TU DORTMUND

DOCTORAL DISSERTATION

Scalable Bayesian Methods for Large-Scale Data: Data Reduction and Efficient Computation of Distribution Metrics

Author:
Zeyu DING

Supervisor:
Prof. Dr. Katja ICKSTADT
Dr. Alexander MUNTEANU

A thesis submitted in fulfillment of the requirements for the degree of Doktor der Naturwissenschaften

in the

Mathematical Statistics and Biometric Applications
Department of Statistics

February 24, 2026

Declaration of Authorship

I, Zeyu DING, declare that this thesis titled, "Scalable Bayesian Methods for Large-Scale Data: Data Reduction and Efficient Computation of Distribution Metrics" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at the TU Dortmund University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at the TU Dortmund University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed my self.

Signed:

Date:

TU DORTMUND

Abstract

Department of Statistics

Doktor der Naturwissenschaften

Scalable Bayesian Methods for Large-Scale Data: Data Reduction and Efficient Computation of Distribution Metrics

by Zeyu DING

In the era of big data, traditional Bayesian inference methods face significant challenges in computational efficiency and scalability. This thesis presents a comprehensive framework addressing these challenges through theoretical innovations and practical implementations. We introduce a novel p-probit model incorporating p-generalized normal distributions, which offers enhanced flexibility in modeling tail behavior through an adaptive parameter p. To address computational challenges with large-scale datasets, we develop an efficient *coreset*-based data reduction technique for the p-probit model, with theoretical guarantees based on the Wasserstein distance. Furthermore, we extend scalable inference to semi-parametric Multivariate Conditional Transformation Models (MCTMs). We propose a novel hybrid *coreset* strategy that combines leverage score sampling with a geometric convex hull approximation. This approach effectively resolves the numerical instabilities of logarithmic terms in the likelihood, enabling efficient learning of complex dependence structures with rigorous error guarantees. This exploration of distribution metrics leads to our investigation of scalable computation methods for probability distribution distances, where we propose novel approximation approaches using sliced-Wasserstein distances and random Fourier features in Physics applications. These theoretical advances are implemented in two open-source software packages: an `R` package for the p-probit model and a `Julia` package for distribution metric computation. Our empirical results demonstrate significant improvements in both computational efficiency and statistical accuracy across various large-scale applications, contributing to both theoretical understanding and practical capabilities in modern Bayesian inference.

Acknowledgements

I would like to express my profound gratitude to my supervisor, Prof. Dr. Katja Ickstadt. Her mentorship has been instrumental not only in guiding my research but also in shaping my development as a researcher. I am deeply thankful for the trust she placed in me, granting me the academic freedom to explore my ideas, while always ensuring I had a supportive environment. Her advice, extending beyond academia to personal well-being, has been a constant source of encouragement.

I am equally indebted to my co-supervisor, Dr. Alexander Munteanu, for his pivotal role in this work. His rigorous approach to research has taught me invaluable lessons in theoretical derivation and academic writing. I appreciate the generosity with which he shared his knowledge; his meticulous feedback has significantly elevated the quality of this dissertation.

A special note of thanks goes to my colleague, Dr. Simon Omlor. I deeply value our close exchange of ideas. Our numerous discussions often provided clarity on complex theoretical issues, and his camaraderie made the daily challenges of research much more enjoyable.

I also wish to extend my sincere gratitude to my collaborators, Prof. Dr. Nadja Klein, Prof. Dr. Kevin Kröninger, Dr. Cornelius Grunwald, and Dr. Salvatore La Cagnina. I appreciate our productive cooperation, the expert insights from our discussions, and the collective effort that contributed to the success of our joint projects.

To my family, and especially my mother, Xuemin Bai, I owe my deepest thanks. You have been my silent backbone. Your unwavering faith in my abilities and your unconditional support have been the foundation upon which I could build this work, giving me the strength to persevere through every challenge.

I am also grateful to my furry companions, Hagen and Lisa, whose presence provided much-needed relief during stressful times.

Finally, I reserve my most special thanks for my girlfriend, Qingxue Huang. Her steadfast support—emotionally and spiritually—has been my anchor. Thank you for your enduring belief in me and for walking this path by my side.

Contents

List of Figures

List of Tables

List of Abbreviations

AE	**A**bsolute **E**rror
BNNs	**B**ayesian **N**eural **N**etworks
CDF	**C**umulative **D**istribution **F**unction
CTMs	**C**onditional **T**ransformation **M**odels
DGP	**D**ata **G**eneration **P**rocess
ESS	**E**ffective **S**ample **S**ize
GAM	**G**eneralized **A**dditive **M**odel
GANs	**G**enerative **A**dversarial **N**etworks
GLM	**G**eneralized **L**inear **M**odel
GP	**G**aussian **P**rocess
IID	**I**ndependent and **I**dentically **D**istributed
KL	**K**ullback-**L**eibler (Divergence)
LP	**L-p** norm
MC	**M**onte **C**arlo
MCMC	**M**arkov **C**hain **M**onte **C**arlo
MCTMs	**M**ulti-variate **C**onditional **T**ransformation **M**odels
MMD	**M**aximum **M**ean **D**iscrepancy
MH	**M**etropolis-**H**astings
MLE	**M**aximum **L**ikelihood **E**stimation
NF	**N**ormalizing **F**low
NLL	**N**egative **L**og-**L**ikelihood
NN	**N**eural **N**etwork
p-GGD	**p**-**G**eneralized **G**aussian **D**istribution
PSRF	**P**otential **S**cale **R**eduction **F**actor
RFF	**R**andom **F**ourier **F**eatures
RKHS	**R**eproducing **K**ernel **H**ilbert **S**pace
RSS	**R**esidual **S**um of **S**quares
SVR	**S**upport **V**ector **R**egression
SWD	**S**liced **W**asserstein **D**istance
VAE	**V**ariational **A**uto**E**ncoders
VC	**V**apnik-**C**hervonenkis (Dimension)

List of Notations

This thesis compiles work from several distinct research projects. While efforts have been made to unify notation, certain symbols may vary in meaning depending on the specific context (e.g., Regression Modeling vs. Distributional Metrics). The following table clarifies the usage of key symbols across different chapters.

Symbol	Description
General & Sets	
$\mathbb{R}^d$	d-dimensional Euclidean space
$[n]$	Set of integers $\{1, 2, \ldots, n\}$
$\|\cdot\|_p$	L_p-norm, defined as $\|x\|_p = (\sum \|x_i\|^p)^{1/p}$
$\mathcal{N}(\mu, \Sigma)$	Multivariate Normal distribution with mean μ and covariance Σ
X	Design matrix or dataset, typically $X \in \mathbb{R}^{n \times d}$ with dimensions of n rows and d columns, however, the dimension notation of the data might change due to specific model settings.
$\mathbb{E}[\cdot]$	Expectation operator
$\mathbb{P}(\cdot)$	Probability of an event
$\mathbb{I}(\cdot)$	Indicator function (1 if condition is true, 0 otherwise)
$\mathcal{L}(\theta)$	Likelihood function with specific population parameter θ
Bayesian p-Probit Models (Chapter 2 & 3)	
y_i	Binary response variable, $y_i \in \{0, 1\}$
β	Regression coefficient vector
p	**Shape parameter** of the p-Generalized Gaussian Distribution (p-GGD), controlling tail behavior ($p = 1$: Laplace, $p = 2$: Gaussian)
$\Phi_p(\cdot)$	Cumulative Distribution Function (CDF) of the p-GGD
$\phi_p(\cdot)$	Probability Density Function (PDF) of the p-GGD
Z	Latent variable used in data augmentation
Coreset & Sensitivity Sampling (Chapter 3)	
$\mu(X)$	μ-complexity of the dataset X (used in Coreset bounds)
ζ_i	Sensitivity of the i-th observation, measuring its importance to the objective function
s_i	Upper bound of the sensitivity ζ_i, used for sampling probabilities
S	Total sensitivity, $S = \sum s_i$ (determines the required coreset size)
w_i	Weight assigned to the i-th point in the coreset

List of Notations – Continued

Symbol	Description
ϵ	Relative error bound for the coreset approximation (e.g., $(1 \pm \epsilon)$-approximation)
Multivariate Conditional Transformation Models (Chapter 3)	
Y	Multivariate continuous response vector, $Y \in \mathbb{R}^J$
$a(\cdot)$	Basis function transformation (e.g., Bernstein polynomials)
ϑ	Coefficients for the basis functions
λ	Parameters governing the dependence structure (interaction terms)
θ	Collection of all model parameters, $\theta = (\vartheta^\top, \lambda^\top)^\top$
u_i	Statistical leverage score of the i-th observation
Distribution Distances & Metrics (Chapter 4)	
μ, ν	Probability measures or distributions to be compared
$W_p(\mu, \nu)$	p-Wasserstein distance, where p denotes the **order of the distance**
SWD	Sliced Wasserstein Distance
θ	Random projection direction vector on the unit sphere S^{d-1} (Specific to SWD context)
$k(\cdot, \cdot)$	Kernel function used in Maximum Mean Discrepancy (MMD)
$\mathcal{H}$	Reproducing Kernel Hilbert Space (RKHS)

List of Publications

This cumulative thesis is based on the following four manuscripts and two open-source scientific computing packages:

Article 1: Zeyu Ding, Simon Omlor, Katja Ickstadt, Alexander Munteanu. *Scalable Bayesian p-generalized probit and logistic regression*. Published in: *Advances in Data Analysis and Classification*, 1–35. DOI: https://doi.org/10.1007/s11634-024-00599-1

Contribution of the thesis author: All authors contributed equally to this work and are listed in alphabetical order.

In terms of theoretical contributions, the author of this thesis developed the Bayesian framework for the p-generalized probit model and derived the Markov chain Monte Carlo (MCMC) sampling algorithm with the helpful inputs from all other co-authors. Dr. Simon Omlor, together with Dr. Alexander Munteanu, provided the mathematical derivation and theoretical guarantees for the coreset construction. Regarding the experimental part, the thesis author implemented the full codebase in `R`*, processed the real-world datasets, and executed all experiments. The design of the simulation study was developed through in-depth discussions with Dr. Simon Omlor, Prof. Dr. Katja Ickstadt and Dr. Munteanu provided overall supervision and conceptual input throughout the project.*

Article 2: Zeyu Ding, Cornelius Grunwald, Katja Ickstadt, Kevin Kröninger, Salvatore La Cagnina. *MCBench: A Benchmark Suite for Monte Carlo Sampling Algorithms*. Preprint available at: *arXiv:2501.03138* URL: https://arxiv.org/abs/2501.03138
[Manuscript also currently under peer review by a scientific journal]
The reuse of this preprint in the thesis complies with the author rights retained under arXiv policy. The authors have not disclosed any submission venue in public documents to preserve the integrity of the double-blind review process.
Contribution of the thesis author: All authors contributed equally to this work and are listed in alphabetical order.

The author of the thesis proposed the utilization of Wasserstein distance and Maximum Mean Discrepancy (MMD) as evaluation metrics and drafted the corresponding theoretical sections. He conducted research on efficient calculation of the metrics and proposed to use the sliced and random Fourier features to approximate the metrics. To establish a comprehensive benchmark, he implemented these approaches in `R` *and* `Python`*. Dr. Cornelius Grunwald and Dr. Salvatore La Cagnina focused on the implementation of these metrics within the* `Julia`

system. Prof. Dr. Katja Ickstadt and Prof. Dr. Kevin Kröninger provided critical input on the manuscript writing, proofreading, and overall supervision.

Article 3: Zeyu Ding, Katja Ickstadt, Nadja Klein, Alexander Munteanu, Simon Omlor. *Scalable Learning of Multivariate Distributions via Coresets*. To appear in *Proceedings of the 29th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2026. *The content of this article is included in this thesis. The article is published under a Creative Commons Attribution 4.0 International License (CC BY 4.0), which permits reuse with appropriate credit.*
Contribution of the thesis author: All authors contributed equally to this work and are listed in alphabetical order.

Dr. Alexander Munteanu and Dr. Simon Omlor derived the theoretical error bounds and established the mathematical guarantees for the coreset construction. Prof. Dr. Nadja Klein provided essential methodological guidance on Multivariate Conditional Transformation Models, contributed to the manuscript drafting, and refined the statistical arguments. Prof. Dr. Katja Ickstadt supervised the project and contributed to the writing and structuring of the manuscript. The author of this thesis was responsible for written part of the draft of the paper and the full computational implementation of the proposed methods. He designed and executed all simulation studies and real-world data experiments, translating the theoretical concepts into a functional algorithmic framework with the helpful discussion with all other co-authors.

Packages: In the course of these projects, the author of this thesis co-developed the following open-source packages with the other co-authors:

1. `MCbench`: *Monte Carlo Benchmark Suite for Bayesian Inference and Sampling Algorithms* `https://github.com/tudo-physik-e4/MCBench`

2. `BayesPprobit`: *Scalable Bayesian p-Generalized Probit and Logistic Regression Models* `https://github.com/zeyudsai/BayesPprobit`

Other publications:
Article 4: Zeyu Ding, Katja Ickstadt, Alexander Munteanu. *Bayesian Analysis for Dimensionality and Complexity Reduction*. Published in: *Machine Learning under Resource Constraints, Volume 3 - Applications*, Berlin, Boston: De Gruyter. Chap. 2.4 (2023), pp. 58-70. Link: `https://www.degruyterbrill.com/document/doi/10.1515/9783110785982/html` [Book chapter]
Contribution of the thesis author: All authors contributed equally to this work and are listed in alphabetical order.

The author of this thesis summarized previous research, conducted a literature review, wrote the first draft of this article, and polished the final draft of the article with other authors.

Chapter 1

Introduction and Structure

1.1 Research Motivation and Context

Recent advances in data science, particularly in Bayesian statistics and generative modeling, have driven demand for tools that not only scale to large datasets but also provide principled measures of uncertainty and distributional discrepancy. Traditional statistical methods have long centered on point estimation and hypothesis testing under strong parametric assumptions. The theoretical guarantees for these approaches derive from asymptotic results, such as the central limit theorem (CLT) and laws of large numbers (LLN), which characterize the limiting behavior of estimators as sample size grows (Kwak and Kim, 2017; DasGupta, 2008). While these results provide powerful guarantees for estimating central tendencies such as population means, they often overlook the broader structural differences between probability distributions (Gneiting and Raftery, 2007).

