most know in the literature depth approaches: the *functional projection depth* based on Stahel-Donoho outlyingness (FSDO) (Hubert et al., 2015) (see Definition 3.1) and the *functional Tukey depth* (FT) (Claeskens et al., 2014) (being univariate version of FD_{MFHD} with constant weighting, see Definition 3.5), and also to the *functional isolation forest* (FIF) algorithm (see Section 4.1 and Staerman et al., 2019)

Table 4.1: *Kendall's tau distances between the ranks returned for normal data (σ_0) and contaminated data (σ_α, over different portion of contamination α with location, isolated and shape anomalies) for ACHD and three state-of-the-art methods. Bold numbers indicate best stability of the ranks over the contaminated data sets.*

$d_\tau(\sigma_0, \sigma_\alpha)(\times 10^{-2})$							
	$\boldsymbol{\alpha}$	0	5	10	15	25	30
ACHD	Location	0	**0.6**	**1.3**	**2.2**	**4.3**	**5.2**
	Isolated	0	**0.3**	**1.3**	**0.9**	**1.6**	**2.4**
	Shape	0	**0.9**	**2**	**2.6**	**4.2**	**4.7**
FSDO	Location	0	3.6	7.3	10	16	20
	Isolated	0	0.8	3.6	3.2	7.2	9.4
	Shape	0	1.6	2.9	4.2	6.6	7.4
FT	Location	0	5.1	9.5	13	20	23
	Isolated	0	0.7	2.7	2.7	5.9	7.2
	Shape	0	1.7	2.9	4.3	6.6	7.7
FIF	Location	0	7	8.2	7.3	7.3	8.9
	Isolated	0	9.3	12	11	10	12
	Shape	0	7.4	7.9	10	14	14

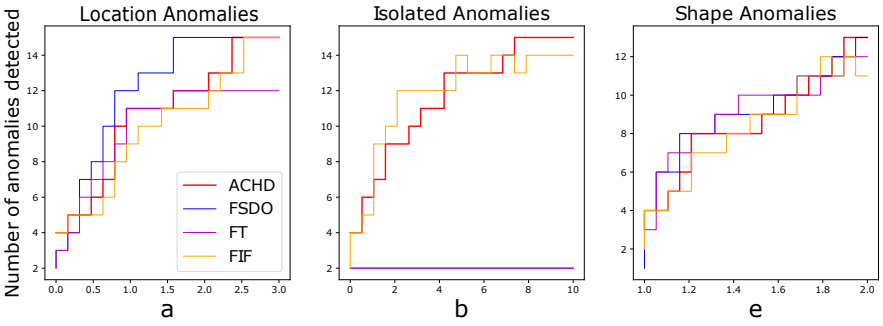


Figure 4.7: *Number of anomalies detected over a grid of parameters for three types of anomalies (location, isolated, and shape) for ACHD and three further state-of-the-art methods.*

which proves satisfactory anomaly detection; see Table 4.1. One can observe that ACH depth consistently preserves depth-induced ranking despite inserted abnormal observations, even if their fraction α reaches 30%. FSDO behaves competitively giving slightly better results than ACH depth for shape anomalies.

Further, let us explore the ability of ACH depth to detect atypical observations. For this, an experiment is conducted in settings similar to the one from above, while changing degree of abnormality gradually for 15 (out of 100 curves) in data set (a). Thus, a in $[0, 3]$ is altered for *location anomalies*, b in $[0, 10]$ for *isolated anomalies*, and e in $[1, 2]$ for *shape anomalies* to amplify the "spikes" of oscillations such that $\tilde{z}(t) = ez(t)$. (For an illustration of abnormal curves the reader is referred to Supplementary Materials of Staerman et al., 2020). Figure 4.7 indicates number of anomalies detected by ACHD, FSDO, FT, and FIF for different parameters of abnormality. While it is difficult to find the general winner, ACHD behaves favorably in all the considered cases and clearly outperforms the two other depths when the data is contaminated with *isolated anomalies*.

Let us conclude this section with a real-world data benchmark based on three data sets: Octane (Esbensen et al., 2002, Rousseeuw et al., 2006), Wine (Larsen et al., 2006), and EOG (Dau et al., 2018). The Wine data set consists of 397 measurements of proton nuclear magnetic resonance spectra of 40 different wine samples, the Octane data set contains 39 near infrared spectra of gasoline samples with 226 measurements, while the EOG data set

Table 4.2: *Portion of detected anomalies of benchmark methods for the Octane, Wine, and EOG data sets.*

	ACHD	FSDO	FT	FIF	IF	OC
Octane	**1**	0.5	0.33	**1**	0.5	0.5
Wine	**1**	0	0	**1**	0	**1**
EOG	**0.73**	0.55	0.48	0.43	0.63	0.6

represents the electrical potential between electrodes placed at points close to the eyes with 1250 measurements (graphs of the three data sets can be found in Supplementary Materials of Staerman et al., 2020). As pointed out by Hubert et al. (2015), it is difficult to detect anomalies in the first two data sets, while they are easily seen during the human eye inspection. For the EOG data set, assign smaller of the two classes to be abnormal. To the existing state-of-the-art methods, *isolation forest* (IF) (Liu et al., 2008) and the *one-class SVM* (OC) (Schölkopf et al., 2001) are added here—multivariate methods applied after a proper dimension reduction (to the dimension 10) using FPCA (Ramsay and Silverman, 2002). Portions of detected anomalies (by all the considered methods), indicated in Table 4.2, hint on very competitive performance of ACH depth in the addressed benchmark.

Chapter 5

Outlook and conclusions

This chapter gathers projects on which I plan to work in the years to come. Naturally, the degree of details is the strongest for the first of them, and gradually decreases throughout the chapter.

Section 5.1 describes the project being a fusion of methodology, computation, implementation and applications aimed to make data depth available for treatment of contemporary large-scale data. Section 5.2 delineates the project involving an industrial collaboration to provide anomaly detection tools for heterogenous and large-scale data measured on a production line. Section 5.3 enumerates two more projects, challenging to different degree. Section 5.4 concludes.

5.1 Large-scale data depth: computation and applications

In the past decades, data analysis became ubiquitous in many scientific fields, with the notion of data depth being one of its powerful tools, which brought answers to important contemporary questions. As such, today, it confronts challenges of big data often impaired by *curse of high dimensionality* on scales unimaginable until recently, with hyperspectral imagery, DNA microarrays, financial high-frequency series and social networks being only a few examples. While data depth is becoming of increasing importance in applications, practitioners are still limited *by computational burden of algorithms and shortage of implementations*.

Analogously with other depth notions, Chapter 1 of this manuscript suggests methods for computation of data depth in *low dimensions*, with algorithms quickly becoming *timely infeasible*. It is the case already, *e.g.*, with $d > 3$ for projection or simplicial depth, $d > 5$ for the halfspace depth, and $d > 20$ for zonoid depth, and this when $\boldsymbol{X}$ contains only several thousand points. Clearly, this is far not sufficient for contemporary demand, *e.g.*: many thousands of brain scans, with around $n = 10^6$ points each, for a single axonal path; many thousands of high-resolution spectra containing (dozens) thousands ($= d$) of wavelengths each. For this reason, depth applications even to moderate-size data are scarce in the lit-

erature with few recent references (Dutta et al., 2016, Jeong et al., 2016, Kleindessner and Von Luxburg, 2017) only. *Large-scale real-data applications of data depth are underdeveloped.*

While axiomatics for data depth has been thoroughly investigated—in the form of required statistical properties—this literature almost entirely ignores its computational aspects, which are so important since being the main limitation. From this point of view, computation of each depth function seems to be rather art than science. Furthermore, computational properties of data depth are as good as never explicitly taken into account during development of succeeding depth notions. It is thus planned to systematically elaborate on both points. By taking into consideration optimization (say in $\boldsymbol{u} \in \mathbb{R}^k$), the depth function $D(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{X})$ shall be regarded simultaneously as both depth measure (for $\boldsymbol{x}$) and optimization problem (with respect to $\boldsymbol{u}$), the novel approach unknown in the preceding literature.

Methodology

Statistical properties of D are expressed with respect to its argument $\boldsymbol{x} \in \mathbb{R}^d$. Computation of D usually narrows down to optimization of a function $\boldsymbol{u} \mapsto D(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{X})$, for a multivariate $\boldsymbol{u}$. Thus, the first and main objective is to establish *connections between properties of D as a function of $\boldsymbol{x}$ and as a function of $\boldsymbol{u}$*. This shall be accomplished using the techniques of the sum-of-squares method of proofs, see, *e.g.*, Hopkins (2020), Barak and Steurer (2017). Research shall start with the class of depths that satisfy the projection property (Dyckerhoff, 2004), *i.e.*, for which it holds $D(\boldsymbol{x}|\boldsymbol{X}) = \inf_{\boldsymbol{u}\in\mathbb{S}^{d-1}} D^1(\boldsymbol{x}^\top\boldsymbol{u}|\boldsymbol{X}^\top\boldsymbol{u})$, with $\boldsymbol{X}^\top\boldsymbol{u}$ being a shortcut for observation-wise projection of $\boldsymbol{X}$ on $\boldsymbol{u}$. The univariate depth D^1 is a well-studied function such as distribution function for the halfspace depth or robustified Wald statistic for projection depth. Based on this particular case, generalizations to more complicated depth functions shall be developed.

The second objective is to tackle the *question of optimization*, accompanied by statistical guarantees in particular tractable cases. For purely stochastic search methods, uniform convergence rates were derived for the population version of the halfspace depth in Section 1.5. These suffer from the curse of dimensionality. However, a guided higher-dimensional optimization setting explored in Section 1.4 has illustrated a substantially improved behavior allowing for satisfactory approximation in as few as several hundred optimization steps. A combination of both approaches shall be developed to enable gradient-based optimization of stochastically smoothed $\boldsymbol{u} \mapsto D(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{X})$.