1.1.1 The Bayesian Paradigm for Uncertainty Quantification

However, in many real-world applications, including simulation of complex social systems, medical risk assessment, and particle physics experiments, the research objective extends beyond estimating particular statistics. Instead, practitioners seek to understand the shape, structure, and uncertainty of entire probability distributions. Bayesian statistics provides a natural framework for this objective by explicitly modeling uncertainty through full posterior distributions rather than point estimates of the parameters (Gelman et al., 2013; McElreath, 2018). By interpreting probability as a degree of belief that is updated in light of new evidence, Bayesian inference naturally propagates uncertainty through all stages of analysis. This full distributional characterization proves particularly valuable when decisions must account for tail risks, multimodal structure, or heterogeneous uncertainty across different regions of the parameter space (King et al., 2019).

1.1.2 Distributional Distances in the Era of Generative Models

Beyond traditional inference tasks, the rise of generative models has fundamentally transformed how we evaluate statistical methods. In applications ranging from Generative Adversarial Networks (GANs, Goodfellow et al., 2014) to Bayesian simulation-based inference, the core challenge shifts from parameter estimation to distributional comparison. Whether comparing generated samples to observed data or evaluating the convergence of posterior approximations, we must quantify how "close" two probability distributions are, a foundational problem that requires principled distance metrics.

Classical divergence measures, particularly the Kullback-Leibler (KL) divergence (Kullback and Leibler, 1951), have long served as the cornerstone of statistical inference, underlying maximum likelihood estimation and variational methods. However, several well-documented pathologies limit the applicability of KL divergence in modern settings. First, its asymmetry and undefined behavior on distributions with disjoint support create theoretical complications. Second, in generative modeling, optimizing KL divergence can lead to undesirable "mode-seeking" behavior, where the learned distribution fails to capture all modes of the target, a phenomenon known as mode collapse (Arjovsky et al., 2017; Chan et al., 2022). Third, in high-dimensional settings where data often concentrates on lower-dimensional manifolds, KL divergence may fail to provide meaningful gradients for optimization, as the distributions rarely share common support (Dhaka et al., 2021; Verine et al., 2023).

These limitations have motivated substantial interest in alternative metrics. The Wasserstein distance, grounded in optimal transport theory, measures the minimum "cost" of transforming one distribution into another and remains well-defined even for distributions with disjoint supports (Arjovsky et al., 2017). Similarly, Maximum Mean Discrepancy (MMD) (Gretton et al., 2012a) offers a kernel-based approach that compares distributions through their embeddings in a reproducing kernel Hilbert space, eliminating the need for explicit density estimation. Both metrics have proven particularly effective in high-dimensional settings, where they capture geometric structure that KL divergence may miss (Gao and Shao, 2023).

1.2 Research Questions and Scope

The developments outlined above highlight three interrelated challenges in modern statistical practice. First, we need flexible probabilistic models that can adapt to diverse data characteristics without imposing restrictive parametric assumptions. Second, we require principled methods to compare and evaluate these models through distributional distance metrics. Third, we must develop computationally efficient inference procedures that scale to large datasets while maintaining theoretical guarantees.

Addressing these challenges forms the central motivation for this dissertation, which is organized around two fundamental questions:

1. How to effectively model uncertainty in complex data and construct flexible and interpretable probabilistic models?

2. How can these probabilistic models be reliably and efficiently inferred under limited computational resources?

Focusing on the above issues, the research of this thesis focuses on three closely related themes:

1. **Bayesian modeling**: full distributional representation of uncertainty;

2. **Data compression and efficient inference**: building computationally friendly probabilistic and Bayesian frameworks;

3. **Distributional distance metrics**: understanding and comparing differences in probabilistic models.

The following section develops each of these three endeavors and points out how they are reflected in the three representative studies in this dissertation.

1.3 Methodological Framework and Contributions

1.3.1 Flexible Modeling and Uncertainty Representation in Bayesian Models

In the first work of this thesis, a new regression modeling framework is proposed: the *Bayesian p-probit regression model*. The flexibility of this model is derived from the use of the $p-$generalized normal distribution, also known as the exponential power distribution, as the latent variable distribution. This family of distributions was first introduced by Subbotin (1923) and later popularized in the context of Bayesian robust modeling by Box and Tiao (2011). By treating the shape parameter p as an unknown to be inferred from the data, the model can adapt its tail behavior, smoothly interpolating between the Gaussian distribution ($p = 2$), which underlies the standard probit model, and the leptokurtic Laplace distribution ($p = 1$), which is associated with robust and sparse models.

Standard link functions in binary regression, such as the probit (Gaussian CDF) and logit (logistic CDF), implicitly assume specific tail behaviors for the underlying latent variable. When real-world data exhibit heavier or lighter tails than these assumptions suggest, forcing a fixed link function can lead to biased inference and poor predictive performance. The robust statistics literature has long emphasized the need for models that adapt to such deviations from assumed distributions rather than imposing them *a priori*, as we demonstrate in Chapter 2

The model's flexibility is further enhanced by adopting a fully Bayesian framework. Rather than treating the shape parameter p as fixed or estimating it through maximum likelihood, we place a prior distribution on p and perform joint posterior

inference over both β and p. This approach offers several advantages over existing p-probit methods (Munteanu et al., 2022). First, it eliminates the need for analysts to pre-specify the level of robustness or tail behavior, instead learning these characteristics directly from the observed data. Second, by providing full posterior distributions rather than point estimates, the Bayesian approach naturally quantifies uncertainty in both the regression coefficients and the inferred tail behavior. This dual uncertainty quantification proves particularly valuable for prediction and model selection, where understanding the confidence in the chosen link function is as important as the regression coefficients themselves. Third, the posterior distribution of p serves as an interpretable diagnostic: values near 2 suggest the data are well-described by a standard probit model, while departures indicate the degree and direction of tail deviation from Gaussianity.

This work not only proposes a new model, but also designs an effective Markov chain Monte Carlo (MCMC) inference method, and incorporates data compression techniques (see the next section) to achieve a significant improvement in inference efficiency.

1.3.2 Coreset Methods for Scalable Probabilistic and Bayesian Inference

Despite their advantages in flexible modeling and uncertainty quantification, Bayesian methods face computational challenges that can become prohibitive in practice. Standard MCMC inference scales poorly with dataset size, as each iteration requires evaluating the likelihood over all n data points. For large-scale applications, this computational burden can render otherwise attractive Bayesian models impractical. This motivates a fundamental question: can we perform inference on a carefully selected subset of the data while maintaining accuracy guarantees for the posterior distribution? The challenge of computational cost in Bayesian inference, particularly in the "Big Data" regime, has motivated a variety of scalable methods. The *Coreset* method provides a compelling, data-centric solution to this problem. Originating in computational geometry, the *Coreset* approach was adapted for Bayesian inference to construct a small, weighted subset of the data such that the posterior distribution conditioned on this subset is provably close to the posterior conditioned on the full dataset (Huggins et al., 2016; Zhang et al., 2021). This dissertation builds upon this foundational work and extends its applicability in two key directions:

- For the p-probit model, we develop a coreset construction method that combines Wasserstein distance (Monge, 1781) with leverage scores. This approach exploits both the geometric structure of the data (via Wasserstein distance) and the statistical influence of individual observations (via leverage scores). We provide both theoretical guarantees for posterior approximation quality and empirical validation on classification tasks;

- For Multivariate Conditional Transformation Models (MCTMs) (Klein et al., 2022), a flexible class of high-dimensional models that capture conditional

dependencies among variables, we devise a sparse coreset method based on convex hull approximation. This geometric approach identifies influential boundary points in the feature space, enabling substantial data reduction while preserving the model's ability to capture complex dependence structures. We demonstrate the method's effectiveness on several real-world datasets from diverse application domains.

Together, these studies point to a single goal: to achieve scalable and efficient probabilistic and Bayesian inference while guaranteeing the quality of probabilistic modeling.

1.3.3 Distance Metrics for Posterior Evaluation and Model Comparison

The rise of generative modeling and simulation-based inference has transformed the landscape of statistical evaluation. In these modern applications, the fundamental inferential question shifts from comparing summary statistics ("are the means equal?") to assessing overall distributional similarity ("are the distributions close?"). Answering this question requires principled distance metrics that can quantify discrepancies between probability distributions in ways that are both statistically meaningful and computationally tractable. In the third work of this thesis, we have developed an evaluation toolkit, `MCbench`, which is able to compute a number of common distribution distances in the programming language `Julia`, including:

1. **Wasserstein distance**: Grounded in optimal transport theory, the Wasserstein distance measures the minimum "cost" of transforming one distribution into another (Monge, 1781; Kantorovich, 1942). Unlike divergence measures, it remains well-defined for distributions with disjoint supports and provides a meaningful metric structure that is particularly sensitive to tail behavior and overall geometric structure;

2. **Maximum Mean Discrepancy (MMD)**: A kernel-based metric that compares distributions through their embeddings in a Reproducing Kernel Hilbert Space (RKHS) (Gretton et al., 2012a). By representing distributions via their mean embeddings, MMD enables comparison without explicit density estimation. The choice of kernel determines which distributional features are emphasized, providing flexibility in tailoring the metric to specific application needs;

3. **Classical measures**: Including chi-square statistic, moment-based comparisons, and summary statistic distances. While computationally simpler, these measures often capture specific aspects of distributional difference.

`MCbench` was developed out of a collaborative demand from the physics community for rigorous distributional comparisons between simulated data and real observations, rather than relying solely on central statistics or traditional hypothesis testing.

While these metrics provide principled measures of distributional discrepancy, their direct computation can become prohibitive for high-dimensional or large-sample settings. The Wasserstein distance, for instance, requires solving an optimal transport problem with $O(n^3)$ complexity for n samples. Similarly, computing MMD with m samples requires constructing an $n \times n$ kernel matrix, leading to $O(n^2)$ memory and computational costs.

To address these limitations, we investigate and implement several scalable approximation methods in `MCbench`. For the Wasserstein distance, we employ the *Sliced* Wasserstein distance (Kolouri et al., 2019), which projects the distributions onto random one-dimensional subspaces where the distance can be computed efficiently in closed form. For MMD, we implement kernel approximation techniques including Random Fourier Features (Li et al., 2019), which approximates the kernel via Monte Carlo sampling in the frequency domain. The Nyström method (Williams and Seeger, 2000), which constructs a low-rank approximation based on a subset of landmark points was also investigated. These approximations reduce computational complexity substantially while maintaining rigorous error bounds, as detailed in Chapter 4.

Beyond serving as a practical toolkit, this work contributes to the broader understanding of distribution-level modeling by systematically comparing the statistical and computational trade-offs inherent in different distance metrics and their approximations.

1.4 Software Contributions and Reproducibility

As part of this research, we developed two open-source packages:

- `BayesPprobit` — An `R` package implementing the Bayesian p-probit model with MCMC sampling;
- `MCbench` — A `Julia` package for benchmarking posterior approximations and distribution distances.

These tools are designed to support reproducible experiments and facilitate further research in scalable Bayesian inference.

1.5 Thesis Structure

In summary, this thesis focuses on the core problem of "how to perform effective Bayesian modeling under large-scale data", and develops the following three research lines:

1. Proposing new probabilistic models to enhance expressive power (Bayesian p-probit);
2. Utilizing the coreset technique to compress data to improve inference efficiency (p-probit and MCTM);

3. Introducing distributional distance metrics to support model evaluation and sample comparison (`MCbench`).

TABLE 1.1: Mapping of thesis chapters to included publications and software packages.

Chapter	Core Topic	Based on Publication
Chapter 2	Bayesian p-Probit Model: Framework & Inference	Article 1 (Ding et al., 2024)
Chapter 3	Scalable Inference via Coresets (p-Probit & MCTMs)	Article 1 (Ding et al., 2024), Article 3 (Ding et al., 2026), Article 4 (Ding et al., 2023)
Chapter 4	Distribution Distances & `MCbench` Framework	Article 2 (Ding et al., 2025)
Chapter 5	Software Design & Implementation	Packages: `BayesPprobit`, `MCbench`

The detailed mapping of these core research topics and their corresponding publications to the specific chapters of this dissertation is summarized in Table 1.1. These three parts are interconnected and together constitute the systematic response to the problem of *large-scale Bayesian probabilistic modeling and distance computation*. Figure 1.1 illustrates the logical structure and interconnections between these research components, providing an overview of how the individual studies are linked to the overarching research theme.

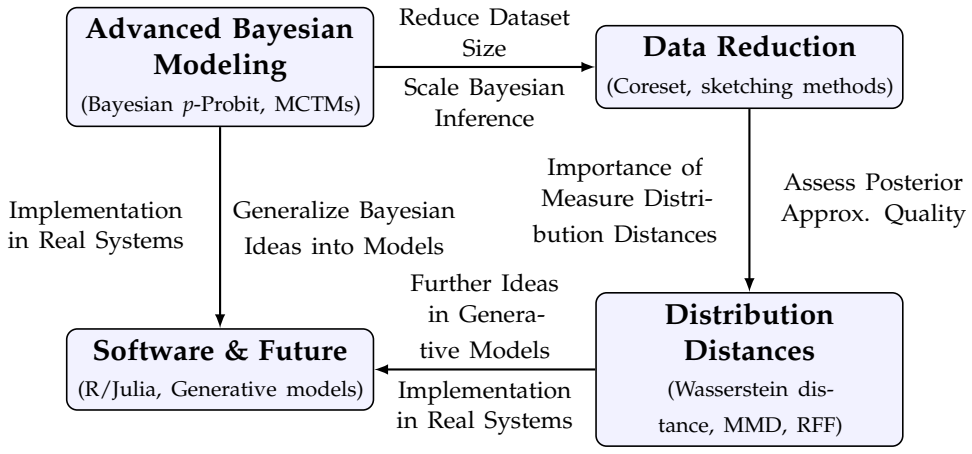


FIGURE 1.1: Research roadmap: From advanced Bayesian modeling to distribution distance metrics and scalable computation.

These works constitute the overall contribution of this research, both in terms of model-level innovations and computational-level method design and software implementation, covering multiple levels of theoretical analysis, algorithm development, and practical applications.

This thesis is structured as a cumulative dissertation, consisting of four main articles. Chapter 2 introduces the p-probit model and its Bayesian inference, the main content of this chapter is published in Ding et al. (2024). Chapter 3 introduces the construction of the coreset approach implemented for scalable inference in the p-probit setting and a coreset algorithm for MCTMs based on convex hull approximations, the main content of this chapter is published in Ding et al. (2023), Ding et al. (2024), and Ding et al. (2026). Chapter 4 introduces the `MCbench` benchmarking framework and explores scalable approximations for distribution distances, the main content of this chapter is under review by a journal, and published as a pre-print version by Ding et al. (2025). Chapter 5 introduces the structure and core functionality implementation of the `BayesPprobit`[1] and `MCbench`[2] package. Chapter 6 concludes this dissertation with a comprehensive summary of the research, highlights key contributions and limitations, and proposes feasible directions for future exploration grounded in the present findings.

[1] `https://github.com/zeyudsai/BayesPprobit`
[2] `https://github.com/tudo-physik-e4/MCBench.jl`

Chapter 2

Bayesian p-Probit Model: A New Framework for Binary Bayesian Inference

2.1 Introduction

In classical binary regression modeling, logistic and probit models have long been the two dominant choices. They both map linear predictors to a probability scale via fixed link functions, enabling estimation of success probabilities in binary outcomes. However, these fixed-form link functions lack flexibility when modeling data with heavy- or light-tailed structures, leading to potential link *misspecification*, which can distort inference and reduce predictive performance.

To address this issue, several recent works have proposed parametric link functions with tunable tail behavior. These models aim to provide a more accurate characterization of the data distribution and more robust model fitting. On the other hand, frequentist models typically provide only point estimates for parameters and lack direct quantification of uncertainty. Bayesian statistics naturally addresses this shortcoming by placing probability distributions over unknown quantities, enabling full posterior inference and uncertainty quantification.

In this chapter, we develop a new probabilistic modeling framework for binary outcomes: the **Bayesian p-probit model**. This model generalizes the classical probit model by adopting the p-generalized Gaussian distribution (p-GGD) as the basis for the link function. The parameter p controls the kurtosis of the distribution, interpolating between the Laplace distribution ($p = 1$), the standard normal distribution ($p = 2$), and heavier-tailed or light-tailed alternatives.

Our approach builds upon the frequentist p-probit model proposed in Munteanu et al. (2022), which was formulated as a convex optimization problem under maximum likelihood estimation. We extend this work into the Bayesian domain, where we jointly estimate the regression coefficients β and the shape parameter p using a Markov chain Monte Carlo (MCMC) approach. Specifically, we combine a data augmentation strategy with Gibbs sampling and integrate a Metropolis-Hastings (MH) step to sample the non-conjugate parameter p.

This model not only generalizes existing logistic and probit models, but also enables posterior inference on the shape of the link function itself, offering a principled way to model uncertainty in both the regression coefficients and the data-generating distribution. Our empirical evaluation includes both simulated and real-world datasets, demonstrating the effectiveness of the proposed framework in various data scenarios, particularly in the presence of tail misspecification.

In summary, in this study, we generalize the standard Bayesian probit model and its estimation into a flexible link. We present a Bayesian formulation of p-probit regression, extending the frequentist MLE framework introduced by Munteanu et al. (2022). We develop a novel Gibbs sampler for joint posterior sampling of model parameters β and the hyperparameter p, extending the classical sampler of Albert and Chib (1993) beyond the case $p = 2$. Our extensions also apply directly to logistic regression when $p = 1$.

2.2 The Bayesian p-Probit Model

Logistic and probit regression are often considered to be similar, and differ only in their tail distributions (Chambers and Cox, 1967). More general families of parametric link functions focusing on modeling the tail distribution are considered for binary regression. In a practical analysis, Albert and Chib (1993) used the t-distribution instead of the standard normal distribution in the probit model. Czado and Santner (1992) and Czado (1992) studied the importance of choosing the link function and proposed several methods for estimating the parameters of single and double tailed parametric links. Koenker and Yoon (2009) introduced the Gosset link and the Pregibon link (Pregibon, 1981), and proposed a Bayesian estimation of their respective parameters. We refer to (Stukel, 1988) for further discussion on the choice of link functions for GLMs. Czado and Santner (1992) investigated possible undesirable effects of using a misspecified link function, and pointed out that bias occurs in both parameter estimation and predicted probability. To overcome such shortcomings, we introduce in the following a parametric link function that generalizes over the logistic link similarly to Prasetyo et al. (2020), yet more specifically over the standard normal probit link, and allows us to control the tail behavior, in particular the kurtosis, of the modeling distribution by varying the parameters of the link function. We point to further, though less related, recent work on flexible link functions for binary regression based on Gaussian processes and Copulas (Li et al., 2016; Mesfioui et al., 2023), respectively.

2.2.1 p-Probit Model

In binary classification problems, we assume that the dependent variable Y_i follows a Bernoulli distribution with the probability of success $\pi_i = \mathbb{E}(Y_i) = \mathbb{P}(Y_i = 1)$. The value of π_i is conditioned on the independent variable X_i and the parameter β and

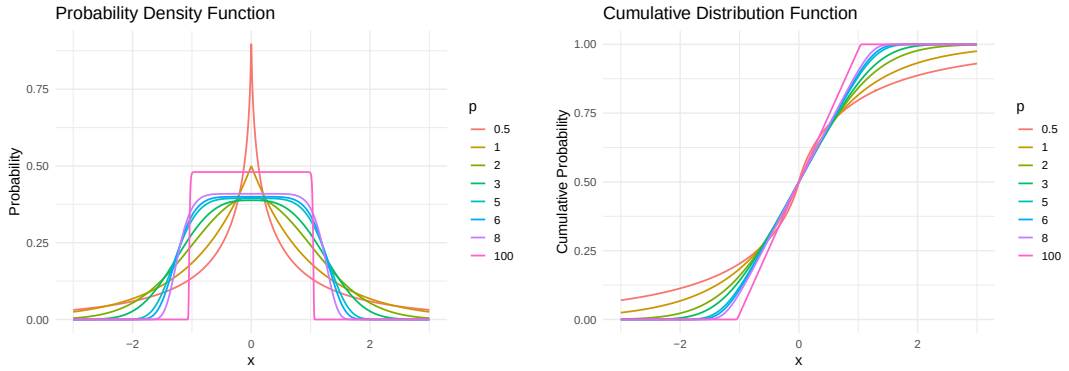


FIGURE 2.1: Probability density function (left) and cumulative distribution function (right) of the p-GGD for various values of p.

linked applying a transformation $h(\cdot)$ to the linear response such that $\mathbb{P}(Y_i = 1|X_i) = \pi_i = h(x_i\beta)$ where $x_i\beta$ is a linear combination of the observed covariates multiplied by the parameter vectors. The inverse of the transformation function $g(\cdot) = h(\cdot)^{-1}$ is called the *link function*. Logistic regression and probit regression are two common types of binary regression models. Logistic regression assumes $g(\pi) = \log(\frac{\pi}{1-\pi})$ as a link function, while the probit model uses the inverse cumulative distribution function (CDF) of the standard normal distribution $g(\pi) = \Phi^{-1}(\cdot)$, where $\Phi(\eta) = \int_{-\infty}^{\eta} \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1}{2}z^2\right)\mathrm{d}z$, for $\eta \in \mathbb{R}$. For the p-probit model, we introduce the p-generalized normal distribution, with the CDF stated as Equation 2.2.

Let $N_p(0, p^{1/p})$ denote the p-GGD with density function $\phi_p(\eta)$ given by

$$\phi_p(\eta) = \frac{p^{(1-1/p)}}{2\Gamma(1/p)} \exp(-|\eta|^p/p), \eta \in \mathbb{R}, p > 0, \tag{2.1}$$

and let $\Phi_p(\eta)$ denote the cumulative distribution function shown in Figure 2.1

$$\Phi_p(\eta) = \int_{-\infty}^{\eta} \phi_p(x)\,\mathrm{d}x, \eta \in \mathbb{R}, p > 0. \tag{2.2}$$

For the observed data $\{(x_i, y_i)\}_{i=1}^n$ with covariates $x_i \in \mathbb{R}^d$ and binary response $y_i \in \{0, 1\}$, we assume that the probability that the positive event occurs is $\mathbb{P}[Y_i = 1] = \mathbb{E}[Y_i] = \Phi_p(x_i\beta)$ and $\mathbb{P}[Y_i = 0] = 1 - \Phi_p(x_i\beta) = \Phi_p(-x_i\beta)$. The likelihood of the model is

$$\mathcal{L}(\beta|X, y) = \prod_{i=1}^{n} \Phi_p(x_i\beta)^{y_i}\Phi_p(-x_i\beta)^{1-y_i} = \prod_{i=1}^{n} \Phi_p((2y_i - 1)\cdot x_i\beta). \tag{2.3}$$

The classical MLE approach estimates the model parameter $\hat{\beta}$ maximizing the likelihood function. To this end, Munteanu et al. (2022) employed convex optimization (Bubeck, 2015) for calculating the MLE by minimizing the negative log likelihood (NLL) of the p-generalized probit model.

2.2.2 Bayesian Estimation of the p-Probit Model

In the Bayesian estimation setting we are interested in the posterior distribution of β. Imposing a prior distribution $\pi(\beta)$ over the parameter space, it can be written as

$$\pi(\beta|X,y) = \frac{\pi(\beta)\prod_{i=1}^{n}\Phi_p(x_i\beta)^{y_i}(1-\Phi_p(x_i\beta))^{1-y_i}}{\int \pi(\beta)\prod_{i=1}^{n}\Phi_p(x_i\beta)^{y_i}(1-\Phi_p(x_i\beta))^{1-y_i}\,\mathrm{d}\beta}. \tag{2.4}$$

Since sampling directly from the posterior distribution is impossible, Albert and Chib (1993) and Tanner and Wong (1987) introduced a latent variable Z to the model that has a linear relationship $Z = X\beta + \eta$ with the design matrix.

In our p-GGD extension, each latent variable Z_i is distributed according to a p-GGD with mean $x_i\beta$, scale parameter $p^{1/p}$, and shape parameter p, i.e., $Z_i \sim N_p(x_i\beta, p^{1/p})$. The joint posterior distribution can be considered as a combination of three components: the latent variable Z, the model parameter β, and the shape parameter p of the generalized probit model:

$$\begin{aligned}\pi(\beta, Z, p \mid Y) \propto \pi(\beta)\pi(p)\prod_{i=1}^{n}\{&\mathbb{1}(Z_i > 0)\mathbb{1}(y_i = 1)\\ &+ \mathbb{1}(Z_i \le 0)\mathbb{1}(y_i = 0)\}\cdot \phi_p(Z_i - x_i\beta).\end{aligned} \tag{2.5}$$

The full conditional distributions of β, Z, and p are specified below.
The posterior density of β given Z and p is

$$\pi(\beta \mid Y, Z, p) \propto \pi(\beta)\prod_{i=1}^{n}\phi_p(Z_i - x_i\beta). \tag{2.6}$$

The full conditional distributions of Z are independent where

$Z_i \mid Y, \beta$ is distributed as $\phi_p(x_i\beta)$ truncated at the left by 0 if $y_i = 1$,

$Z_i \mid Y, \beta$ is distributed as $\phi_p(x_i\beta)$ truncated at the right by 0 if $y_i = 0$.

When the parameter p of the distribution is known and fixed in advance, samples from the posterior distribution of β are obtained by repeatedly sampling Z from a truncated p-GGD and β from the full conditional distribution of β from the p-GGD with parameters $\hat{\beta}_Z, \Sigma$ and p, respectively.

For the joint posterior distribution that includes p as a variable, this becomes more complicated. In frequentist linear regression under the ℓ_p-norm, estimation methods for p have been proposed before, see Giacalone et al. (2018) and Goodman and Kotz (1973). In the generalized Bayesian framework, no direct way is known to sample from the full conditional distribution of $p \mid Y, Z, \beta$:

$$\pi(p \mid Y, Z, \beta) \propto \pi(p)\prod_{i=1}^{n}\phi_p(Z_i - x_i\beta). \tag{2.7}$$

We thus introduce another Metropolis-Hasting (MH) rejection sampling step into the Gibbs sampler for estimating the parameter p conditioned on the current values of Z

and β.

In the original Bayesian approach of Albert and Chib, 1993, the distribution of the latent variable Z is $N(X\beta, 1)$ truncated to $Z > 0$ when $Y = 1$, and to $Z \leq 0$ when $Y = 0$. Therefore, the conditional distribution of β given Y and Z can be derived from the least squares estimator of the linear model and its covariance. Using a uniform prior for β, it follows a normal distribution specified by $\beta \sim N((X^TX)^{-1}X^TZ, (X^TX)^{-1})$. Thus, by introducing the latent variable, we can obtain the posterior distribution of β using Gibbs sampling to sample in an alternating manner from the truncated normal distribution of Z given Y and β, and from the conditional distribution of β given Z.

In the p-GGD case, the error distribution generalizes to $\eta_i \sim N_p(0, p^{1/p})$. We would like to sample β from the multivariate distribution $N_p(\hat{\beta}_Z, \Sigma)$. The mean is obtained from the MLE of the linear ℓ_p regression $\hat{\beta}_Z = \arg\min_\beta \|Z - X\beta\|_p^p$. In the classical case $p = 2$ above, it is exactly the ordinary least squares estimator, while for other values of p, it involves a convex optimization under the ℓ_p-norm. The covariance matrix can be derived by $\Sigma = \tau(p)CC^T$, where C is the Cholesky factor of $(X^TX)^{-1}$ and $\tau(p) := p^{2/p}\Gamma(3/p)/\Gamma(1/p)$ (Goodman and Kotz, 1973). When the parameter p of the distribution is known and fixed in advance, samples from the posterior distribution of β are obtained by repeatedly sampling Z from a truncated p-GGD and β from the full conditional distribution of β from the p-GGD with parameters $\hat{\beta}_Z, \Sigma$ and p, respectively.

To include p into the estimation, we employ a uniform prior distribution $\pi(p)$ within some predefined interval $p \in [p_{\min}, p_{\max}]$. Our MH-sampling approach for p is inspired by the methods of Koenker and Yoon (2009), who suggest to generate the proposal candidate from a small step length deviation to the previous value. Therefore, with a given step length L, we generate p^* from the the proposal distribution $Q(p^* \mid p) \sim U[p - L, p + L]$. Then, noting that the proposal cancels by symmetry, we compute an acceptance probability upper bound $r(p^* \mid p_{t-1})$ by

$$r(p^* \mid p_{t-1}) = \frac{\pi(p^* \mid y, Z, \beta)Q(p \mid p^*)}{\pi(p_{t-1} \mid y, Z, \beta)Q(p^* \mid p)} = \frac{\prod_{i=1}^n \phi_{p^*}(Z_i - x_i\beta)}{\prod_{i=1}^n \phi_{p_{t-1}}(Z_i - x_i\beta)}. \tag{2.8}$$

Finally, we generate a random number $u \sim U[0, 1]$ from a uniform distribution. If $u \leq r(p^* \mid p_{t-1})$, then we accept by letting $p_t = p^*$. Otherwise, if $u > r(p^* \mid p_{t-1})$, we reject the proposal, in which case $p_t = p_{t-1}$. Our full sampling procedure is summarized in Algorithm 1 below.

2.3 Simulation Study

We conduct a simulation study to investigate the following research questions:

(i). First, we investigate whether manually adjusting the value of p can mitigate the issue of link misspecification. In particular, given data generated for a particular value, how well do different links perform?