Third, an efficient *implementation as a* `Python` *freely accessible library* shall be built, with the working name `data-depth`. With `Python` being the most used programming language in the area of machine learning and artificial intelligence, this should increase the impact of the developed methods and algorithms, facilitate search for the tool, and reduce the effort of the practitioners.

Fourth, *attention-attracting applications* shall be created based on codes of the developed `Python` library, here are the two examples:

- Large-scale solution for disease cause investigation in the area of *brain imaging* based on DTI scans of the Old Australian Twins Study (OATS) which includes the tasks of curve registration, outlier detection, and statistical comparison of monozygotic and dizygotic twins, for high-dimensional imaging on thousands of individuals.

- Large-scale statistical treatment of multivariate spectra of the *construction materials* is aimed to provide automatized quality control of the rock mining. Outlier and novelty detection for dozens of thousands of these curves shall increase the amount of rock locations being tested, and reduce the cost of the elaborate procedure involving visual inspection by an engineer and experimental testing.

These topics assemble into the project named "*Large-scale data depth: computation and applications*" (acronym LS-Depth-CaP), that has been funded by the Young Researcher Grant of the French National Research Agency (ANR AAPG JCJC 2021) and shall be conducted together with PhD student Jérémy Guérin, who started his PhD thesis under my supervision in April 2022.

5.2 Anomaly detection for large-scale and heterogeneous data of production lines

The main topic of this project is anomaly detection (Chandola et al., 2009)—a branch of machine learning aiming at identification of abnormal, outlying events. Methods of this rapidly developing field usually return a score, by thresholding which normal and abnormal observations are separated. In a number of applications, the anomalies learning process is not supervised completely (Rousseeuw and Hubert, 2018), *i.e.*, it is either semi-supervised or unsupervised at all; this is due to the difficulty to define an anomaly (*e.g.*, an item passing all production tests can still break when in field) and their normally rare occurrence in the data.

Today, anomaly detection applied to industrial production processes faces numerous challenges. First of them is the *large scale* of the acquired data which is due to rapid development of sensors and the interconnectivity of the devices in the Industry 4.0 concept. Second, *heterogeneous data* that arrive in differing numbers of measurements in an asynchronous manner complicates employment of existing generic techniques. Third challenge consists in *interpretability*, *i.e.*, capacity not only to detect abnormalities but as well provide (a hint on) the reason for their occurrence.

Industrial context

The planned project will be closely related to a group of assembly production lines owned by the Valeo enterprise. More precisely, four production lines are considered, namely three lines manufacturing inverter, rotor, and stator, and the fourth one assembling the mentioned components into a single motor. Each production line consists of a number of workstations,

sequentially connected by workflow of passing them items being assembled, and ends with the testing equipment (finalized with labeling and visual inspection for quality items). The data shall be provided on the settings of the workstations, there measured parameters of the passing items, as well as detailed testing results from all the four lines; accompanied with the corresponding tracing time stamps and identifiers, respectively. For this ensemble of lines, techniques of anomaly detection shall be developed to detect defects/deviations on the level of produced items, personnel work shifts, stations' equipment, and to analyze/establish the cause(s) of potential problem(s).

Involved methods

Introduced several decades ago (Tukey, 1975), and having undergone theoretical formalization (Zuo and Serfling, 2000, Mosler, 2013) and recent computational developments (Pokotylo et al., 2019), the concept of the data depth function proves a universal methodology for anomaly detection (Hubert et al., 2015). For any observation, it returns its measure of centrality in a given data set, and thus observations with higher depth values are representative while those possessing low depth are considered as abnormalities. Multiple definitions of data depth applicable not only for multivariate, but also for functional and time-series data (Nieto-Reyes and Battey, 2016), as well as curve data (Lafaye De Micheaux et al., 2022) (useful when treating functional observations with differing duration) transformed it into a versatile tool particularly suitable for anomaly detection.

Another angle to anomaly detection consists in building minimum volume sets for the empirical probability distribution. Then, data points external to these sets are detected as anomalies. Among several minimum level sets estimation approaches, one-class support vector machines (OC-SVM) (Schölkopf et al., 2001) have emerged as one of the most powerful unsupervised methods as witnessed by numerous successful real-world applications during the last decade (Shin et al., 2005). OC-SVM leverages two key ingredients, a relevant loss function and the kernel trick. Kernel-based machine learning methodology (Schölkopf and Smola, 2002) comes as a universal remedy to tackle in a unique framework data of various nature. Kernels which can be thought of as similarities between data allow to handle structured (Gärtner, 2003) or dimension-varying (Song et al., 2013) data, a key feature here since the manufacturing is performed in phases (work shifts) and without any constant time grid—results of input/output that varies in time and dimension.

Material of this section constitutes the project named "*Anomaly detection for large-scale and heterogenous data of production lines*", that has been funded under the Industrial Convention of Education by Research (CIFRE) in collaboration with the global automotive supplier Valeo and shall be conducted together with PhD student—co-supervised with Florence D'Alché-Buc—Romain Valla, whos started his PhD thesis in February 2022.

5.3 Miscellaneous

The multi-scale α-procedure

The first development of my PhD thesis was the $DD\alpha$-classifier, a proper combination of the depth transform and of the α-procedure, published in a cited also today Lange et al. (2014b). Together with a number of depths, the $DD\alpha$-classifier has been implemented in the `R`-package `ddalpha` (Pokotylo et al., 2019). Nevertheless, caged by the data depth, the potential of the α-procedure has not been fully explored there.

The proposal is to embed the α-procedure (Lange and M., 2014)—an iterative heuristic synthesizing the space of relevant features—in a reproducing kernel Hilbert space. To provide a finite-dimensional basis (necessary to run the α-procedure), projections on the training sample are planed to be used (other choices are, of course, possible), which can be orthogonalized by the Gram-Schmidt method employing the so-called kernel trick. Due to built-in robustness (by the indicator-risk nature) and regularization (by the iterative nature), the method proves "reserve against over-fitting". The reverse Gram-Schmidt process shall further allow for a simple multi-scale classification rule of the form, *e.g.* $c(\boldsymbol{x}) = \texttt{sign} \sum_{i=1}^{n} \alpha_i K_{\sigma_i}(\boldsymbol{x}, \boldsymbol{x}_i)$ with $\boldsymbol{x}$ being the observation to be classified, $\{\boldsymbol{x}_i\}_{i=1}^{n}$ being the points form the training sample, and $K_{\sigma_i}(\cdot, \cdot)$ being the kernel with parameter σ_i. (Due to the multi-scale nature only a few coefficients should differ from zero.)

Iterative nature of the α-procedure allows to attack directly the desired risk function not resorting to a surrogate loss. However, the kernel (parameter(s)) should be tuned. The suggestion is thus, to incorporate tuning (*e.g.* by cross-validation (CV)) in each iteration of the α-procedure, thus allowing for multi-scale training, without affecting the complexity compared to global tuning (which is not possible in most existing methods, *e.g.* those based on the support vector machine Cortes and Vapnik, 1995).

Possessing exact computational complexity of $O(n^3 \log n)$ and approximate complexity of $O(n^2 \log n)$ in each step (which can be further reduced using sub-sampling), the method becomes competitive even computationally with the state-of-the-art of the supervised classification. Different to the artificial neural networks, the solution is example-based, explainable, and is potentially inferential. Although the proposal is on an early stage and needs both theoretical and implementation development, there exists an experimental implementation and first results show encouraging performance for high-dimensional data; *e.g.* delivering almost perfect separation in a non-trivial case of 100 points in dimension 100.

Theoretical challenge consists in proving convergence of the classifier, as well as deriving probability bounds—a highly demanded question necessary to deliver reliable solution also by industrial actors. This is planned to be done in three steps: (1) derivation of the theoretical guarantees for the original α-procedure (Lange and M., 2014) in the Euclidean space, a still open question; (2) extending this to the case where the CV-estimated error during tuning is assumed to be exact; (3) development of a full argument for the proposed methodology.

On the probability of general position

Statistical methods usually involve computational procedures, which eventually constitute algorithms, are programmed in software and executed on a computer. During development of such algorithms, the question of numerical precision is rarely taken seriously into account. On the other hand, these statistical methods often depend on—even if very general—assumptions of different nature. If these assumptions, in order, involve numerical precision as well, and are in practice difficultly verifiable, the validity of the promised theoretical results can be questioned.

Here, the assumption of general position (see Definition 1.7) is planned to be studied from this point of view. While the assumption of general position guarantees the theoretical result on outcome of several algorithms, it can be more rarely than thought satisfied in practice, as it shall be seen right below.

For many statistical methods and algorithms, unless especially designed for such conditions, clamping of points and/or their concentration in sub-spaces of $\mathbb{R}^d$ is troublesome. This is usually expressed by assumptions on the results that guarantee the correctness of the outcome, one of the main ones being that $\boldsymbol{X}$ satisfies Definition 1.7. This is especially the case for the literature on the statistical data depth function (Zuo and Serfling, 2000, Mosler, 2013)—the emerging area of numerous theoretical developments and increasing practical importance. Thus, algorithms for computation of the most famous Tukey's halfspace depth (Tukey, 1975) rely (Liu and Zuo, 2014a, Liu, 2017) on or mention (Dyckerhoff and M., 2016) this assumption; it is also a necessary condition for computation of the halfspace trimmed regions (Hallin et al., 2010, Paindaveine and Šiman, 2011, Liu et al., 2019). Further, for the celebrated halfspace depth, this is a necessary condition for certain theoretical results (Donoho and Gasko, 1992, Liu et al., 2020). This also refers to other depth notions: Definition 1.7 is crucial for computation of zonoid trimmed regions (Mosler et al., 2009) as well as the general weighted-mean trimmed regions (Bazovkin and Mosler, 2012), and the projection depth, contours, and median (Liu and Zuo, 2014b).