Algorithm 1 p-generalized probit Gibbs sampling with Metropolis-Hastings rejection sampling algorithm

1: **for** $m = 1$ **To** $m_{\text{simulation}}$ **do**

2: Sample each latent variable Z_i from a p-GGD

$$Z_i \sim \phi_p(x_i\beta)$$

truncated to $(0, \infty)$ if $y_i = 1$, and truncated to $(-\infty, 0]$ if $y_i = 0$.

3: Given Z, we generate candidates p^* by the Metropolis-Hastings (MH) MCMC sampler with acceptance probability upper bound

$$r(p^* \mid p_{t-1}) = \frac{\pi(p^*)Q(p \mid p^*)}{\pi(p)Q(p^* \mid p)} = \frac{\prod_{i=1}^n \phi_{p^*}(Z_i - x_i\beta)}{\prod_{i=1}^n \phi_{p_{t-1}}(Z_i - x_i\beta)}.$$

At the end of those MH-MCMC iterations, we obtain a new value of p.

4: Given p and Z, we sample from the full conditional distribution of β. For a flat prior, the full conditional distribution of β is given by

$$N_p(\hat{\beta}_{Z,p}, \Sigma), \text{ where } \hat{\beta}_{Z,p} = \arg\min_{\beta} \|Z - X\beta\|_p^p \text{ and } \Sigma = \tau(p)CC^T.$$

5: **end for**

(ii). Second, we evaluate the capacity of the model to estimate the hyperparameter p. How well does the estimated posterior mean correspond to the true value of p that was simulated?

First, we randomly generate a design matrix $X \in \mathbb{R}^{N\times d}$, with the sample size of $N \in \{5\,000, 10\,000, 20\,000, 50\,000\}$ and $d = 10$, whose rows follow a multivariate normal distribution $N(\mu, \Sigma)$. The means are set to $\mu = (-2, -2, 2, 2, -3, -3, 3, 3, 0, 0)$, and the ijth element of the covariance matrix is set to $\Sigma_{ij} = 2 \cdot (0.5^{|i-j|})$. The parameter β is generated from a uniform distribution $U[-3, 3]^d$. Such a symmetric setup is intended to keep the linear combination $X\beta$ around 0, and to get a dataset that is not perfectly linearly separable. We generate response variables Y from different Bernoulli distributions, using the logit, standard normal, and p-GGD links with different p.

When estimating p, we use uniform priors specified in the corresponding sections. We do not impose priors for β[1], which reflects the worst-case scenario[2] in the estimation of β, where no prior information is available. We stress that our data simulation ensures inseparability. A degenerate likelihood or posterior distribution can thus not occur. In general, however, one should ensure a non-degenerate Bayesian model specification. A sensible informative prior $\pi(\beta)$ is given by the multivariate p-GGD (Goodman and Kotz, 1973), while uninformative choices were studied by Gelman et al. (2008) and Piironen and Vehtari (2017b).

[1] which corresponds to an uninformative uniform prior over $\mathbb{R}^d$.

[2] up to degenerate cases where an informative miss-specified prior is chosen.

2.3.1 Simulation for Fixed p

We first consider p as a fixed parameter for generating p-probit data and estimate the model using different link functions, p-probit, the classical 2-probit, logit, and cloglog, for the sake of comparison. We run the MCMC chain for 1 000 iterations. For the p-probit links, we use the corresponding predefined value of p. For the logit data, we set $p = 1$. We measure the model performances using the root mean square error (RMSE) $\sqrt{\frac{1}{n}\sum_{i=1}^{n}(\pi_i - \hat{\pi}_i)^2}$ and mean absolute error (MAE) $\frac{1}{n}\sum_{i=1}^{n}|\pi_i - \hat{\pi}_i|$. Table 2.1 shows the results for $N = 50\,000$.

Scenario	p-probit link		probit link		logit link		cloglog link	
	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE
Logit data	0.01634	0.00791	0.01236**	0.00633**	0.00767*	0.00329*	0.03670	0.02132
probit data	0.00520**	0.00174**	0.00467*	0.00156*	0.00821	0.00327	0.02251	0.01007
$p = 0.5$	0.00876*	0.00307*	0.06345	0.03964	0.04611**	0.02778**	0.07936	0.04918
$p = 1$	0.00628*	0.00258*	0.02202	0.01056	0.01267**	0.00572**	0.03998	0.02040
$p = 1.5$	0.00636**	0.00218*	0.00951	0.00376	0.00601*	0.00219**	0.02974	0.01360
$p = 3$	0.00550*	0.00176*	0.00721**	0.00268**	0.01197	0.00487	0.02070	0.00870
$p = 4$	0.00454*	0.00143*	0.01185**	0.00415**	0.01620	0.00610	0.02065	0.00832
$p = 5$	0.00717*	0.00218*	0.01341**	0.00462**	0.01756	0.00648	0.02147	0.00834
$p = 8$	0.00486*	0.00142*	0.01488**	0.00510**	0.01877	0.00680	0.02211	0.00817

TABLE 2.1: Model comparison for different fixed p data scenarios, estimated using p-probit, probit, logit and cloglog link functions for $N = 50\,000$. * indicates the smallest RMSE and MAE values, ** indicates the second smallest RMSE and MAE values.

The results provide a clear illustration of the model performances. Specifically for probit (i.e., $p = 2$) data, the (identical) p-probit and traditional probit models demonstrate the best performance, yielding significantly low RMSE and MAE values. When it comes to logit data, the p-probit model (p fixed to 1) does not keep up with the performance of the conventional logit and probit models.

Moreover, the logit model shows strong performance for cases where $p \in \{0.5, 1, 1.5\}$. Interestingly, the empirical results indicate that data generated with $p = 1.5$ is fitted even better by the logit link than data generated with $p = 1$. This is a particularly noteworthy observation: although our previous theoretical discussion suggests that the Laplace distribution ($p = 1$) is conceptually closer to the logistic distribution in terms of heavy tails, the finite-sample empirical results show that a shape parameter of $p = 1.5$ provides an even closer match to the logistic link function's behavior. Further results for other values of N are shown in Appendix B.1.3.

This simulation demonstrates that for logit data and $p \in [0.5, 1.5]$, the p-probit and logit models generally perform better in terms of RMSE and MAE, while the p-probit and probit models dominate when $p \geq 2$. Overall, the p-probit model provides highly competitive and stable performance across different tail behaviors, and should be the preferred choice whenever the true value of p in the data generation process is known or can be reliably estimated prior to the Bayesian analysis.

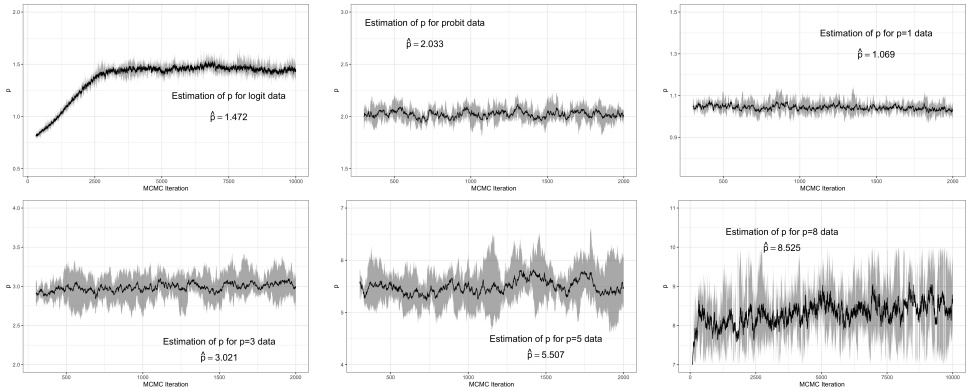


FIGURE 2.2: The MCMC chains estimating p for different simulation data. First row: logit data, probit data, and $p = 1$. Second row: $p = 3$, $p = 4$, and $p = 8$. The figures represent thinned Markov chains omitting a burn-in phase of 500 iterations.

2.3.2 Simulation for Estimation of p

Next, we use a similar setting to evaluate the estimation of p. The prior distribution of p is set to $p \sim U[0.1, 5]$ when the true value is $p \leq 5$ and $p \sim U[0.1, 10]$ when the true value is $p > 5$. Depending on the data distribution, we expect that as the model link is closer to the real link function, the regression parameters will be closer to the simulated parameters, and the residual based evaluation measures will be smaller. An MCMC convergence illustration of the simulation for $N = 50\,000$ is given in Figure 2.2. The black solid line represents the average values of the five chains, while the gray shaded area indicates the range from the minimum to the maximum values for each iteration. We initially set the number of MCMC iterations to 1 000, which was sufficient for all except the challenging cases of the logit model and $p = 8$. We adjusted the number of iterations to 10 000 to ensure convergence for the latter. All MCMC chains are simulated five times and we use the Potential Scale Reduction Factors (PSRF) of Gelman and Rubin (1992) to detect convergence of the chains. Specifically, we decide that the MCMC chains are converged when the PSRF is smaller than 1.2. Furthermore, we confirm that the posterior distribution integrates to 1 over the entire parameter space, verifying that it is a proper posterior. The results are summarized in Table B.5. From Figure 2.2 we can see that for the logit data, the model took over 5 000 iterations to converge at around $p = 1.5$. For other p-probit data with $p = 2$ and $p = 3$, the model converges very quickly. Besides, the variance of the MCMC chain increases as the value of p grows. This is due to the fact that the tail distribution is getting narrower, and the whole distribution is closer to the uniform distribution. For data with $p = 8$, the model overestimates p, and the variance is very large.However, we can see that our model becomes more accurate in estimating p with growing sample size. The summaries of all investigated scenarios are given in Table B.4.

dataset	n	d	discrete	continuous	positive	negative
Covertype	581 012	52	42	10	297 711	283 301
Credit card	284,807	29	29	0	492	284,315
Heart Disease	253 680	22	14	8	23 893	229 787

TABLE 2.2: Summary of the real-world datasets

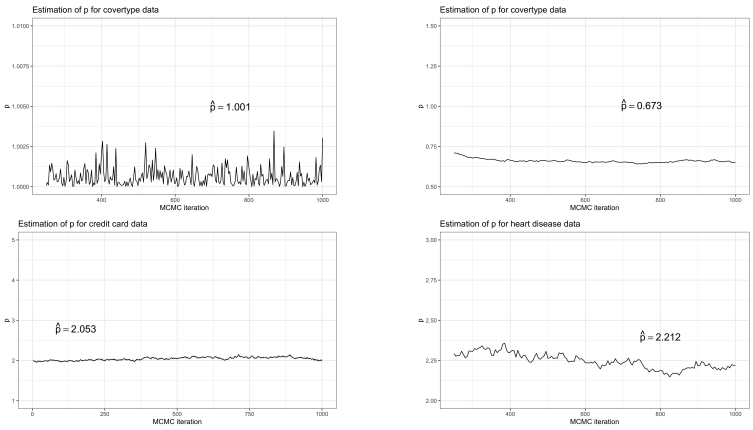


FIGURE 2.3: The MCMC chain estimation of p for the real-world dataset. Upper-left for Covertype data, p is restricted between $[1, 5]$, upper-right for Covertype data, $p \in (0, 5]$, bottom-left, credit card data $\hat{p} = 2.053$, bottom-right, heart disease data $\hat{p} = 2.21$. Burn-in phase has been taken out.

2.4 Real-World Data Application

In this section, we apply the methods to three real-world datasets, namely the covertype data, the credit card data, and the heart disease data. Their details are summarized in Table 2.2. The covertype data (Blackard, 1998) measures vegetation types in four wilderness areas in the Roosevelt National Forest in the USA. The dependent variable `covertype` in the original data has seven categories. We label type 2 as class 1 and the remaining six types as class 0, transforming into a binary problem.

The credit card data (Pozzolo et al., 2015) is a collection of fraud data on bank information. The aim is to predict default (class 1) or non-default (class 0) behavior. There are only 492 defaulters in this data. It thus suits for testing the predictive ability under extremely unbalanced conditions.

The heart disease data stems from the Behavioral Risk Factor Surveillance System (BRFSS) recorded annually by the Centers for Disease Control and Prevention (2015) in the USA. The survey collects risk behaviors with a response variable indicating whether one has ever had a heart disease.

We first run MCMC sampling for 2 000 iterations on each of the three datasets, thinned out by keeping only every fifth sample of the last 1 000 iterations, from which we estimate the posterior distribution of the coefficients. We also compare the parameters' credible intervals to confidence intervals of frequentist logistic and probit regression to measure the coverage of these methods. For the covertype dataset,

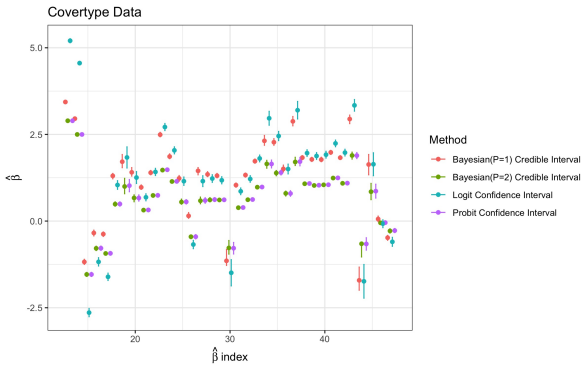


FIGURE 2.4: The credible intervals for $p = 1$ and $p = 2$ under Bayesian estimation, and the confidence intervals using logit and probit links under maximum likelihood estimation.

Figure 2.3 shows in the upper-left that p is constantly approaching 1 on the Markov chain if p is restricted to $p \in [1, \infty)$, while in the upper-right it shows p is actually close to 0.67 for $p \in (0, \infty)$. Such a value of p indicates that there is a significant difference between the distribution of the data and the model with a standard normal link. If the probit model is fitted to the data, it is likely to cause a link misspecification problem. We compare the regression coefficients for $p = 1$ and $p = 2$ in Figure 2.4. We find that for the covertype data, the results of the frequentist and the Bayesian 2-probit model are similar, both having relatively small variances and similar values. For $p = 1$, we find that most of the coefficients are closer to logistic regression, as expected.

We observe similar results for the credit card and heart disease data in Figure B.6, where we find that the coefficients estimated by frequentist logistic regression are usually close to Bayesian 1-probit, while the results of frequentist probit and Bayesian 2-probit models are even more similar.

2.5 Conclusion and Outlook

We extend the p-generalized probit model from MLE to a Bayesian framework, providing posterior estimation jointly for the parameters β and model parameter p. This also allows us to understand the distribution of the data, and parameterize the link function appropriately to adapt flexibly to the data.

In further research, it is worth investigating link functions that allow parametric control over their skewness (Prasetyo et al., 2020; Hosking and Wallis, 1997). This enables learning the data distribution even more flexibly in the presence of unbalanced data. The Bayesian treatment of such models provides quantification of uncertainty on the choice of parameters. Further, regularization and sparse regression are worth investigating (Piironen and Vehtari, 2017b; Mai et al., 2023).

Chapter 3

Large-Scale Data Reduction Based on Coresets

3.1 Introduction

With the proliferation of large-scale datasets, Bayesian methods face significant computational bottlenecks. Standard MCMC algorithms often require evaluating the likelihood function over the entire dataset at each iteration, leading to prohibitive computational costs. To address this, data reduction and compression techniques have emerged as essential tools for scaling statistical inference. Among these, the *coresets* framework is representative. The fundamental idea is to select a weighted subsample of the original data such that the model estimated on this subset approximates the full-data posterior or likelihood with theoretically bounded errors.

While initially applied to linear regression and clustering, leverage-score-based sampling methods have recently seen significant theoretical advancements. Their versatility extends beyond sample compression; for instance, in high-dimensional regimes ($p \gg n$), Teschke et al. (2024) demonstrated that generalized leverage scores can efficiently detect interaction effects in genetic data. Furthermore, regarding sampling efficiency, Munteanu and Omlor (2024a) recently established that augmenting ℓ_p sensitivity sampling with ℓ_2 leverage scores achieves optimal sample complexity bounds. This theoretical breakthrough suggests that hybrid sampling strategies involving ℓ_2 information are crucial for handling complex loss functions. Additionally, Lie and Munteanu (2024) addressed the challenges of unbounded sensitivities in Poisson regression by introducing rigorous domain shifting techniques, highlighting the necessity of handling singularities in generalized linear models.

However, bridging these theoretical advances with practical Bayesian and semi-parametric inference remains a frontier challenge. Unlike standard frequentist settings where parameters are fixed, Bayesian inference requires evaluating likelihoods over changing parameters, necessitating coreset mechanisms that remain valid across continuous domains. This chapter addresses these challenges through two distinct contributions:

The first part presents a coreset methodology for the **Bayesian p-generalized probit model** (developed in Chapter 2). A key challenge here is that the shape

parameter p is treated as a random variable during MCMC sampling. To avoid the prohibitive cost of re-computing sensitivities for every new p, we propose a *one-shot coreset* strategy. This construction uses a sensitivity distribution derived from ℓ_p leverage scores to create a single coreset valid simultaneously for a continuous range of p values. Theoretically, we provide an upper bound on the Wasserstein distance between the approximate and true posterior distributions.

The second part extends data reduction to semi-parametric multivariate analysis, specifically the **Multivariate Conditional Transform Model (MCTM)** (Klein et al., 2022). While MCTMs offer flexibility in modeling dependence structures, their log-likelihood functions involve unstable logarithmic terms and Jacobians, posing severe stability issues in large-scale optimization. Drawing from the sensitivity sampling framework for Poisson regression (Lie and Munteanu, 2024), we leverage the monotonicity property of sensitivities to design our coreset. We construct a hybrid sampling strategy that combines ℓ_2 leverage scores to capture the dominant quadratic structure with a convex hull approximation to cover the geometric boundaries of the derivatives. This approach effectively stabilizes the logarithmic terms and significantly reduces training time while maintaining estimation accuracy.

3.2 Coreset Implementation for p-Probit Models

MCMC methods used in Bayesian statistics often require a large number of iterations to complete convergence in practice. The problem becomes particularly challenging when the dataset is massively large, which aggravates calculations in *each* iteration. It is important to reduce the data in advance and obtain an approximate result efficiently in both running time and memory consumption. To this end, data compression methods, such as coresets, sketching, and random projections, received a lot of attention (Munteanu and Schwiegelshohn, 2018) and have been introduced to statistical models for tackling the limitations that arise with large-scale datasets (Munteanu, 2023). Coresets and sketching refer to a specific methodology for computing subsets comprising the most important points of the original dataset such as to guarantee a close approximation of the negative log likelihood (NLL) of the original dataset. Employing an algorithm or model to this small subset guarantees to obtain a result much more efficiently and hereby it provably approximates the result obtained from the original massive dataset to within a small controllable error bound (Munteanu, 2023). Coresets and sketching for linear ℓ_p-regression have been studied, by Clarkson (2005), Dasgupta et al. (2009), Sohler and Woodruff (2011), and Woodruff and Zhang (2013), who focus on linear regression $Y = X\beta + \eta$, where η follows a p-GGD. To approximate the linear model within $(1+\varepsilon)$ error, they obtain a coreset by subsampling a few observations using the ℓ_p norm of rows of a *well-conditioned* basis. Munteanu et al. (2022) extended this approach to *generalized* linear models, in particular for the p-generalized probit model for dichotomous data. Here, we extend this data subsampling method even further from MLE to the Bayesian p-generalized

probit model, even beyond fixed p. This continues recent research on improving the efficiency of *Bayesian* methods. Several data reduction methods have been studied, such as streaming (Broderick et al., 2013; Campbell et al., 2015), subsampling for MCMC (Quiroz et al., 2019; Ahn et al., 2012; Bardenet et al., 2014; Bardenet et al., 2017), random projections (Geppert et al., 2017), Merge & Reduce (Geppert et al., 2020; Ding et al., 2023), consensus Monte Carlo (Rabinovich et al., 2015; Scott et al., 2016), sketching (Munteanu et al., 2021; Munteanu et al., 2023), and coresets (Huggins et al., 2016; Campbell and Broderick, 2018; Campbell and Broderick, 2019).

In summary, we extend the coreset constructions of Munteanu et al. (2022) developed for frequentist fixed p-probit to their Bayesian counterpart, allowing for estimation of p, while enhancing scalability of our sampler for large data.

3.2.1 Coresets for the Bayesian p-Probit Model

In this section, we introduce the concept of coresets and briefly present the coreset construction method of Munteanu et al. (2022) for the p-generalized probit model. We use the ℓ_p leverage scores to approximate the sensitivity (or importance) of the observations for preserving the p-generalized probit model. These scores are used to specify a distribution for subsampling the data. We call the subsampled dataset a *coreset* and run our Bayesian estimation on it. We argue that the approximation properties of coresets for preserving the likelihood continue to hold when adding an arbitrary prior to the likelihood. The same construction thus approximates the posterior distribution of our Bayesian p-generalized probit model. For providing a theoretical guarantee on the proximity of the posterior distribution induced by the coreset compared to the one obtained from the original data, it thus suffices to analyze the respective likelihood distributions. We quantify their *Wasserstein*-distance, which we bound to within small constant factors of the NLL and its entropy.

We estimate the posterior distribution of the model parameter β and the link function parameter p using Bayesian methods. However, it is notorious that MCMC methods require a substantial number of iterations for the Markov chain to converge. In addition, the MH method that is used to estimate p in each iteration also requires hundreds of iterations to yield a new sample of p. Therefore, it takes considerable time to estimate the model and it requires a substantial amount of memory when dealing with large-scale data. We thus introduce data reduction methods for Bayesian p-generalized probit models. We build on the coreset construction method of Munteanu et al. (2022), which has been shown to approximate the NLL for p-probit models very accurately. In the following definition, f denotes the NLL corresponding to (2.3) where labels are folded into the design matrix X[1], and f_w denotes its weighted version.

Definition 3.2.1 (Coreset). *Let $X \in \mathbb{R}^{n\times d}$ be the original design matrix, let f be the loss function we aim to optimize. We define a weighted ε-coreset $C = (X', w)$ for f to consist of a*

[1] since the factors $(2y_i - 1)$ always appear together with x_i in Equation (2.3) and its logarithm

matrix $X' \in \mathbb{R}^{k\times d}$ *and a weight vector* $w \in \mathbb{R}^k_{>0}$*, such that*

$$\forall \beta \in \mathbb{R}^d \colon |f_w(X'\beta) - f(X\beta)| \leq \varepsilon \cdot f(X\beta).$$

We want X' to be a subset of the original dataset of significantly reduced size $k \ll n$. On this subset, our algorithm runs much more efficiently while the error remains bounded by a small ε-fraction of the original value. It was argued in Munteanu et al. (2022) that coresets for the p-probit model do not exist in general, cf. (Huggins et al., 2016; Munteanu et al., 2018). To quantify the obtainable size of a coreset for a given dataset X, a data dependent parameter $\mu(X)$ was introduced.

Definition 3.2.2 (μ-complexity, Munteanu et al., 2018; Munteanu et al., 2022)**.** *For a dataset* $X \in \mathbb{R}^{n\times d}$ *and a fixed* $p \geq 1$ *define*

$$\mu_p(X) = \sup_{\beta\in\mathbb{R}^d\setminus\{0\}} \frac{\sum_{x_i\beta>0} |x_i\beta|^p}{\sum_{x_i\beta<0} |x_i\beta|^p}.$$

$X \in \mathbb{R}^{n\times d}$ *is called* μ*-complex if* $\mu_p(X) \leq \mu < \infty$.

Intuitively, μ-complexity measures quantitatively the separability of a dataset, and has an interpretation as the balance between probability mass attributed to the two different classifications, cf. Munteanu et al. (2018).

In the Bayesian framework, the likelihood can be denoted $\exp(-f(X\beta))$, since $f(X\beta)$ is the NLL. The posterior distribution in Equation (2.4) additionally comprises a prior $\pi(\beta)$ and from this we obtain the marginal likelihood

$$m = \int \exp(-f(X\beta))\pi(\beta)\,\mathrm{d}\beta.$$

Huggins et al. (2016) argued that if $C = (X', w)$ satisfies the coreset bound on the NLL then the guarantee propagates to the negative marginal log likelihood, i.e.,

$$|\log m - \log \tilde{m}| \leq \varepsilon\,|\log m|,$$

where $\tilde{m}$ is obtained similarly to m by replacing the data by the coreset. This was proven by a direct application of Jensen's inequality. A more intuitive explanation is that marginalizing by a normalized prior corresponds to a convex projection, which can only contract but not dilate distances between the respective distributions. The main takeaway from this is that we can focus in the remainder on analyzing the distance between the likelihoods derived from the original data and from the coreset. To this end, we aim to show that if the NLL $f(X\beta)$ for a dataset X in p-probit regression is approximated by the coreset X' as in Definition 3.2.1, then the deviation of the approximated likelihood from the original likelihood can be bounded in terms of their Wasserstein distance of order p.[2]

[2]We note that a result similar to the case $p = 1$ can be obtained verbatim for logistic regression.

3.2.2 Wasserstein Distance Bound

As we have noted above, it suffices to bound the Wasserstein distance of the likelihood and its approximation obtained from using a coreset. To this end, we require an ℓ_p analogue $\sigma_p^{\min}(M) = \inf_{x \in \mathbb{R}^d \setminus \{0\}} \frac{\|Mx\|_p}{\|x\|_p}$ of the smallest singular value of a matrix (Golub and Loan, 2013). We have the following result:

Theorem 3.2.3 (Wasserstein bound, Ding et al., 2024)**.** *Given $X \in \mathbb{R}^{n \times d}$, let (X', w) be an ε-coreset for the p-probit or logistic loss (in which case the lemma applies with $p = 1$). Let $q \propto \exp(-f(X\alpha))$ and $q' \propto \exp(-f_w(X'\beta))$. Let Z_1^m, Z_2^m be the mode-centered versions of the random variables $Z_1 \sim q$ and $Z_2 \sim q'$. Also let μ, ν minimize $f_w(X'\eta), f(X\eta)$, respectively, over $\eta \in \mathbb{R}^d$. Then the Wasserstein distance of order p is bounded by*

$$\mathcal{W}_p(q, q') \leq 2 \frac{(p(2+\varepsilon)(1+\mu))^{1/p}}{\sigma_p^{\min}(X)} \left(f(X\nu) + \mathbb{E}_q\left[f(XZ_1^m)\right]\right)^{1/p}.$$

We note that Theorem 3.2.3 bounds the Wasserstein distance to within constant factors of the optimal NLL and the related entropy. As we discussed above, the factor μ is necessary to allow sublinear data summaries to exist. While additive errors of ε are desirable (Geppert et al., 2017), it is known that only constant factors, in particular $(2+\varepsilon)$, can be achieved when $p \neq 2$. The last term in our bound corresponds to the entropy of the parameter distribution. By the maximum entropy property of the multivariate normal distribution, this term can be further bounded by the trace of the covariance matrix of the normal distribution $\operatorname{tr}\left((X'X)^{-1}\right)$.[3] The proof of Theorem 3.2.3 is deferred to the appendix. Here, we give a high level intuition. The Wasserstein distance can be decomposed into two components: the p-norm distance of the respective MLE estimators, i.e., the modes of the distributions, and the p-norm measure of variability of the mode-centered distributions. The first component can be bounded by relating the p-norm to roughly μ times the NLL and leveraging the proximity guaranteed by the coreset. Bounding the second component requires a coupling between random variables drawn from the two distributions. The coupling is constructed to be a deterministic bijection between vectors with a similar NLL, which implies that they lie close to each other. Taking the expectation over the distances between coupled vectors completes the bound.

3.2.3 Sampling Algorithm for Bayesian p-Probit Coresets

In this section we give details on how to construct coresets for Bayesian p-probit regression for a given fixed p. Then we extend the construction to hold simultaneously for the several values of p that our Gibbs sampler evaluates. We adopt the so-called *sensitivity sampling* framework (Feldman and Langberg, 2011). It has become a popular gold standard across the coreset literature, used for instance in Huggins

[3] up to normalizing constants and factors depending on μ and $\tau(p) = p^{\frac{2}{p}} \frac{\Gamma(3/p)}{\Gamma(1/p)}$.

et al. (2016), Munteanu et al. (2018), and Munteanu et al. (2022).[4] The *sensitivity* of point $x_i, i \in [n] := \{1, 2, \ldots, n\}$ is defined as $\zeta_i = \sup_{\beta \in \mathbb{R}^d} \frac{f(x_i\beta)}{f(X\beta)}$ and measures its importance for preserving the NLL. We subsample the data proportional to the sensitivity measure, which guarantees that the structure and distribution of the data is preserved accurately, even in the presence of few very important points. The number of samples required to obtain a certain error bound will be specified later, and is dominated by the *total sensitivity* $Z = \sum_{i=1}^{n} \zeta_i$. For most problems, however, the true sensitivity of each data point is as difficult to compute as solving the original large problem. Therefore, we calculate more efficiently an approximation of the sensitivities such that $S = \sum_{i=1}^{n} s_i \geq \sum_{i=1}^{n} \zeta_i = Z$. The overestimation needs to be controlled carefully since the necessary sample size grows to $S \geq Z$. Here, we use the so-called ℓ_p-leverage scores, which are defined as $u_i^{(p)} = \sup_{\beta \neq 0} \frac{|x_i\beta|^p}{\|X\beta\|_p^p}$. They can be calculated from so called well-conditioned bases (Dasgupta et al., 2009), which we approximate very efficiently using sketching techniques of Woodruff and Zhang (2013). Finally, we overestimate the sensitivities by $\zeta_i \leq c\mu \left(\frac{1}{n} + u_i^{(p)}\right) =: s_i$ and finally take a random subsample of the data proportional to these scores.