To proceed, let us first restate the assumption of general position more formally. Consider an ensemble of the in a d-dimensional Euclidean space $\mathbb{R}^d$ which contains n elements (=points or observations) $\mathcal{S}_n = (\boldsymbol{x}_1, ..., \boldsymbol{x}_n)$. The (...) operator is used here to insist on the fact that subsets of identical (or indistinguishable) elements can be included in $\mathcal{S}_n$, and thus strictly speaking it does not fall in to the category of a "set". This is nevertheless very often the case in practice, even for continuous variables, due to the measurement or registration precision, which is directly connected to the *leitmotiv* of the present task. For an arbitrary ensemble $\mathcal{X}$ of points in $\mathbb{R}^d$, denote $\mathrm{AS}(\mathcal{X})$ the affine subspace they engender:

$$\mathrm{AS}(\mathcal{X}) = \Big\{ \sum_{j=1}^{k} \lambda_j \boldsymbol{x}_j \,|\, \boldsymbol{x}_1, ..., \boldsymbol{x}_k \in \mathcal{X}\,,\ \lambda_1, ..., \lambda_k \in \mathbb{R}\,,\ \sum_{j=1}^{k} \lambda_j = 1 \Big\}\,.$$

Denoting by $\#(A)$ the number of elements in the set A, one can define the *general position*.

Definition 5.1 (General position) *An ensemble $\mathcal{S}_n$ is in* general position *if:*

$$\#\big(\mathrm{AS}(\boldsymbol{x}_{j_1}, ..., \boldsymbol{x}_{j_k}) \cap \mathcal{S}_n\big) = k \quad \forall \quad j_1, ..., j_k \subset \{1, ..., n\} \quad \forall \quad k = 1, ..., d\,.$$

Algorithm 5.1 Exact algorithm for checking the ε-numerical general position

function IsInGeneralPosition($d, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_n, \varepsilon$)
  for $k \leftarrow 1$ to d **do**
    for each subset $I \subset \{1, \ldots, n\}$ of cardinality k **do**
      if $k = 1$ **then** ▷ Special treatment for $k = 1$, *i.e.*, single point
        $j \leftarrow$ element of I
        for $i \leftarrow 1$ to n **do**
          if $i \neq j$ and $\|\boldsymbol{x}_i - \boldsymbol{x}_j\| \leq \varepsilon$ **then** ▷ Not the same point but too close
            return False
      else
        $\boldsymbol{A} \leftarrow [\boldsymbol{a}_1 \ \ldots \ \boldsymbol{a}_l] =$ the (column) orthogonal normal basis of $\mathrm{AS}\big((\boldsymbol{x}_i)_{i\in I}\big)$
        if $l < k - 1$ **then** ▷ $(\boldsymbol{x}_i)_{i\in I}$ not linearly independent
          return False
        for $i \leftarrow 1$ to n **do**
          if $i \notin I$ and $\|\boldsymbol{x}_i - \boldsymbol{A}\boldsymbol{A}^\top \boldsymbol{x}_i\| \leq \varepsilon$ **then** ▷ Too close to I's subspace
            return False
  return True

The question to be answered is the following: For a data set $\mathcal{S}_n$ (consisting of n observations) generated from a known (or suspected) distribution in $\mathbb{R}^d$, what is the probability that it is in ε-numerical general position?

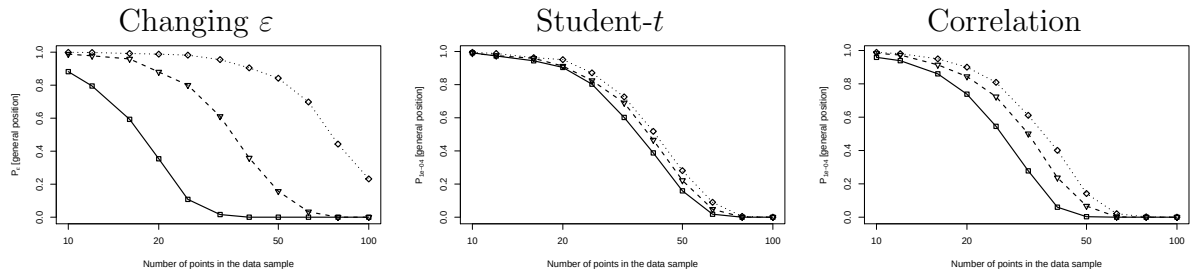
Let us start by developing simple algorithmic foundations which allow for checking whether a sample $\mathcal{S}_n$ is in general position provided precision ε, estimation of the probability of general position by a Monte Carlo simulation, and as a consequence verification of the approximation strategies to be developed.

Basic computations are gathered in Algorithm 5.1 that tests $\mathcal{S}_n$ for the ε-numerical general position based on the following idea: For each subset of $I \subset \{1, \ldots, n\}$ of cardinality $\#I = k \in \{1, \ldots, d\}$, one compares to ε the Euclidean distance from each point (except those $\in (\boldsymbol{x}_i)_{i\in I}$) to (its projection on) $\mathrm{AS}\big((\boldsymbol{x}_i)_{i\in I}\big)$. Provided an orthogonal normal basis $\boldsymbol{A}$ of $\mathrm{AS}\big((\boldsymbol{x}_i)_{i\in I}\big)$, for an arbitrary point $\boldsymbol{x} \in \mathbb{R}^d$, this distance can be calculated as $\|\boldsymbol{x}_i - \boldsymbol{A}\boldsymbol{A}^\top \boldsymbol{x}_i\|$ with $\|\cdot\|$ standing for the Euclidean norm. Clearly, to assure the correctness of Algorithm 5.1 the numerical precision of the resting routines, such as basis computation or matrix multiplication, should be order(s) higher than ε, with the preference for ε^2.

The time complexity of Algorithm 5.1 is dominated by the number of combinations of k points out of n and thus totals to $O(d^3 n^{d+1})$. On the other hand, early stop criterion can be applied as long as deviation from the ε-numerical general position is found for at least one point and one affine subspace engendered by k points from $\mathcal{S}_n$.

Algorithm 5.1 allows to gain first insights into the requested probability of the general position assumption; see Figure 5.1.

Algorithm 5.1 is simple and is presented only for verification purposes: although it allows for a very precise Monte Carlo estimation of this probability for any distribution



(a) (b) (c)

Figure 5.1: *Empirically approximated (over* 1000 *repetitions) probability of the ε-numerical general position as a function of the size of the bivariate data sample $\mathcal{S}_n$ for n varying from* 10 *to* 100*. (a): For the standard Gaussian distribution with ε equal to 10^{-3} (solid line),* 10^{-4} *(dased line), and* 10^{-5} *(dotted line). (b): For Gaussian (solid line), Student-t_2 (dashed line), and Cauchy (dotted line) distributions without correlation; $\varepsilon = 10^{-4}$. (c): For a Gaussian distribution with variance* 1 *on both axes and correlation equal to* 0.95 *(solid line),* 0.75 *(dashed line), and without correlation (dotted line); $\varepsilon = 10^{-4}$.*

from which observations can be generated, such an estimation is very time consuming, especially with increasing space dimension and/or sample size. *E.g.*, for the trivial case with $d = 5, n = 75, \varepsilon = 10^{-8}$, single execution of Algorithm 5.1 takes (on an average) around 12 seconds if the data are in general position and around 6 seconds if they are not, an thus an estimation based on 1000 repetitive draws would take up to two hours (measured on a single core of the 3.5 `GHz` Dual-Core Intel Core i7 processor).

Thus, to answer this question in a reasonable time, for real data sets and first examples of distributions, either an analytical tool or a precise approximating algorithm should be developed.

5.4 Final word

Learning from data has illustrated uncountable successes in the past decades and became a general methodology for knowledge extraction from real-world observations. Its approaches rely profoundly on providing a meaningful ordering of data. In unsupervised machine learning, data ordering uncovers the structure of raw, unlabeled data, *e.g.*, by data clustering or anomaly detection. It measures degree of adherence to a class in the supervised learning. In data analysis, data ordering explores the geometry of data, allows for their relevant visualization, estimation of location and scatter, and statistical inference. Intrinsically based on data ordering, cumulative distribution function, quantiles, and ranks are ubiquitous for presentation of data or summary of analysis and crucial for definition of losses in machine learning models.

Distinguishable from existing methods, data depth defines centrality-based order, and therefore extends median, quantiles, ranks, and outliers to higher dimensions, and eventually to more complex data. It is nonparametric and fully data driven, and thus does not assume

a data generating process. It possesses attractive asymptotic and finite-sample properties (*e.g.*, parametric rates) and gives rise to statistical inference. Definitions of data depth are robust, explicit and easy to interpret. By dint of these advantages, depth is used in a variety of tasks across domains and—though on small scale—has already been applied to contemporary challenges

For data depth in particular and in statistics in general, computational constituent plays an important role in its sustainable development. To survive practitioner's challenge, and be used during a period of time, statistical methods shall be accompanied with (theoretically or heuristically founded) computing algorithms, them implementing software packages, and attention-attracting application examples (preferably based on the real-world data). Moreover, efficient implementation (developed with good understanding of the installed operating system and architecture), though (usually) does not change algorithmic complexity itself, can substantially reduce its constants, and thus push further the limits of computational intractability. While correctness of the results of both methods and algorithms can be conditional on assumptions, these should take into account—in the best case and if this is possible—machine precision as well.

On the other hand, it becomes quickly clear that satisfying certain ensembles of statistical requirements (*e.g.*, nonparametricity, robustness, demanded invariance(s), fast convergence rates, *etc.*) can be computationally costly: halfspace depth being one of the main topics of this manuscript is an example. This of course does not mean that the task shall be abandoned and compromises can be found. If the data set is not very large, and/or computational resources/time are available, computation can be still performed, with respect to the trade-off between task importance and ecological consequences. Should one refuse using the statistical method of choice if this is not the case? One way to nevertheless keep on is to use approximate computation, searching for a compromise between precision and calculation time; this may affect statistical properties as well. Another way—to still stick to exact computing—consists in weakening statistical requirements to the method. One can hope, that providing a general-purpose framework for this aim would help statistician/practitioner to target better the application task.