Theorem 3.2.4 (Munteanu et al., 2022)**.** *For a μ-complex dataset $X \in \mathbb{R}^{n \times d}$ and a fixed link parameter $p \in [1, \infty)$, with constant probability, Algorithm 6 computes an ε-coreset (X', w) for p-probit regression of size $k = O(\frac{S}{\varepsilon^2}(d \log(\varepsilon^{-1}\mu) \log S)$ in two passes over the data, where $S = O(\mu d)$ for $p = 2$, $S = O(\mu d^p (d \log d)^2)$ for $p \in [1, 2)$, and $S = O(\mu d^{2p} (d \log d)^2)$ for $p \in (2, \infty)$.*

For any *fixed* value of p we could readily run our Gibbs sampler on the resulting coreset. However, our extension includes an MH sampling step to incorporate p as a variable into the sampling process. To tackle this difficulty, we need to generate coresets applicable to a wide range of $p \in [p_{\min}, p_{\max}]$ without knowing the particular values in advance. A solution to this problem was provided in the context of clustering by Bachem et al. (2018), called *one-shot* coresets. We adapt their approach to work for Bayesian p-probit regression. The idea is to cover the interval $[p_{\min}, p_{\max}]$ by an exponential grid $P = \{p_{\min}, (1+\Delta)p_{\min}, (1+\Delta)^2 p_{\min}, \ldots, p_{\max}\}$. After calculating the sensitivities $s_i^{(p)}$ for each $p \in P$, the new sensitivity upper bound for x_i becomes $s_i = \sum_{p \in P} s_i^{(p)}$. The advantage of the resulting one-shot coreset is that it satisfies the coreset guarantee simultaneously for several values of $p \in [p_{\min}, p_{\max}]$, even when $p \notin P$, without explicitly fixing them in advance. The disadvantage is that we need to calculate the sensitivities for all $p \in P$. However, we prove that $\Delta = \frac{1}{\log(n)}$ is a suitable increment, which limits the size $|P|$ to a small logarithmic amount $l = \frac{\log(p_{\max}/p_{\min})}{\log(1+\Delta)} = O(\log(p_{\max}/p_{\min}) \log(n))$.

Theorem 3.2.5. *Let $1 \leq p_{\min} < p_{\max} < \infty$. Let $X \in \mathbb{R}^{n \times d}$ be μ-complex for any fixed $p \in [p_{\min}, p_{\max}] \subset [1, \infty)$. Let $Q = \{p_1, \ldots, p_m\}$ be an arbitrary subset of $[p_{\min}, p_{\max}]$. Then, with constant probability, we can compute an ε-coreset (X', w) for p-probit regression*

[4] We note that there are recent improvements using different importance measures (Mai et al., 2021; Woodruff and Yasuda, 2023).

of size $k = O(\frac{S}{\varepsilon^2}(d\log(\varepsilon^{-1}\mu)\log S + \log(m)))$ *in two passes over the data that is valid simultaneously for all* $p \in Q$*, where*

$$S = \begin{cases} O\left(\mu d^{p_{\max}}(d\log d)^2(\log(n)\cdot\log(p_{\max}/p_{\min}))\right), & \text{if } p_{\max} \in [1,2] \\ O\left(\mu d^{2p_{\max}}(d\log d)^2(\log(n)\cdot\log(p_{\max}/p_{\min}))\right), & \text{if } p_{\max} \in (2,\infty). \end{cases}$$

3.2.4 Simulation for Evaluating Coresets

The purpose of this experiment is to observe how well different coresets preserve the original data distribution, and to explore the performance of the classifier on these datasets. Figure 3.1 shows the results of this simulation. The ℓ_2-leverage scores

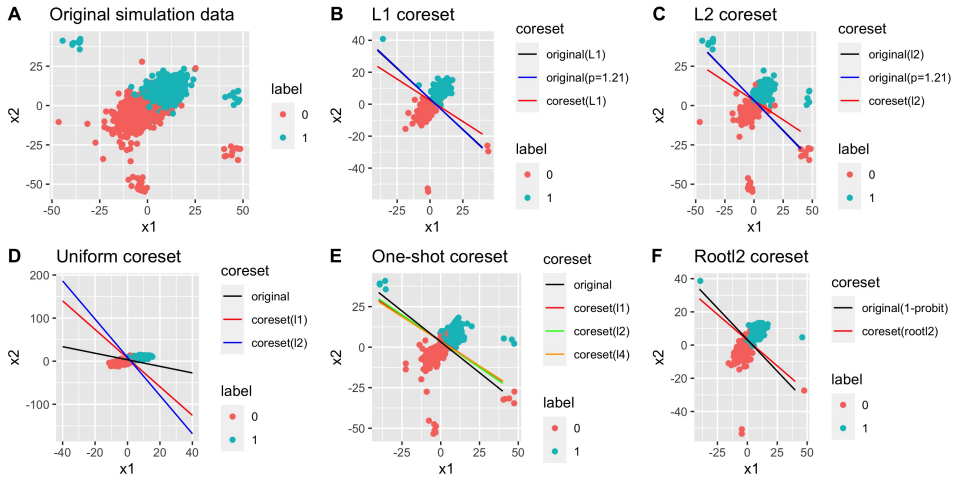


FIGURE 3.1: The performance of different coreset approaches on a simulated dataset, and models regressed using different Bayesian p-probit models on coresets. Original data: $n = 100\,000$, coreset: $n = 100$.

have higher sensitivity to points far from the central distribution than the ℓ_1-scores. The one-shot coreset also represents the original distribution of the data well. The `rootl2` method, designed by Munteanu et al. (2018) for logistic regression, uses the ℓ_2 norm (instead of its square), which performs more similarly to the ℓ_1-scores. Uniform sampling performs clearly the worst, with no ability to identify any of the extremal points. Turning our attention to the linear classifiers, `original` in the legend represents the choice of fixed values $p = 1$ or $p = 2$, respectively. The model from the uniform sample is far from the original model. For one-shot coresets, we compare several different cases. As p grows the model becomes more sensitive to outliers.

For evaluating the coresets, we generate a classification dataset of $N = 100\,000$ as described above, manually set a few outliers, and subsample coresets of the data with the parameter p set to different values. Then we run the MCMC sampler with fixed p on the subsampled dataset.

In a second stage, we incorporate the data reduction methods. We sample according to different sensitivity distributions and run the MCMC sampler on the resulting coresets. We compare 2-probit sampling, the one-shot coreset, and square

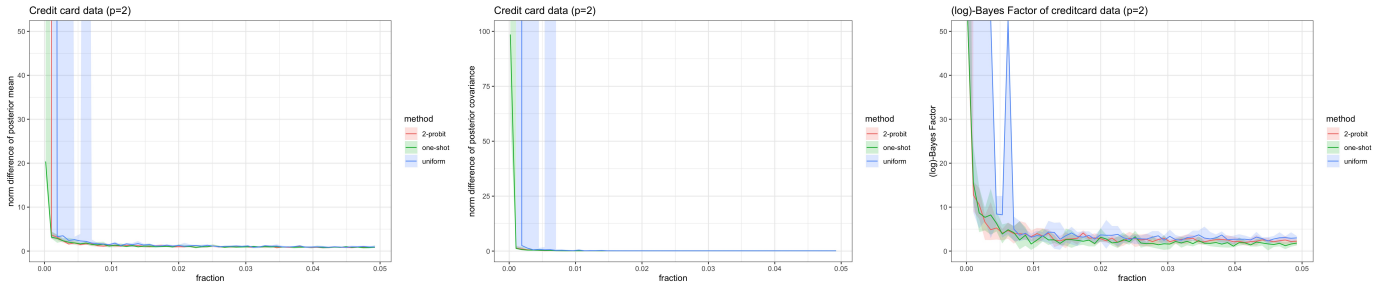


FIGURE 3.2: The approximation ratio measured by ℓ_2 norm difference of posterior mean and posterior covariance, and Bayes factor for credit card data.

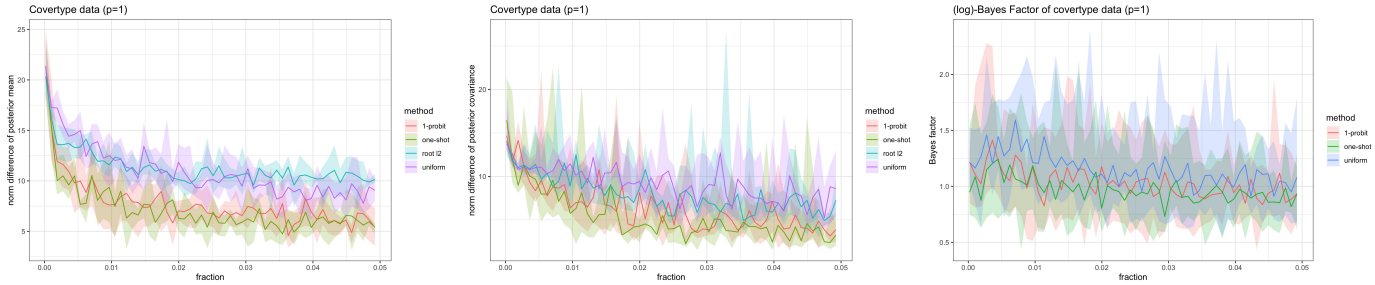


FIGURE 3.3: The approximation ratio measured by ℓ_2 norm difference of posterior mean and posterior covariance, and Bayes factor for covertype data.

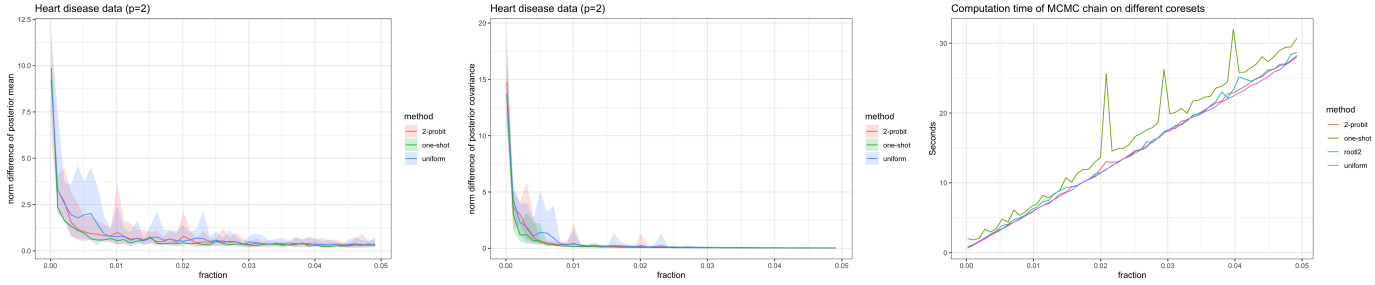


FIGURE 3.4: The approximation ratio measured by ℓ_2 norm difference of posterior mean and posterior covariance for heart disease data, and computation time.

root ℓ_2-leverage scores for $p = 1$ with uniform sampling as a baseline. We use the ℓ_2-norm posterior mean difference $\|\mu_\beta - \tilde{\mu}_\beta\|_2$, the spectral norm posterior covariance difference $\|\Sigma_\beta - \tilde{\Sigma}_\beta\|_2$[5], and the Bayes factor as our measures for the quality of approximation.

The results are shown in Figure 3.4. On the credit card dataset, the one-shot coreset method converges faster than the 2-probit method, and also better than the uniform sampling method. For the covertype dataset, we observe a similar phenomenon in the case $p = 1$, which outperforms uniform sampling and root ℓ_2 sampling along with p-probit. On the heart disease dataset, we rarely see a difference. All sampling methods converge quickly to almost optimality.

We also record the time required for constructing different coresets and running the MCMC methods. For 2 000 iterations, the MCMC algorithm takes a few hours on the original dataset, but it takes only 30 seconds on our coreset. For extremely large datasets, it is possible to encounter situations where the original dataset is too large to be analyzed at all. However, the size of our coreset grows only logarithmically, which is a huge advantage.

3.2.5 Conclusion

To improve the time efficiency of our MCMC method, we believe it is critical to compress the data. To this end, we extend and further develop coreset methods by sensitivity subsampling with approximated ℓ_p leverage scores and adapt them to the Bayesian p-probit model. The flexibility of the coresets is further extended by introducing one-shot coresets for ℓ_p regression. We compare our methods on real-world data with uniform sampling and root ℓ_2 subsampling, and observe superior performance of our new methods. In theory, we prove that the approximation error to the true posterior is small in terms of the Wasserstein distance.

Being computationally demanding, the Bayesian framework can hugely benefit from further investigating and integrating coreset methods. In the deep learning community, Bayesian Neural Networks (BNNs) are adept at handling uncertainty and providing probabilistic predictions (Goan and Fookes, 2020). Integrating coreset methods can also significantly enhance BNNs by allowing for efficient posterior estimation with reduced data without sacrificing accuracy and is a promising endeavor for future research.

3.3 Data Reduction for Multivariate Conditional Transformation

In the context of big data, efficient and scalable non-parametric or semi-parametric regression analysis and density estimation are of crucial importance to the fields of statistics and machine learning. However, available methods are relatively limited

[5] corresponding to the location and variability components captured by the Wasserstein distance.

in their ability to handle large-scale and high-dimensional data. We address these issues by developing a novel coreset construction for multivariate conditional transformation models (MCTMs) to enhance their scalability and training efficiency. To the best of our knowledge, these are the first coresets for semi-parametric distributional models. Our approach yields substantial data compression via importance subsampling, while ensuring that with high probability the log-likelihood remains within prescribed multiplicative error bounds of $(1 \pm \varepsilon)$ for any fixed $\varepsilon > 0$ and thereby maintains statistical model accuracy. Compared to conventional full-parametric models, where coresets have been incorporated before, our semi-parametric approach exhibits enhanced adaptability, particularly in scenarios where complex distributions and non-linear relationships are present, though not fully understood. To effectively address potential numerical problems associated with normalizing logarithmic terms in the log-likelihood, we follow a geometric approximation strategy, based on the convex hull of points derived from input data. This ensures feasible, stable, and accurate inference in scenarios involving large amounts of data. Numerical experiments demonstrate that the proposed algorithm notably enhances computational efficiency when handling large and complex datasets, thus laying the foundation for a broad range of applications within the statistics and machine learning communities, particularly through an established connection of MCTMs to normalizing flows.

In today's era of Big Data, the vast volume and increasing complexity of data in many real-world applications pose new challenges to statistics and machine learning. Traditional methods require intense computation times on large data and straightforward solutions such as uniform subsampling may lead to infeasible solutions. Also, they often rely on strong modeling assumptions that are too restrictive to capture complicated phenomena accurately. These limitations demand efficient and scalable techniques for flexible multivariate models.

To address the need of scalable learning, the method of *coresets* received a lot of attention in recent years. It aims to sample or select a relatively small but representative subset of the original data that can be used to accurately approximate the objective function for any solution, thereby significantly reducing computing, memory and storage requirements (Phillips, 2017).

However, most of the existing coreset literature focused on clustering, linear regression, or generalized models that can handle only relatively simple and limited distributional assumptions. Little research has been conducted on more flexible non-parametric, semi-parametric or highly non-linear multivariate models. This gap becomes more critical as the machine learning and statistics communities increasingly rely on complex distributional regression and estimation methodologies including not only multivariate features but also multivariate outcomes.

3.3.1 Brief Introduction to MCTMs

In this paper, we specifically focus on multivariate conditional transformation models (MCTMs; Klein et al., 2022), which have the unique advantage of modeling both

unconditional and conditional distributions, non-linear dependence structures, and flexible effects of features. The basic idea of MCTMs is to model marginal distributions non-parametrically via monotonic transformation functions. The multiple univariate distributions are then combined into a multivariate distribution via a copula, that contributes all information required to model the multivariate dependence structure. The well-known Sklar's Theorem (Sklar, 1959) states that every multivariate distribution can be composed and represented in that way. Of course, representing and calculating such a model explicitly in a lossless way for arbitrary multivariate distributions may require solving infinite-dimensional numerical problems which is arguably not practical.

MCTMs thus formulate a semi-parametric model that rely on the monotonic univariate transformations being approximated as a linear combination of Bernstein polynomial basis functions whose choice ensures monotonicity of the composed functions. The dependence structure between the univariate output components is imposed by a Gaussian copula (Song, 2000). We note that those modeling choices may be replaced by different functional basis families and different copulas, but we stress and demonstrate that it is not too restrictive as it may seem. For instance although Gaussian copulas are centrally symmetric, the resulting model need not be symmetric unless all marginal distributions are symmetric as well, see for instance the density contour plots in Figures 3.5 to 3.9 in Section 3.3.5.

For the sake of a concise presentation, we restrict ourselves to the unconditional case of density estimation without features, see Klein et al. (2022) for further details, including the conditional regression case. The goal is to learn the density $f_Y(y)$ of a J-dimensional random vector $Y = (Y_1, \ldots, Y_J)^T \in \mathbb{R}^J$ and its distribution function $F_Y(y) = P(Y \leq y) = P(h(Y) \leq h(y))$.

This involves a bijective, strictly monotonically increasing transformation function $h : \mathbb{R}^J \to \mathbb{R}^J$ estimated from the data, which maps Y to another random vector $Z \in \mathbb{R}^J$ that follows some convenient distribution. In particular, the common marginal distribution P_Z of the components $Z_j \sim P_Z$, $j \in [J]$ is a modeling choice and is assumed to not depend on the data. It thus holds that

$$h(Y) = (h_1(Y), \ldots, h_J(Y))^T \overset{d}{=} (Z_1, \ldots, Z_J)^T = Z \in \mathbb{R}^J .$$

Furthermore, it is assumed that each component h_j can be expressed as a linear combination of strictly monotonically increasing marginal transformation functions $\tilde{h}_j : \mathbb{R} \to \mathbb{R}$ of the form

$$h_j(y_1, \ldots, y_j) = \lambda_{j1}\tilde{h}_1(y_1) + \ldots + \lambda_{jj}\tilde{h}_j(y_j),$$

where λ_{jl}, $l \leq j$ are the unrestricted entries of a $J \times J$ lower triangular matrix Λ, such that $\Sigma = \Lambda^{-1}(\Lambda^{-1})^T$ is the covariance matrix of Z. Hence, Λ describes the conditional dependencies between the J components, and $\tilde{h}_j(y_j), j \in [J]$, are scaled

marginal transformation functions each of which carry an intercept term and that allow the generation of arbitrary marginal distributions. The marginal transformations are modeled semi-parametrically via some basis functions $a_j : \mathbb{R} \to \mathbb{R}^d$ and basis coefficients $\vartheta_j \in \mathbb{R}^d$. The a_j are chosen, for example, as Bernstein polynomials which allow for a convenient way to impose monotonicity. The unknown model parameters are collected in the vector $\theta = (\vartheta^T, \lambda^T)^T$. For consistency with other coreset literature, we consider *minimizing* the negative log-likelihood, or specifically the sum of loss contributions of each data point $y_i \in \mathbb{R}^J, i \in [n]$, to the negative log-likelihood, which is equivalent to calculating the maximum likelihood estimator

$$\hat{\theta}_n = \text{argmin}_{\theta=(\vartheta^T,\lambda^T)^T} \sum_{i=1}^{n} \frac{1}{2} \sum_{j=1}^{J} \left(\sum_{J=1}^{j-1} \lambda_{jJ} a_J(y_{ij})^T \vartheta_J + a_j(y_{ij})^T \vartheta_j \right)^2 - \log(a_j'(y_{ij})^T \vartheta_j), \tag{3.1}$$

see Klein et al. (2022) for details and a derivation of the log-likelihood and its gradients.

MCTMs can be fitted to data by optimizing the linear basis coefficients ϑ of univariate transformation models and the covariance structure of the Gaussian copula, which is parameterized through the unrestricted entries of the lower triangular matrix of the modified Cholesky factor Λ. In that regard, MCTMs can be seen as a multivariate extension of univariate conditional transformation models (CTMs; Hothorn et al., 2014) to handle multi-dimensional outputs and their dependence structure.

As can be seen in Equation (3.1), the log-likelihood function of an MCTM contains both quadratic and logarithmic parts in terms of the parameters to optimize. The quadratic part is numerically tractable and can be reformulated in a way that enables handling within the framework of ℓ_2-subspace embeddings (Woodruff, 2014), in particular via ℓ_2 leverage score subsampling (Drineas et al., 2006; Rudelson and Vershynin, 2007). The logarithmic part is known to be unstable and computationally burdensome when optimizing models to fit large-scale datasets. Considering sensitivity sampling for similarly structured loss functions has required special efforts and novel techniques in recent work on Poisson regression models (Lie and Munteanu, 2024). For the MCTMs considered here, the situation is even more aggravated since the terms that show up in the log-likelihood do not directly depend on the plain data, but instead on the polynomial basis transformations in the quadratic part and on their derivatives in the logarithmic part.

3.3.2 Our Goals and Contributions

In this paper, we develop the first coreset approximation framework for MCTMs, aiming to achieve the following goals.

1. **Significant computational reduction while guaranteeing an accurate log-likelihood approximation.** By combining the ℓ_2 leverage score sampling with a

geometric convex-hull approximation, we provide a $(1 \pm \varepsilon)$-factor log-likelihood approximation for MCTMs using only a small fraction of the original data.

2. **Solving the problem of unstable values of logarithmic terms.** We construct coresets separately for $a(y)$ that represent polynomial basis transformations of plain data y, and their respective derivatives $a'(y)$. This allows to use standard methods for the former part while avoiding certain extreme directions where the corresponding logarithms tend to infinity or lead to infeasible solutions.

3. **Compatibility with other popular probabilistic models:** MCTMs build a flexible semi-parametric framework that has connections to contemporary machine learning methods such as deep learning and normalizing flows (Kobyzev et al., 2021; Papamakarios et al., 2021). Our coresets bring significant speedup and scaling possibilities to subsequent learning tasks that build upon MCTMs.

Theoretically, we develop a leverage score and sensitivity sampling analysis along with a convex hull approximation scheme to provide a small subset of data. We prove that it approximates the log-likelihood to within a $(1 \pm \varepsilon)$ factor relative to the full dataset.

Empirically, we demonstrate through simulation experiments and real data cases that the method offers significant benefits in large-scale data settings from multiple perspectives, such as computational efficiency and scalability, fitting accuracy, likelihood ratio, and estimation bias.

We summarize the main contributions of this paper as follows:

1. **The first coresets for the MCTM framework.** To the best of our knowledge, we are the first to develop a coreset construction method for MCTMs, which fills the gap of data compression techniques for highly flexible semi-parametric model fitting methods for multivariate (conditional) distributions.

2. **Convex hull based stabilization of logarithmic normalizing terms.** Potential numerical issues of logarithmic terms in the log-likelihood are eliminated by a convex-hull approximation of the derivative $a'(y)$ of transformed data y. The properties of this geometric approximation are investigated theoretically and empirically.

3. **Illustrative large-scale data scenarios.** We illustrate through numerical simulations and data from real-world applications that the new method operates significantly more efficiently than using the original large-scale data. Moreover, it also retains almost the same accuracy as the original full-data estimation illustrating our theoretical guarantees in practice.

3.3.3 Related Work

Probabilistic transformation models. Originally restricted to univariate outputs, Hothorn et al. (2014) proposed a boosting approach for estimating a monotonically

invertible mapping that transforms the output variable to a convenient, data independent distribution (e.g., standard normal). The mapping, called transformation function, is a semi-parametric, monotonic spline approximation that can depend on input variables in a flexible manner. This framework indirectly models the full conditional distribution of the output and was later extended towards a fully likelihood-based approach, known as *most likely transformations* by Hothorn et al. (2018). Klein et al. (2022) built on this approach and proposed MCTMs, which can estimate the joint conditional distribution of a multivariate response directly based on the features, using likelihood inference. The original work of Klein et al. (2022) considers datasets of up to ten thousand observations and up to ten-dimensional outputs. In contrast to previous probabilistic methods, MCTMs directly estimate the joint distribution function of a high-dimensional output vector, avoiding the loss of information that would result from modeling only the mean or certain quantiles.

Transformation models are closely related to normalizing flows (NFs; Kobyzev et al., 2021; Papamakarios et al., 2021), which are popular in machine learning. We elaborate on this connection in Appendix A.2.7. The main idea of both approaches is to apply a data-driven transformation such that the transformed variable follows a standard normal or some other convenient distribution. However, NFs transform their input via some kind of deep neural network as a black box, whereas our model implements a semi-parametric transformation using a Bernstein polynomial basis, which remains analytically tractable.

NFs are most commonly employed as flexible variational distributions in machine learning, whereas our focus is on modeling the distribution of multivariate data along with their dependence structure. MCTMs have also been extended towards regression, where density estimation is conditioned on feature variables. The linear underlying structure of transformation functions of MCTMs have the great advantage of enhancing interpretability compared to the neural network approach of NFs. For instance, in MCTMs we can directly interpret the dependence structure between components of the outcome and clearly separate these effects from those modeling the marginal distributions. Further, MCTMs are likelihood-based and therefore yield access to confidence intervals via bootstrapping techniques.

Coresets for large-scale statistical modeling. The main idea of the coreset method is to select the *most important* subset of data based on their contribution to the objective function (e.g., log-likelihood or loss) and reweight them, so that training the model on this subset closely approximates the results of the original dataset. Early works on coresets focus on parametric models, such as linear regression (Clarkson, 2005; Drineas et al., 2006; Dasgupta et al., 2009) or generalized linear models (Munteanu et al., 2018; Molina et al., 2018; Munteanu et al., 2022; Lie and Munteanu, 2024). These also make up the bulk of work of coresets for statistical models.

Inferring whole probability distributions received relatively little attention in the coreset literature. Among them are coresets for univariate kernel density estimation

(Phillips and Tai, 2018; Phillips and Tai, 2020; Charikar et al., 2020). Bayesian coreset based models focus on multivariate parameter distributions rather than multivariate outcomes (Geppert et al., 2017; Huggins et al., 2016; Campbell and Broderick, 2018; Ding et al., 2024). To our knowledge, the only attempt to learn multivariate data distributions using coresets for scalability has been made in the context of dependency networks that infer inter-variable dependencies with a graphical structure (Molina et al., 2018). This approach is again composed of multiple parametric models.

In summary, the coreset method has shown great potential in improving the efficiency of algorithms for regression models and Bayesian inference. However, we stress that these methods have to date been mainly developed for parametric models (e.g. linear/generalized linear models), and, to our knowledge, only little work has considered complex semi-parametric distribution models for potentially high-dimensional multivariate outputs. Our attempt to combine coresets with MCTMs is, to the best of our knowledge, the first of its kind, leveraging the flexibility of multivariate distributional regression with efficient data compression, and fills the gap in the methodology for inference of large-scale semi-parametric multivariate distributional models.