Appendix A

A note on implementations

All of the mentioned developed methods described in this manuscript possess implementations. Below, these are briefly mentioned, with a section per software. Section A.3 provides information about the fast procedures implemented for nonparametric frontier analysis (in `Stata` and `R`), which are closely connected in logic and source codes particularly to Section 1.3. Two articles resulted from these programs: Pokotylo et al. (2019) (corresponding to Section A.1 below) and Badunenko and M. (2016) (corresponding to Section A.3 below).

A.1 `R`-package `ddalpha`

Having undergone theoretical and computational developments, data depth is today employed in numerous applications with classification being a popular one. The `R` package `ddalpha` (Pokotylo et al., 2020) is a software directed to fuse experience of the applicant with recent achievements in the area of data depth and depth-based classification.

`ddalpha` provides an implementation for exact and approximate computation of most known and widely applied notions of data depth. These can be further used in the depth-based multivariate and functional classifiers implemented in the package, where the $DD\alpha$-procedure (Lange et al., 2014b, M., 2015) is in the main focus. The package is expandable with user-defined custom depth methods and separators. The implemented functions for depth visualization and the built-in benchmark procedures may also serve to provide insights into the geometry of the data and the quality of pattern recognition.

Today, the `R`-package `ddalpha` contains procedures for exact and approximate computation of 11 notions of multivariate data depth function, and versatile tools for depth-based supervised learning, including necessary visualization, diagnosis, and inference. It further offers an extension to functional data and routines for calculating certain notions of statistical depth functions. Most of the functions of the package are programmed in `C++`, in order to be fast and efficient. Moreover, 50 multivariate and 5 functional ready-to-use classification data sets are included.

A comprehensive practical guide for the `R`-package `ddalpha` was published as Pokotylo et al. (2019).

A.2 R-package TukeyRegion

Tukey (or halfspace) regions are polytopes in the Euclidean space, *viz.* upper-level sets of the halfspace depth function on given data. Using implementation of the algorithms of Section 1.3, in R-package `TukeyRegion` (M. and Barber, 2021) the bordering hyperplanes of a Tukey region can be computed, as well as its vertices, facets, centroid, and volume. In addition, the Tukey median set, which is the non-empty Tukey region having highest depth level, and its barycenter (= Tukey median) are calculated. Tukey regions are visualized in dimension two and three.

A.3 Stata commands for non-parametric frontier analysis and R-package npsf

The concept of efficiency is at the core of production economics. Beginning with the pioneering work by Cobb and Douglas (1928), there were many attempts to parametrize the production process: *e.g.*, Leontieff, constant elasticity of substitution, transcendental logarithmic production and cost functions. Conceptually, however, researchers looked at the "average" input-output relationship assuming no inefficiency. Yet, it was no longer plausible to assume that all units are homogeneous, that is, operating at the same level of efficiency. Among the first to offer an appropriate modification was Farrell (1957), who built up on the concept of efficiency postulated by Koopmans (1951) and Debreu (1951) and put forward a foundation, which has become a distinct field in economics—the efficiency analysis. Färe (1988), Färe et al. (1994), Färe and Primont (1995) provide many insights into nonparametric efficiency measurement.

Data envelopment analysis (DEA), a leading analytical technique for measuring relative efficiency, has been widely used by both academic researchers and practitioners in evaluating the efficiency of decision making units in terms of converting inputs into outputs. Researchers choose this technique because it does not impose *a priori* functional form and allows for multiple output technologies.

Although the DEA method is typically considered to be deterministic, the efficiency is still computed relatively to estimated and not true frontier. The efficiency scores obtained from a finite sample are subject to sampling variation of the estimated frontier. Simar and Wilson (1998, 2000, 2002) have laid out a statistical model and proposed consistent bootstrap procedures to provide statistical inference regarding technical efficiency measures in nonparametric frontier models.

Efficiency analysis

Implemented measures of technical efficiency for the production data points are conventional radial Debreu-Farrell measures of efficiency loss (Debreu, 1951, Farrell, 1957). Due to the use of several spaces, a change of notation is necessary. For each data point k ($k = 1, \ldots, K$) vector $\boldsymbol{x}_k = (x_{k1}, \ldots, x_{kN}) \in \mathbb{R}^N$ denotes N inputs, vector $\boldsymbol{y}_k = (y_{k1}, \ldots, y_{kM}) \in \mathbb{R}^M$

denotes M outputs. It is assumed that under technology T the data $(\boldsymbol{y}, \boldsymbol{x})$ are such that outputs are producible by inputs,

$$T = \{(\boldsymbol{x}, \boldsymbol{y}) : \boldsymbol{y} \text{ are producible by } \boldsymbol{x}\}. \tag{A.1}$$

The technology is fully characterized by its production possibility set,

$$P(\boldsymbol{x}) \equiv \{\boldsymbol{y} : (\boldsymbol{x}, \boldsymbol{y}) \in T\} \tag{A.2}$$

or input requirement set,

$$L(\boldsymbol{y}) \equiv \{\boldsymbol{x} : (\boldsymbol{x}, \boldsymbol{y}) \in T\}. \tag{A.3}$$

Conditions (A.2) and (A.3) imply that the available outputs and inputs are feasible. The upper boundary of the production possibility set and lower boundary of the input requirement set define the frontier. How far is a given data point from the frontier represents its efficiency. In output-based radial efficiency measurement, the amount of necessary (proportional) expansion of outputs to move a data point to a boundary of the production possibility set $P(\boldsymbol{x})$ serves as a measure of technical efficiency. In input-based radial efficiency measurement, it is the amount of necessary (proportional) reduction of inputs to move a data point to a boundary of the input requirement set $L(\boldsymbol{y})$.

Empirically, technical efficiencies are estimated via activity analysis models. For K data points, each possessing M outputs and N inputs, an estimate of the radial Debreu-Farrell output-based measure of technical efficiency can be calculated by solving a linear programming problem for each data point l $(l = 1, \ldots, K)$:

$$\begin{aligned} \hat{F}_l^o(\boldsymbol{y}_l, \boldsymbol{x}_l, \mathcal{Y}, \mathcal{X}|\text{CRS}) = \max_{\boldsymbol{\theta}, \boldsymbol{\lambda}} \boldsymbol{\theta} \\ \text{s.t.} \quad \sum_{k=1}^{K} \lambda_k y_{km} \geq y_{lm}\theta_m, m = 1, \cdots, M, \\ \sum_{k=1}^{K} \lambda_k x_{kn} \leq x_{ln}, n = 1, \cdots, N, \\ \lambda_k \geq 0. \end{aligned} \tag{A.4}$$

$\mathcal{Y}$ is $K \times M$ matrix of available data on outputs $\mathcal{Y} = (\boldsymbol{y}_1, \ldots, \boldsymbol{y}_K)^\top$, $\mathcal{X}$ is $K \times N$ matrix of available data on inputs $\mathcal{X} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_K)^\top$. The estimate of $P(\boldsymbol{x})$ is the smallest convex free-disposal hull that envelops the observed data, and upper boundary of which is a piece-wise linear estimate of the true best-practice frontier of $P(\boldsymbol{x})$. Equation (A.4) defines constant returns to scale (CRS) specification. Other returns to scale are modeled by adjusting process operating levels λ_k's; for variable returns to scale (VRS) a convexity constraint $\sum_{k=1}^{K} \lambda_k = 1$ is added[A.1], while for non-increasing returns to scale (NIRS), $\sum_{k=1}^{K} \lambda_k \leq 1$ inequality is added[A.2] to set of restrictions of linear programming problem in equation (A.4).

[A.1] This equality ensures that data point l is compared only to data points of similar size; under CRS assumption, data points of different sizes might be compared to one another.

[A.2] This inequality ensures that data point l is not compared to other data points that are considerably larger, but maybe compared to smaller data points.

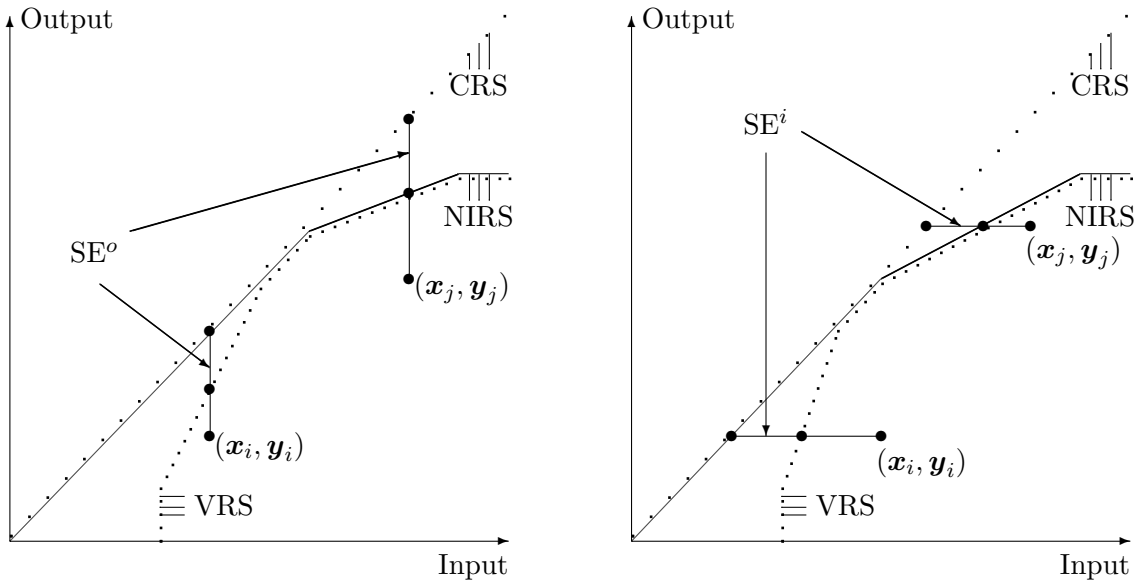


Figure A.1: *Output-based (left) and input-based (right) technical and scale efficiency.*

To facilitate the discussion, Figure A.1 presents hypothetical one-input one-output production process with three different technologies CRS, VRS and NIRS. Conceptually, in Figure A.1, left (right) the vertical (horizontal) distance from a data point $(\boldsymbol{x}_i, \boldsymbol{y}_i)$ or $(\boldsymbol{x}_j, \boldsymbol{y}_j)$ to CRS/VRS/NIRS best-practice frontier stands for output-based (input-based) technical efficiency under assumption of CRS/VRS/NIRS technology. In a multi-dimensional case, the required distance is the radial path from a data point that is parallel to axes along which all outputs (inputs) are measured.

For data point $(\boldsymbol{y}_l, \boldsymbol{x}_l)$, radial measure expands (shrinks) all M outputs $\boldsymbol{y}_l = (y_{l1}, \ldots, y_{lM})$ (N inputs $\boldsymbol{x}_l = (x_{l1}, \ldots, x_{lN})$) proportionally until the frontier is reached. At the reached frontier point, some but not all outputs (inputs) can therefore be expanded (shrinked) while remaining feasible. Nonradial efficiency measures compensate for this by allowing outputs (inputs) to change disproportionately, while frontier restrictions are still to be satisfied. For more details on its difference from the radial efficiency measure, the reader is invited to consult Section 2.2 of Badunenko and M. (2016).

In Badunenko and M. (2016), five new `Stata` commands are introduced, that estimate and provide statistical inference in nonparametric frontier models. First two commands, `tenonradial` and `teradial`, estimate data envelopment models where technical efficiency measures are computed (Färe, 1988, Färe and Lovell, 1994, Färe et al., 1994). These are the core `Stata` commands for efficiency calculation, and are used in the three following ones. Technical efficiency measures are obtained by solving linear programming problems. The rest of the commands, `teradialbc`, `nptestind`, and `nptestrts`, give tools for making statistical inference regarding radial technical efficiency measures (Simar and Wilson, 1998, 2000, 2002). The main computational advantage is gained by employing the step of first computing the convex hull (using the procedure by Barber et al., 1996) to reduce the constraints of the linear problem, when this is necessary. The article (Badunenko and M., 2016)

provides brief overview of the nonparametric efficiency measurement (Section 2), as well as the description of syntax and options of new commands (Sections 3 to 7). Additionally, an example showing the capabilities of new commands is provided. Finally, a small empirical study of productivity growth is performed (Section 8).

Further, R-package `npsf` (Badunenko et al., 2020) implements the same functionality in R and is continuously developed and maintained.

Moreover, an extensive simulation study regarding inference for nonradial efficiency measurement can be found in Badunenko and M. (2020).

A.4 R-package `imputeDepth`

The R-package `imputeDepth` can be found in the following GitHub repository: https://github.com/pavlomozharovskyi/imputeDepth. It provides implementation of the material of Chapter 2. More precisely, `imputeDepth` implements functions for single imputation of missing data using 5 notions of multivariate depth function: Mahalanobis, zonoid, halfspace, projection and spatial depth. Several outsider-treating procedures are provided as well. Further, procedures for imputation using local depth (according to the notion by Paindaveine and Van Bever, 2013) and multiple imputation (for elliptically-symmetric distributions), as well as those for "improper imputation" are included.

A.5 R-package `curveDepth`

The R-package `curveDepth` (M. et al., 2019) implements the developments of Chapter 3, which are ready for practical use. The Monte-Carlo estimation procedure is implemented using both exact and approximate versions for the point curve depth. It is noteworthy, that these implementations owe much to Sections 1.2 and 1.4. More precisely, the Tukey curve depth is implemented, as well as it's two-stage version to allow for maximal control over the estimation process. Further, implementation of the Fréchet metric used on the space of curves is provided. Finally, the voxelization procedure as well as the "0"-"1"-"7" part of the MNIST data set (mentioned in Sections 6.2.1 and 6.2.2 of Lafaye De Micheaux et al., 2022) [A.3] are included with an example of pre-processing. It is expected, that R-package `curveDepth` provides ready-to-use tools for conducting data analysis for (unparametrized) curve data.

[A.3] For a comprehensive reference on the data, see http://yann.lecun.com/exdb/mnist/.

Bibliography

Adler, D., D. Murdoch, et al. (2021). *rgl: 3D Visualization Using OpenGL*. R package version 0.108.3, https://CRAN.R-project.org/package=rgl.

Aizenman, M. and A. Burchard (1999). Hölder regularity and dimension bounds for random curves. *Duke Mathematical Journal 99*(3), 419–453.

Azzalini, A. (2013). *The Skew-Normal and Related Families*. Institute of Mathematical Statistics Monographs. Cambridge University Press.

Azzalini, A. and A. Capitanio (1999). Statistical applications of the multivariate skew normal distribution. *Journal of the Royal Statistical Society: Series B (Methodological) 61*, 579–602.

Badunenko, O. and P. M. (2016). Nonparametric frontier analysis using stata. *Stata Journal 16*(3), 550–589.

Badunenko, O. and P. M. (2020). Statistical inference for the russel measure of technical efficiency. *Journal of the Operational Research Society 71*, 517–527.

Badunenko, O., P. M., and Y. Kolomiytseva (2020). *npsf: Nonparametric and Stochastic Efficiency and Productivity Analysis*. R package version 0.8.0, https://CRAN.R-project.org/package=npsf.

Barak, B. and D. Steurer (2017). *Proofs, Beliefs and Algorithms Through the Lens of Sum of Squares*. https://www.sumofsquares.org/public/lec-definitions-general.html.

Barber, C. B., D. P. Dobkin, and H. Huhdanpaa (1996). The quickhull algorithm for convex hulls. *ACM Transactions on Mathematical Software 22*(4), 469–483.

Bazovkin, P. and K. Mosler (2012). An exact algorithm for weighted-mean trimmed regions in any dimension. *Journal of Statistical Software 47*(13), 1–29.

Bazovkin, P. and K. Mosler (2015). A general solution for robust linear programs with distortion risk constraints. *Annals of Operations Research 229*(1), 103–120.

Beaudouin, V., I. Bloch, D. Bounie, S. Clémençon, F. D'Alché-Buc, W. Eagan, J. nad Maxwell, P. M., and J. Parekh (2020a). Flexible and context-specific ai explainability: A multidisciplinary approach. *arXiv:2003.07703*.

Beaudouin, V., I. Bloch, D. Bounie, S. Clémençon, F. D'Alché-Buc, W. Eagan, J. nad Maxwell, P. M., and J. Parekh (2020b). Identifying the "right" level of explanation in a given situation. In A. Saffiotti, L. Serafini, and P. Lukowicz (Eds.), *Proceedings of the First International Workshop on New Foundations for Human-Centered AI (NeHuAI 2020 with ECAI 2020)*, pp. 63–66.

Bernholt, T. (2006). Robust estimators are hard to compute. Technical report, Dortmund University.

Bertsekas, P. D. (1999). *Nonlinear programming. Second edition.* MIT Press.

Boček, P. and M. Šiman (2016). Directional quantile regression in octave (and matlab). *Kybernetika 52*(1), 28–51.

Bremner, D., D. Chen, J. Iacono, S. Langerman, and P. Morin (2008). Output-sensitive algorithms for Tukey depth and related problems. *Statistics and Computing 18*, 259–266.

Brys, G., M. Hubert, and A. Struyf (2004). A robust measure of skewness. *Journal of Computational and Graphical Statistics 13*(4), 996–1017.

Burago, D., Y. Burago, and S. Ivanov (2001). *A Course in Metric Geometry*, Volume 33 of *Graduate Studies in Mathematics*. American Mathematical Society, Providence, RI.

Burr, M. A. and R. J. Fabrizio (2017). Uniform convergence rates for halfspace depth. *Statistics and Probability Letters 124*, 33–40.

Cascos, I. (2009). Data depth: multivariate statistics and geometry. In W. S. Kendall and I. Molchanov (Eds.), *New Perspectives in Stochastic Geometry*. Oxford: Oxford University Press.

Cascos, I. and I. Molchanov (2007). Multivariate risks and depth-trimmed regions. *Finance and Stochastics 11*(3), 373–397.

Chakraborty, A. and P. Chaudhuri (2014). The spatial distribution in infinite dimensional spaces and related quantiles and depths. *The Annals of Statistics 42*(3), 1203–1231.

Chandola, V., A. Banerjee, and V. Kumar (2009). Anomaly detection: A survey. *ACM Computing Surveys 41*(3), 1–58.

Chen, D., P. Morin, and U. Wagner (2013). Absolute approximation of Tukey depth: Theory and experiments. *Computational Geometry: Theory and Applications 46*, 566–573.

Chernozhukov, V., A. Galichon, M. Hallin, and M. Henry (2017). Monge–kantorovich depth, quantiles, ranks and signs. *The Annals of Statistics 45*(1), 223–256.

Claeskens, G., M. Hubert, L. Slaets, and K. Vakili (2014). Multivariate functional halfspace depth. *Journal of the American Statistical Association 109*(505), 411–423.

Clayden, J., M. Modat, B. Presles, T. Anthopoulos, and P. Daga (2020). *RNiftyReg: Image Registration Using the 'NiftyReg' Library*. R package version 2.7.0, https://CRAN.R-project.org/package=RNiftyReg.

Cobb, C. W. and P. H. Douglas (1928). A theory of production. *American Economic Review,* Supplement *18*(1), 139–165.

Collet, J.-P., H. Shuman, R. E. Ledger, S. Lee, and J. W. Weisel (2005). The elasticity of an individual fibrin fiber in a clot. *Proceedings of the National Academy of Sciences of the United States of America 102*(26), 9133–9137.

Cortes, C. and V. Vapnik (1995). Support vector networks. *Machine Learning 20*, 273–297.

Cuesta-Albertos, J. and A. Nieto-Reyes (2008). The random Tukey depth. *Computational Statistics and Data Analysis 52*, 4979–4988.

Cuevas, A., M. Febrero, and R. Fraiman (2007). Robust estimation and classification for functional data via projection-based depth notions. *Computational Statistics 22*(3), 481–496.

Dau, H. A., E. Keogh, K. Kamgar, C.-C. M. Yeh, Y. Zhu, S. Gharghabi, C. A. Ratanamahatana, Y. Chen, B. Hu, N. Begum, A. Bagnall, A. Mueen, G. Batista, and Hexagon-ML (2018, October). The ucr time series classification archive. https://www.cs.ucr.edu/~eamonn/time_series_data_2018/.

Debreu, G. (1951). The coefficient of resource utilization. *Econometrica 19*, 273–292.

Dempster, A. P., N. M. Laird, and D. B. Rubin (1977). Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society. Series B (Methodological) 39*(1), 1–38.

DeVore, R. A. and G. G. Lorentz (1993). *Constructive Approximation*, Volume 303 of *Grundlehren der mathematischen Wissenschaften*. Springer-Verlag.

Donoho, D. (1982). *Breakdown Properties of Multivariate Location Estimators*. Ph. D. thesis, Harvard University.

Donoho, D. L. and M. Gasko (1992). Breakdown properties of location estimates based on halfspace depth and projected outlyingness. *The Annals of Statistics 20*(4), 1803–1827.

Dua, D. and C. Graff (2017). UCI machine learning repository. http://archive.ics.uci.edu/ml.

Dutta, S. and A. K. Ghosh (2012). On robust classification using projection depth. *Annals of the Institute of Statistical Mathematics 64*, 657–676.

Dutta, S., A. K. Ghosh, and P. Chaudhuri (2011). Some intriguing properties of Tukey's half-space depth. *Bernoulli 17*(4), 1420–1434.

Dutta, S., S. Sarkar, and A. K. Ghosh (2016). Multi-scale classification using localized spatial depth. *Journal of Machine Learning Research 17*, 1–30.

Dyckerhoff, R. (2002). Datentiefe: Begriff, Berechnung, Tests. Mimeo, Fakultät für Wirtschafts-und Sozialwissenschaften, Universität zu Köln.

Dyckerhoff, R. (2004). Data depths satisfying the projection property. *Allgemeines Statistisches Archiv 88*(2), 163–190.

Dyckerhoff, R., G. Koshevoy, and K. Mosler (1996). Zonoid data depth: Theory and computation. In A. Prat (Ed.), *COMPSTAT '96 – Proceedings in Computational Statistics*, Heidelberg, pp. 235–240. Physica-Verlag.

Dyckerhoff, R. and P. M. (2016). Exact computation of the halfspace depth. *Computational Statistics and Data Analysis 98*, 19–30.

Dyckerhoff, R., P. M., and S. Nagy (2021). Approximate computation of projection depths. *Computational Statistics and Data Analysis 157*, 107166.

Dyckerhoff, R. and K. Mosler (2011). Weighted-mean trimming of multivariate data. *Journal of Multivariate Analysis, 102*(3), 405–421.

Edelsbrunner, H. (1987). *Algorithms in Combinatorial Geometry*. Berlin, Heidelberg: Springer.

Einmahl, J. and D. Mason (1992). Generalized quantile process. *The Annals of Statistics 20*, 1062–1078.

Einmahl, J. H. J., J. Li, and R. Y. Liu (2015). Bridging centrality and extremity: Refining empirical data depth using extreme value statistics. *The Annals of Statistics 43*(6), 2738–2765.

Elmore, R. T., T. P. Hettmansperger, and F. Xuan (2006). Spherical data depth and a multivariate median. In R. Y. Lui, R. Serfling, and D. L. Souvaine (Eds.), *Data Depth: Robust Multivariate Anaysis, Computational Geometry and Applications*, Volume 72 of *Series in Discrete Mathematics and Theoretical Computer Science (DIMACS)*, pp. 87–102. Providence, Rhode Island: American Mathematical Society.

Enqvist, E. (1978). *On sampling from sets of random variables with application to incomplete U-statistics*. Ph. D. thesis, Lund University.

Esbensen, K., D. Guyot, F. Westad, and L. Houmoller (2002). *Multivariate Data Analysis: In Practice : an Introduction to Multivariate Data Analysis and Experimental Design*. CAMO.

Fang, K., S. Kotz, and K. Ng (1990). *Symmetric multivariate and related distributions*. Monographs on statistics and applied probability. Chapman and Hall.

Färe, R. (1988). *Fundumentals of Production Theory*. Berlin: Springer.

Färe, R., S. Grosskopf, and C. A. K. Lovell (1994). *Production Frontiers*. Cambridge, U.K.: Cambridge University Press.

Färe, R. and C. A. K. Lovell (1994). Measuring the technical efficiency of production. *Journal of Economic Theory 19*, 150–162.

Färe, R. and D. Primont (1995). *Multi-Output Production and Duality, Theory and Applications*. Boston: Kluwer Academic Publishers.

Farrell, M. J. (1957). The measurement of productive efficiency. *Journal of the Royal Statistical Society. Series A (General) 120*(3), 253–290.

Febrero-Bande, M. and M. Oviedo de la Fuente (2012). Statistical computing in functional data analysis: The R package fda.usc. *Journal of Statistical Software 51*(4), 1–28.

Fischer, D., K. Mosler, J. Möttönen, K. Nordhausen, O. Pokotylo, and D. Vogel (2020). Computing the oja median in R: The package OjaNP. *Journal of Statistical Software 92*(8), 1–36.

Fojtík, V., P. Laketa, P. Mozharovskyi, and S. Nagy (2022). On exact computation of Tukey depth central regions. *arXiv:2208.04587*.

Fraiman, R. and G. Muniz (2001). Trimmed means for functional data. *Test 10*(2), 419–440.

Gärtner, T. (2003). A survey of kernels for structured data. *ACM SIGKDD Explorations Newsletter 5*(1), 49–58.

Genest, M., J.-C. Masse, and J.-F. Plante (2019). *depth: Nonparametric Depth Functions for Multivariate Analysis*. R package version 2.1-1.1, https://CRAN.R-project.org/package=depth.

Ghosh, A. K. and P. Chaudhuri (2005). On data depth and distribution free discriminant analysis using separating surfaces. *Bernoulli 11*, 1–27.

Gijbels, I. and S. Nagy (2017). On a general definition of depth for functional data. *Statistical Science 32*(4), 630–639.

Goibert, M., S. Clémençon, E. Irurozki, and P. M. (2022). Statistical depth functions for ranking distributions: definitions, statistical learning and applications. In G. Camps-Valls, F. J. R. Ruiz, and I. Valera (Eds.), *Proceedings of The Twenty Fifth International Conference on Artificial Intelligence and Statistics (AISTATS 2022)*, Volume 151, pp. 10376–10406.

Habel, K., R. Grasman, R. B. Gramacy, P. M., and D. C. Sterratt (2019). *geometry: Mesh Generation and Surface Tessellation*. R package version 0.4.5, https://CRAN.R-project.org/package=geometry.

Hallin, M., E. Del Barrio, J. Cuesta-Albertos, and C. Matrán (2021). Distribution and quantile functions, ranks and signs in dimension d: A measure transportation approach. *The Annals of Statistics 49*(2), 1139–1165.

Hallin, M., D. Paindaveine, and M. Šiman (2010). Multivariate quantiles and multiple-output regression quantiles: From l_1 optimization to halfspace depth. *The Annals of Statistics 38*(2), 635–669.

Hariri, S., M. C. Kind, and R. J. Brunner (2021). Extended isolation forest. *IEEE Transactions on Knowledge and Data Engineering 33*(4), 1479–1489.

Hastie, T., R. Mazumder, D. J. Lee, and R. Zadeh (2015). Matrix completion and low-rank svd via fast alternating least squares. *Journal of Machine Learning Research 16*, 3367–3402.

Hastie, T. and W. Stuetzle (1989). Principal curves. *Journal of the American Statistical Association 84*(406), 502–516.

Hopkins, S. B. (2020). Mean estimation with sub-gaussian rates in polynomial time. *The Annals of Statistics 48*, 1193–1213.

Hubert, M., P. J. Rousseeuw, and P. Segaert (2015). Multivariate functional outlier detection. *Statistical Methods and Applications 24*(2), 177–202.

Ieva, F. and A. Paganoni (2013). Depth measures for multivariate functional data. *Communications in Statistics: Theory and Methods 41*, 1265–1276.

Jeong, M.-H., Y. Cai, C. J. Sullivan, and S. Wang (2016). Data depth based clustering analysis. In M. Ali, S. Newsam, S. Ravada, M. Renz, and G. Trajcevski (Eds.), *SIGSPACIAL '16: Proceedings of the 24th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, Volume 29, New York, pp. 1–10. Association for Computing Machinery.

Johnson, D. S. and F. P. Preparata (1978). The densest hemisphere problem. *Theoretical Computer Science 6*, 93–107.

Jörnsten, R. (2004). Clustering and classification based on the l_1 data depth. *Journal of Multivariate Analysis 90*(1), 67–89.

Josse, J. and F. Husson (2012). Handling missing values in exploratory multivariate data analysis methods. *Journal de la Société Française de Statistique 153*(2), 79–99.

Josse, J. and J. P. Reiter (2018). Introduction to the special section on missing data. *Statistical Science 33*(2), 139–141.

Kemppainen, A. and S. Smirnov (2017). Random curves, scaling limits and Loewner evolutions. *The Annals of Probability 45*(2), 698–779.

Kleindessner, M. and U. Von Luxburg (2017). Lens depth function and k-relative neighborhood graph: versatile tools for ordinal data analysis. *Journal of Machine Learning Research 18*, 1889–1940.

Koenker, R. and G. Basset (1978). Regression quantiles. *Econometrica 46*, 33–50.

Kong, L. and I. Mizera (2012). Quantile tomography: using quantiles with multivariate data. *Statistica Sinica 22*(4), 1589–1610.

Koopmans, T. C. (1951). An analysis of production as an efficient combination of activities. In T. C. Koopmans (Ed.), *Activity Analysis of Production and Allocation*. New York: Wiley.

Koshevoy, G. and K. Mosler (1997). Zonoid trimming for multivariate distributions. *The Annals of Statistics 25*, 1998–2017.

Kosiorowski, D. and Z. Zawadzki (2019). Depthproc: An R package for robust exploration of multidimensional economic phenomena. *arXiv:1408.4542*.

Lafaye De Micheaux, P., P. M., and M. Vimond (2022). Depth for curve data and applications. *Journal of the American Statistical Association 116*(536), 1881–1897.

Lange, T. and P. M. (2014). The alpha-procedure: a nonparametric invariant method for automatic classification of multi-dimensional objects. In M. Spiliopoulou, L. Schmidt-Thieme, and R. Janning (Eds.), *Data Analysis, Machine Learning and Knowledge Discovery*, Berlin, pp. 79–86. Springer.

Lange, T., K. Mosler, and P. M. (2014a). $DD\alpha$-classification of asymmetric and fat-tailed data. In M. Spiliopoulou, L. Schmidt-Thieme, and R. Janning (Eds.), *Data Analysis, Machine Learning and Knowledge Discovery*, Berlin, pp. 71–78. Springer.

Lange, T., K. Mosler, and P. M. (2014b). Fast nonparametric classification based on data depth. *Statistical Papers 55*(1), 49–69.

Larsen, F., F. Berg, and S. Engelsen (2006). An exploratory chemometric study of h nmr spectra of table wine. *Journal of Chemometrics 20*, 198–208.

LeCun, Y., L. Bottou, Y. Bengio, and P. Haffner (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE 86*(11), 2278–2324.

Lee, A. J. (1990). *U-statistics: Theory and practice*. New York: Marcel Dekker, Inc.

Lehmann, E. and H. D'Abrera (1975). *Nonparametrics: Statistical Methods Based on Ranks*. Holden-Day series in probability and statistics. Holden-Day.

Li, J., J. A. Cuesta-Albertos, and R. Y. Liu (2012). DD-classifier: Nonparametric classification procedure based on DD-plot. *Journal of the American Statistical Association 107*, 737–753.

Little, R. and D. Rubin (2014). *Statistical Analysis with Missing Data (2nd Edition)*. Wiley.

Liu, F. T., K. M. Ting, and Z.-H. Zhou (2008). Isolation forest. In *2008 Eighth IEEE International Conference on Data Mining*, pp. 413–422.

Liu, F. T., K. M. Ting, and Z.-H. Zhou (2012). Isolation-based anomaly detection. *ACM Transactions on Knowledge Discovery from Data 6*(1), 1–39.

Liu, R. Y. (1990). On a notion of data depth based on random simplices. *The Annals of Statistics 18*(1), 405–414.

Liu, R. Y. (1992). Data depth and multivariate rank tests. In Y. Dodge (Ed.), L_1-*Statistics Analysis and Related Methods*, pp. 279–294. Amsterdam: North-Holland.

Liu, R. Y., J. M. Parelius, and K. Singh (1999). Multivariate analysis by data depth: descriptive statistics, graphics and inference, (with discussion and a rejoinder by liu and singh). *The Annals of Statistics 27*(3), 783–858.

Liu, R. Y. and K. Singh (1993). A quality index based on data depth and multivariate rank tests. *Journal of the American Statistical Association 88*(401), 252–260.

Liu, X. (2017). Fast implementation of the Tukey depth. *Computational Statistics 32*, 1395–1410.

Liu, X., S. Luo, and Y. Zuo (2020). Some results on the computing of Tukey's halfspace median. *Statistical Papers 61*, 303–316.

Liu, X., K. Mosler, and P. M. (2019). Fast computation of Tukey trimmed regions and median in dimension $p > 2$. *Journal of Computational and Graphical Statistics 28*(3), 682–697.

Liu, X. and Y. Zuo (2014a). Computing halfspace depth and regression depth. *Communications in Statistics - Simulation and Computation 43*(5), 969–985.

Liu, X. and Y. Zuo (2014b). Computing projection depth and its associated estimators. *Statistics and Computing 24*, 51–63.

Liu, X. and Y. Zuo (2015). Comppd: A MATLAB package for computing projection depth. *Journal of Statistical Software 65*(2), 1–21.

Liu, Z. and R. Modarres (2011). Lens data depth and median. *Journal of Nonparametric Statistics 23*(4), 1063–1074.

López-Pintado, S. and J. Romo (2009). On the concept of depth for functional data. *Journal of the American Statistical Association 104*(486), 718–734.

López-Pintado, S. and J. Romo (2011). A half-region depth for functional data. *Computational Statistics and Data Analysis 55*(4), 1679–1695.

López-Pintado, S., Y. Sun, J. K. Lin, and M. G. Genton (2014). Simplicial band depth for multivariate functional data. *Advances in Data Analysis and Classification 8*(3), 321–338.

M., P. (2015). *Contributions to Depth-Based Classification and Computation of the Tukey Depth*. Ph. D. thesis, University of Cologne.

M., P. (2016). Tukey depth: linear programming and applications. *arXiv:1603.00069*.

M., P. and C. B. Barber (2021). *TukeyRegion: Tukey Region and Median*. R package version 0.1.4, https://CRAN.R-project.org/package=TukeyRegion.

M., P., J. Josse, and F. Husson (2020). Nonparametric imputation by data depth. *Journal of the American Statistical Association 115*(529), 241–523.

M., P., P. Lafaye De Micheaux, and M. Vimond (2019). *curveDepth: Tukey Curve Depth and Distance in the Space of Curves*. R package version 0.1.0.9, https://CRAN.R-project.org/package=curveDepth.

M., P., K. Mosler, and T. Lange (2015). Classifying real-world data with the $DD\alpha$-procedure. *Advances in Data Analysis and Classification 9*, 287–314.

M., P. and J. Vogler (2016). Composite marginal likelihood estimation of spatial autoregressive probit models feasible in very large samples. *Economics Letters*.

Mahalanobis, P. C. (1936). On the generalized distance in statistics. *Proceedings of the National Institute of Sciences of India 12*, 49–55.

Mahalanobish, O. and S. Karmakar (2015). *depth.plot: Multivariate Analogy of Quantiles*. R package version 0.1, https://CRAN.R-project.org/package=depth.plot.

Mallat, S. and Z. Zhang (1993). Matching pursuits with time-frequency dictionaries. *IEEE Transactions on Signal Processing 41*(12), 3397–3415.

Maz'ya, V. (2011). *Sobolev Spaces: with Applications to Elliptic Partial Differential Equations*. Berlin Heidelberg: Springer-Verlag.

Mercier, C., P. Gori, D. Rohmer, M.-P. Cani, T. Boubekeur, J.-M. Thiery, and I. Bloch (2018). Progressive and Efficient Multi-Resolution Representations for Brain Tractograms. In *Eurographics Workshop on Visual Computing for Biology and Medicine*. The Eurographics Association.

Mirzargar, M., R. T. Whitaker, and R. M. Kirby (2014). Curve boxplot: Generalization of boxplot for ensembles of curves. *IEEE Transactions on Visualization and Computer Graphics 20*(12), 2654–2663.

Modat, M., D. Cash, P. Daga, G. Winston, J. Duncan, and S. Ourselin (2014). Global image registration using a symmetric block-matching approach. *Journal of Medical Imaging 1*(2), 024003.

Mosler, K. (2002). *Multivariate Dispersion, Central Regions, and Depth: The Lift Zonoid Approach*. Lecture Notes in Statistics. Springer New York.

Mosler, K. (2013). Depth statistics. In C. Becker, R. Fried, and S. Kuhnt (Eds.), *Robustness and Complex Data Structures: Festschrift in Honour of Ursula Gather*, pp. 17–34. Springer, Berlin.

Mosler, K., T. Lange, and P. Bazovkin (2009). Computing zonoid trimmed regions in dimension $d > 2$. *Computational Statistics and Data Analysis 53*(7), 2500–2510.

Mosler, K. and P. M. (2017). Fast DD-classification of functional data. *Statistical Papers 58*(4), 1055–1089.

Mosler, K. and P. M. (2022). Choosing among notions of multivariate depth statistics. *Statistical Science 37*(3), 348–368.

Mosler, K. and Y. Polyakova (2018). General notions of depth for functional data. *arXiv:1208.1981*.

Nagy, S., R. Dyckerhoff, and P. M. (2020). Uniform convergence rates for the approximated halfspace and projection depth. *Electronic Journal of Statistics 14*(2), 3939–3975.

Narisetty, N. N. and V. N. Nair (2016). Extremal depth for functional data and applications. *Journal of the American Statistical Association 111*(516), 1705–1714.

Nelder, J. A. and R. Mead (1965). A simplex method for function minimization. *The Computer Journal 7*(4), 308–313.

Nieto-Reyes, A. and H. Battey (2016). A topologically valid definition of depth for functional data. *Statistical Science 31*(1), 61–79.

Oja, H. (1983). Descriptive statistics for multivariate distributions. *Statistics and Probability Letters 1*(6), 327–332.

Paindaveine, D. and M. Šiman (2011). On directional multiple-output quantile regression. *Journal of Multivariate Analysis 102*(2), 193–212.

Paindaveine, D. and M. Šiman (2012a). Computing multiple-output regression quantile regions. *Computational Statistics and Data Analysis 56*(4), 840–853.

Paindaveine, D. and M. Šiman (2012b). Computing multiple-output regression quantile regions from projection quantiles. *Computational Statistics 27*(1), 29–49.

Paindaveine, D. and G. Van Bever (2013). From depth to local depth: A focus on centrality. *Journal of the American Statistical Association 108*(503), 1105–1119.

Parekh, J., P. M., and F. D'Alché-Buc (2021). A framework to learn with interpretation. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan (Eds.), *Advances in Neural Information Processing Systems 34 (NeurIPS 2021)*.

Parthasarathy, K. R. (1967). *Probability Measures on Metric Spaces*. New York: Academic Press.

Pokotylo, O., P. M., and R. Dyckerhoff (2019). Depth and depth-based classification with R package ddalpha. *Journal of Statistical Software 91*(5), 1–46.

Pokotylo, O., P. M., R. Dyckerhoff, and S. Nagy (2020). *ddalpha: Depth-Based Classification and Calculation of Data Depth*. R package version 1.3.11, https://CRAN.R-project.org/package=ddalpha.

R Core Team (2022). *R: A Language and Environment for Statistical Computing*. Vienna, Austria: R Foundation for Statistical Computing. https://www.R-project.org.

Ramsay, J. O. and B. W. Silverman (2002). *Applied Functional Data Analysis: Methods and Case Studies*. New York: Springer-Verlag.

Ramsay, J. O. and B. W. Silverman (2005). *Functional Data Analysis*. New York: Springer-Verlag.

Rousseeuw, P. J. Struyf, A. (1998). Computing location depth and regression depth in higher dimensions. *Statistics and Computing 8*, 193–203.

Rousseeuw, P. J., M. Debruyne, S. Engelen, and M. Hubert (2006). Robustness and outlier detection in chemometrics. *Critical Reviews in Analytical Chemistry 36*(3–4), 221–242.

Rousseeuw, P. J. and M. Hubert (1999). Regression depth. *Journal of the American Statistical Association 94*, 388–433.

Rousseeuw, P. J. and M. Hubert (2018). Anomaly detection by robust statistics. *WIREs Data Mining and Knowledge Discovery 8*(2), e1236.

Rousseeuw, P. J. and A. M. Leroy (1987). *Robust Regression and Outlier Detection*. New York : Wiley.

Rousseeuw, P. J. and I. Ruts (1996). Algorithm as 307: Bivariate location depth. *Journal of the Royal Statistical Society. Series C: Applied Statistics 45*, 516–526.

Rousseeuw, P. J. and I. Ruts (1998). Constructing the bivariate Tukey median. *Statistica Sinica 8*(3), 827–839.

Rousseeuw, P. J. and K. Van Driessen (1999). A fast algorithm for the minimum covariance determinant estimator. *Technometrics 41*(3), 212–223.

Ruts, I. and P. J. Rousseeuw (1996). Computing depth contours of bivariate point clouds. *Computational Statistics and Data Analysis 23*(1), 153–168.

Sachdev, P., A. Lammel, J. Trollor, T. Lee, M. Wright, D. Ames, W. Wen, N. Martin, H. Brodaty, P. Schofield, and the OATS research team (2009). A comprehensive neuropsychiatric study of elderly twins: The Older Australian Twins Study. *Twin Research and Human Genetics 12*(6), 573–582.

Sangalli, L. M., P. Secchi, S. Vantini, and A. Veneziani (2009). Efficient estimation of three-dimensional curves and their derivatives by free-knot regression splines, applied to the analysis of inner carotid artery centrelines. *Journal of the Royal Statistical Society. Series C (Applied Statistics) 58*(3), 285–306.

Schölkopf, B., J. Platt, J. Shawe-Taylor, A. Smola, and R. Williamson (2001). Estimating the support of a high-dimensional distribution. *Neural Computation 13*(7), 1443–1471.

Schölkopf, B. and A. J. Smola (2002). *Learning with kernels: Support vector machines, regularization, optimization, and beyond*. MIT press.

Scott, C. and R. Nowak (2006). Learning minimum volume sets. *Journal of Machine Learning Research 7*, 665–704.

Segaert, P., M. Hubert, P. Rousseeuw, and J. Raymaekers (2020). *mrfDepth: Depth Measures in Multivariate, Regression and Functional Settings.* R package version 1.0.13, https://CRAN.R-project.org/package=mrfDepth.

Serfling, R. (2002). A depth function and a scale curve based on spatial quantiles. In Y. Dodge (Ed.), *Statistical Data Analysis Based on the L_1-Norm and Related Methods*, Statistics for Industry and Technology book series (SIT), Basel, pp. 25–38. Birkhäuser.

Shin, H. J., D. Eom, and S. S. Kim (2005). One-class support vector machines – an application in machine fault detection and classification. *Computers and Industrial Engineering 48*(2), 395–408.

Simar, L. and P. Wilson (1998). Sensitivity analysis of efficiency scores: How to bootstrap in nonparametric frontier models. *Management Science 44*, 49–61.

Simar, L. and P. Wilson (2000). A general methodology for bootstrapping in nonparametric frontier models. *Journal of Applied Statistics 27*, 779–802.

Simar, L. and P. Wilson (2002). Nonparametric tests of return to scale. *European Journal of Operational Research 139*, 115–132.

Snyder, J. P. (1987). Map projections: A working manual. Professional Paper 1395, U. S. Geological Survey, Washington D.C.

Song, L., K. Fukumizu, and A. Gretton (2013). *IEEE Signal Processing Magazine 30*(4), 98–111.

Staerman, G., P. Laforgue, P. M., and F. D'Alché-Buc (2021). When ot meets mom: Robust estimation of wasserstein distance. In A. Banerjee and K. Fukumizu (Eds.), *Proceedings of The Twenty Fourth International Conference on Artificial Intelligence and Statistics (AISTATS 2021)*, Volume 130, pp. 136–144.

Staerman, G., P. M., and S. Clémençon (2020). The area of the convex hull of sampled curves: a robust functional statistical depth measure. In S. Chiappa and R. Calandra (Eds.), *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics (AISTATS 2020)*, Volume 108, pp. 570–579.

Staerman, G., P. M., S. Clémençon, and F. d'Alché Buc (2019). Functional isolation forest. In W. S. Lee and T. Suzuki (Eds.), *Proceedings of The Eleventh Asian Conference on Machine Learning (ACML 2019)*, Volume 101, pp. 332–347.

Statzer, C., E. Jongsma, S. X. Liu, A. Dakhovnik, F. Wandrey, P. M., F. Zülli, and C. Y. Ewald (2021). Youthful and age-related matreotypes predict drugs promoting longevity. *Aging Cell 20*, e13441.

Stekhoven, D. J. and P. Bühlmann (2012). MissForest – non-parametric missing value imputation for mixed-type data. *Bioinformatics 28*(1), 112–118.

Su, J., S. Kurtek, E. Klassen, and A. Srivastava (2014). Statistical analysis of trajectories on Riemannian manifolds: Bird migration, hurricane tracking and video surveillance. *The Annals of Applied Statistics 8*(1), 530–552.

Theussl, S. and K. Hornik (2019). *Rglpk: R/GNU Linear Programming Kit Interface*. R package version 0.6-4, https://CRAN.R-project.org/package=Rglpk.

Tournier, J., F. Calamante, and A. Connelly (2012). MRtrix: diffusion tractography in crossing fiber regions. *International Journal of Imaging Systems and Technology 22*(1), 53–66.

Troyanskaya, O., M. Cantor, G. Sherlock, P. Brown, T. Hastie, R. Tibshirani, D. Botstein, and R. B. Altman (2001). Missing value estimation methods for dna microarrays. *Bioinformatics 17*(6), 520–525.

Tukey, J. W. (1975). Mathematics and the picturing of data. In R. James (Ed.), *Proceedings of the International Congress of Mathematicians*, Volume 2, pp. 523–531. Canadian Mathematical Congress.

Udell, M. and A. Townsend (2018). Why are big data matrices approximately low rank? *arXiv:1705.07474*.

Väisälä, J. (2006). *Lectures on n-Dimensional Quasiconformal Mappings*. Lecture Notes in Mathematics. Berlin Heidelberg: Springer.

Van Buuren, S. (2012). *Flexible Imputation of Missing Data*. Chapman and Hall/CRC.

Van Der Walt, S., J. Schänberger, J. Nunez-Iglesias, F. Boulogne, J. Warner, N. Yager, E. Gouillart, T. Yu, and the scikit-image contributors (2014). *scikit-image: image processing in Python*.

Vardi, Y. and C.-H. Zhang (2000). The multivariate L1-median and associated data depth. *Proceedings of the National Academy of Sciences 97*(4), 1423–1426.

Šiman, M. and P. Boček (2019). *modQR: Multiple-Output Directional Quantile Regression*. R package version 0.1.2, https://CRAN.R-project.org/package=modQR.

Wen, W., A. Thalamuthu, K. A. Mather, W. Zhu, J. Jiang, P. Lafaye De Micheaux, M. J. Wright, D. Ames, and P. S. Sachdev (2016). Distinct genetic influences on cortical and subcortical brain structures. *Scientific Reports 6*, 32760.

Xu, Y., L. Cheng, L. Zhang, H. Yin, and X. Yin (2001). Mechanical properties of 3D fiber reinforced C/SiC composites. *Materials Science and Engineering: A 300*(1), 196–202.

Yang, M. and R. Modarres (2018). β-skeleton depth functions and medians. *Communications in Statistics - Theory and Methods 47*(20), 5127–5143.

Yeh, I.-C., K.-J. Yang, and T.-M. Ting (2009). Knowledge discovery on RFM model using bernoulli sequence. *Expert Systems with Applications 36*(3, Part 2), 5866–5871.

Yuan, G., P. Sun, J. Zhao, D. Li, and C. Wang (2017). A review of moving object trajectory clustering algorithms. *Artificial Intelligence Review 47*(1), 123–144.

Zawadzki, Z., D. Kosiorowski, K. Slomczynski, M. Bocian, and A. Wegrzynkiewicz (2020). *DepthProc: Statistical Depth Functions for Multivariate Analysis*. R package version 2.1.3, https://CRAN.R-project.org/package=DepthProc.

Zuo, Y. and R. Serfling (2000). General notions of statistical depth function. *The Annals of Statistics 28*, 461–482.