3.3.4 MCTM Coreset Construction

Considering the MCTM model whose negative log-likelihood function can be defined as in Equation (3.1), the goal of the coreset approach is to obtain a small, weighted subset of data $C \subset D$ (following the general ε-coreset framework established in Definition 3.2.1), such that the approximation of the original negative log-likelihood function is bounded within a factor of $(1 \pm \epsilon)$. The full details and proofs are deferred to Appendix A.2. After applying the basis functions a and their derivatives a' to the raw data, we can assume that for $(i, j) \in [n] \times [J]$ we are given data points $a_{ij} \in \mathbb{R}^d$ and $a'_{ij} \in \mathbb{R}^d$. We use $A, A' \in \mathbb{R}^{nJ \times d}$ to denote the corresponding data matrices. Now for $i \in [n], j \in [J]$ consider $f(a_{ij}, \vartheta, \lambda) = \frac{1}{2}(\sum_{k=1}^{j} \lambda_{j,k}\vartheta_k a_{ij})^2 - \log(\vartheta_j a'_{ij})$ which is the negative logarithm of $g(i, j) = \vartheta_j a'_{ij} \exp(-\frac{1}{2}(\sum_{k=1}^{j} \lambda_{j,k}\vartheta_k a_{ij})^2)$. We assume that for all $i \in [n], j \in [J]$ and all choices of parameters it holds that $g(i, j) \leq c$ for some constant $c \in \mathbb{R}_{\geq 1}$. This is a natural Lipschitz-type restriction ensuring that the distribution function has a smooth transition from 0 to 1 preventing sudden jumps. Note, that this is equivalent to $-\ln(g(i, j)) \geq -\ln(c)$. We further add to the nJ loss contributions a total shift of $\log \mathcal{N} = nJ \ln(c) + n/J$. This corresponds to a normalization term $\mathcal{N}$ that ensures non-negativity and thus allows a relative approximation to be meaningful, but does not affect the optimization because it is independent of the parameters that we optimize. Additionally, we assume that there is an intercept, i.e., that for each (i, j) the first coordinate of both a_{ij} and a'_{ij} is 1 to make it consistent with the presence of intercepts in the transformation functions h defined above. In the following, we show that if we sample with probabilities that are larger than the ℓ_2 leverage scores plus a uniform term and add the convex hull of $\{a'_{ij} \mid i \in [n], j \in [J]\}$, then we get a coreset

for $f(A, \vartheta, \lambda)$ to be minimized as in Equation (3.1). Let $w \in \mathbb{R}^{n \times J}$ be a weight vector. We split the weighted version of f into three parts (w_{ij} are omitted in the unweighted case):

$$\begin{aligned}
\text{squared part:} \quad & f_1(A, \vartheta, \lambda, w) = \frac{1}{2} \sum_{(i,j) \in [n] \times [J]} w_{ij} \left(\sum_{k=1}^{j} \lambda_{j,k} \langle \vartheta_k, a_{ij} \rangle \right)^2 \\
\text{positive log part:} \quad & f_2(A, \vartheta, \lambda, w) = \sum_{(i,j) \in [n] \times [J]} w_{ij} \max\{\log(\langle \vartheta_j, a'_{ij} \rangle), 0\} \\
\text{negative log part:} \quad & f_3(A, \vartheta, \lambda, w) = \sum_{(i,j) \in [n] \times [J]} w_{ij} \max\{-\log(\langle \vartheta_j, a'_{ij} \rangle), 0\}.
\end{aligned}$$

Squared part. We let $u_i = \sup_{\|x\|_2 = 1} \frac{|M_i x|^2}{\|Mx\|_2^2}$ for the i-th row of a matrix M denote their ℓ_2 leverage score. We show that sampling proportionally to the ℓ_2 leverage scores preserves the squared part f_1: given rows $a_{ij} \in \mathbb{R}^d$ where $i \in [n]$ and $j \in [J]$, we are looking for an ε-coreset, which is given by a matrix comprising a subset of rows indexed by $S \subseteq [n] \times [J]$ and corresponding weights $w_{i,j}$ for every $(i, j) \in S$, such that for all $\vartheta_1, \ldots, \vartheta_J \in \mathbb{R}^d$ and $\lambda \in \mathbb{R}^{J \times J}$ it holds that

$$\begin{aligned}
&\left| \sum_{i=1}^{n} \sum_{j=1}^{J} \left(\sum_{k=1}^{j} \lambda_{j,k} \langle \vartheta_k, a_{ij} \rangle \right)^2 - \sum_{(i,j) \in S} w_{i,j} \left(\sum_{k=1}^{j} \lambda_{j,k} \langle \vartheta_k, a_{ij} \rangle \right)^2 \right| \\
&\qquad \leq \varepsilon \left| \sum_{i=1}^{n} \sum_{j=1}^{J} \left(\sum_{k=1}^{j} \lambda_{j,k} \langle \vartheta_k, a_{ij} \rangle \right)^2 \right|.
\end{aligned}$$

In the following, we arrange data points in a matrix such that sampling rows of this matrix yields a coreset for the function defined above. We set $B \in \mathbb{R}^{nJ \times dJ^2}$ to be the matrix whose rows equal $(b_{iJ+j})_k = a_{il}$ if $k = (j-1)J + l$ for some $l \in [J]$ and $(b_{iJ+j})_k = 0$ otherwise. Then B consists of n vertically stacked blocks. The i-th block, for $i \in [n]$, is defined by

$$B_i = \begin{bmatrix}
b_i & 0 & 0 & 0 & \cdots & 0 \\
0 & b_i & 0 & 0 & \cdots & 0 \\
0 & 0 & b_i & 0 & \cdots & 0 \\
\vdots & \vdots & \vdots & \ddots & \cdots & 0 \\
0 & 0 & 0 & 0 & \cdots & b_i
\end{bmatrix} \in \mathbb{R}^{J \times dJ^2},$$

where $b_i = (b_{i1}, b_{i2}, \ldots, b_{iJ})$. The idea is that for any possible parametrization, the squared part can be represented by a product of the new matrix B with the vector $\theta = (\vartheta^T, \lambda^T)^T$. An ε-subspace embedding for ℓ_2 is therefore sufficient to approximate the squared part, i.e., it yields that $\forall \theta \colon \|B'\theta\|_2^2 = (1 \pm \varepsilon)\|B\theta\|_2^2$, where B' consists of few weighted rows subsampled from B according to their ℓ_2 leverage scores (Woodruff, 2014).

Lemma 3.3.1. *There is a coreset S for f_1 of size $O(J^2 d/\varepsilon^2)$ which can be computed in time $O(\mathrm{nnz}(B) \log(nJ) + \mathrm{poly}(dJ))$ and with high probability by sampling and reweighting*

according to the ℓ_2 leverage scores of B, where $\operatorname{nnz}(B)$ *denotes the number of non-zero entries of B. We then have $|f_1(A,\vartheta,\lambda) - f_1(A(S),\vartheta,\lambda,w)| \leq \varepsilon f_1(A,\vartheta,\lambda)$, where $A(S)$ denotes the restriction to S.*

Logarithmic parts. We now turn our attention to the positive logarithmic part f_2. This is going to be handled using the sensitivity framework, which generalizes the previous leverage score sampling for the ℓ_2 norm to more general families of functions, we refer to Appendix A.2.6 for details. First of all, we bound the VC dimension of the logarithmic function. This is done in a standard way. Using strict monotonicity, the logarithmic function of the inner product can be inverted (respecting their domain and range), relating it to linear classifiers in d dimensions. The latter have a known VC dimension of $d+1$ using classic learning theory results (Kearns and Vazirani, 1994). The second part is bounding the total sensitivity, which is the sum of all sensitivity scores $s_i = \sup_{\vartheta,\lambda} \frac{f_2(a_{ij},\vartheta,\lambda)}{\sum f_2(a_{ij},\vartheta,\lambda)}$ of single data point contributions. To bound this value, we leverage that for all parameters λ and ϑ it holds that $\ln(\vartheta_j a'_{ij}) - \frac{1}{2}(w_{i,j}\sum_{k=1}^{j}\lambda_{j,k}\vartheta_k a_{ij})^2 \leq \ln(c)$, which allows us to relate the contribution of the logarithmic part to the squared part up to a constant $\gamma > 1$ such that $s_i \leq \gamma(u_i + 1/n)$. This allows to reuse the ℓ_2 leverage scores for this part as well, albeit with an additional uniform component, and with an increase of the sample size, which comes from incorporating the VC dimension and the Lipschitz bound c. We also note that the ε-error is relative to f_1 instead of f_2 directly.

Lemma 3.3.2. *Assume that S is a sample consisting of $O(J^2d^2\ln(cdJ)c^6/\varepsilon^2)$ points drawn with probability $p_i \geq \alpha(u_i + 1/n)$, where $\alpha \in O(J^2d\ln(cdJ)c^6/\varepsilon^2)$. Then with high probability it holds that $|f_2(A,\vartheta,\lambda) - f_2(A(S),\vartheta,\lambda,w)| \leq \varepsilon f_1(A,\vartheta,\lambda)$, where $A(S)$ denotes the restriction to S.*

Next, we handle the remaining negative logarithmic part given by f_3. We note that the asymptote at 0 precludes finite bounds on the sensitivity. We handle this similarly to Lie and Munteanu (2024) by restricting the optimization space to $D(\eta) = \{(\vartheta,\lambda) \mid \forall(i,j) \in [n]\times[J] : \langle\vartheta_j, a'_{ij}\rangle > \eta\}$ comprising only solutions for which the inner product is bounded away by $\eta \geq 0$ from zero. Setting $\eta = 0$ corresponds to the original domain.[6] By avoiding high sensitivity points in this way, f_3 can be bounded in terms of uniform sensitivities together with the VC dimension bound $d+1$ and approximated by invoking the sensitivity framework again.

Lemma 3.3.3. *Let (ϑ^*,λ^*) be an optimal solution. Then there exist $(\vartheta,\lambda) \in D(\eta)$ with $|f(A,\vartheta,\lambda) - f(A,\vartheta^*,\lambda^*)| \leq 2J\eta f_1(A,\vartheta^*,\lambda^*) + \eta n + J^2\eta^2$. Further, if S is a sample consisting of points drawn with probability $p_i \geq \alpha(1/n)$ where $\alpha \in O(d\ln(cdJ)/\eta^2)$ combined with all points that are in the convex hull of $\{a'_{ij} \mid i \in [n], j \in [J]\}$. Then with high probability, it holds for all $(\vartheta,\lambda) \in D(\eta)$ that $|f_3(A,\vartheta,\lambda) - f_3(A(S),\vartheta,\lambda,w)| \leq \eta f_1(A,\vartheta,\lambda) + \eta n$.*

[6] We also note that the final choice will be $\eta = \Theta(\varepsilon)$ and negative value correction into the positive domain is common practice and was implemented in Klein et al. (2022) before our theoretical investigations.

Main result. We get the following theorem by a union bound over Lemmas 3.3.1 to 3.3.3 and putting their error bounds together using the triangle inequality. The additive errors of Lemma 3.3.3 are further charged against the optimal cost using the normalization given by $\log \mathcal{N}$, see above.

Theorem 3.3.4. *Assume that S is a sample consisting of $O(J^2d^2\ln(cdJ)c^6/\varepsilon^2)$ points drawn with probability $p_i = \alpha(u_i + 1/n)$, where $\alpha \in O(J^2d\ln(cdJ)c^6/\varepsilon^2)$ combined with all extreme points $(i,j) \in [n] \times [J]$ that are on the convex hull of $\{a'_{ij} \mid i \in [n], j \in [J]\}$. Then with high probability it holds for any $(\vartheta, \lambda) \in D(\eta)$ that $|f(A,\vartheta,\lambda) - f(A(S),\vartheta,\lambda,w)| \leq \varepsilon f(A)$.*

The formal proof can be found in Appendix A.2. We remark that the convex hull can comprise $\Omega(nJ)$ points, however, different η-kernel coresets of size $\Theta(1/\eta^{(d-1)/2})$ exist for the problem (Agarwal et al., 2004; Chan, 2004), surveyed in Agarwal et al. (2005), in the field of computational geometry. These η-kernels also match the requirements of the shifted domain $D(\eta)$. We discuss our particular choice (Blum et al., 2019) in the experimental section.

Lower bounds. We also give two lower bounds under different natural assumptions on the λ coefficients that define the dependence structure of the Gaussian copula. These indicate that there are only small remaining gaps to our upper bounds. The details of their construction are again given and proven in Appendix A.2.

Lemma 3.3.5. *There exists a dataset $\{a_{ij}\}_{i\in[n]j\in[J]}$ such that any coreset for MCTMs with $\varepsilon < 1$ has size at least $\Omega(dJ^2)$ even if $\lambda_{ij} = 0$ for all $i,j \in [n]$ with $j > i$ and $|\lambda_{ij}| \leq 1$ for all $i,j \in [J]$.*

Lemma 3.3.6. *There exists a dataset $\{a_{ij}\}_{i\in[n]j\in[J]}$ such that any coreset for MCTMs with $\varepsilon < 1$ has size at least $\Omega(dJ)$ even if $\lambda_{ii} = 1$ for all $i \in [J]$ and $\lambda_{ij} = 0$ for $i < j$.*

3.3.5 Empirical Evaluation

In this section, we systematically investigate *coreset* compression techniques for MCTMs with large-scale data for a variety of different 2-dimensional data generation processes as well as two large-scale and high-dimensional real-world applications. Our objectives are to

(i) validate the effectiveness of the proposed sampling and convex hull algorithm for MCTMs at fitting different distributions, different correlation structures, and non-linear or heavy-tailed scenarios; and

(ii) quantify the performance of comparing uniform sampling, pure ℓ_2 sensitivity sampling, and sensitivity sampling combined with the convex hull approximation on various metrics.

Our sensitivity sampling algorithm is detailed in Algorithm 2. As convex hull approximation, we chose to implement the algorithm of Blum et al. (2019) given in Algorithm 7 in Appendix Appendix C. For mild data that admits an η-kernel below the $\Omega(1/\eta^{(d-1)/2})$ lower bound, it yields an η-kernel of size $O(k^*/\eta^2)$, where k^* is

Algorithm 2 Hybrid coreset construction for MCTMs

Input: Full dataset $\mathcal{D} = \{y_i\}_{i=1}^n \subset \mathbb{R}^J$, target coreset size k
Output: Weighted coreset $\mathcal{C} = \{(y_j, w_j)\}_{j=1}^k$
Compute transformed statistics: Apply Bernstein polynomial transformation a of degree d to each y_i to obtain $B \in \mathbb{R}^{nJ \times dJ^2}$ as described in Section 3.3.4 resp. Appendix A.2.2
Compute ℓ_2 leverage scores of B: Use fast leverage score computation as in Theorem 2.13 of Woodruff (2014) to get a constant factor approximation for $u_i = \sup_{x \neq 0} \frac{|B_i x|^2}{\|Bx\|_2^2}$ for each $i \in [nJ]$
Compute sensitivity proxy: For each i, set $s_i \leftarrow u_i + 1/n$
Normalize to probabilities: $p_i \leftarrow \frac{s_i}{\sum_{j=1}^n s_j}$
Sampling phase:
    Let $k_1 = \lfloor \alpha k \rfloor$ (e.g. $\alpha = 0.8$)
    Sample k_1 points independently with probability $\{p_i\}_{i=1}^n$
    Assign weights $w_i \leftarrow \frac{1}{k_1 \cdot p_i}$ for each sampled point
Convex hull augmentation:
    Let $k_2 = k - k_1$
    Compute ε/J-kernel convex hull approximation as in Blum et al. (2019) over the derivatives $\{a_j'(y_{ij})\}_{i \in [n], j \in [J]}$ and select k_2 extremal points
    Assign weight $w_j \leftarrow 1$ to each convex hull point
Form final coreset: Combine sampled points and hull points into $\mathcal{C}$ with associated weights w
Coreset fitting: Fit MCTM using weighted log-likelihood:

$$\hat{\theta}_{\text{coreset}} = \text{argmax}_\theta \sum_{(Y_j, w_j) \in \mathcal{C}} w_j \cdot \log f_\theta(Y_j).$$

the smallest possible size. All experiments were carried out on a 2021 MacBook Pro equipped with an Apple M1 Pro chip and 16 GB of RAM.[7]

Simulation Study Based on 2-Dimensional Data

We conduct a set of 2-dimensional data simulation experiments encompassing different dependence structures to validate the advantages of the proposed coreset approach over simple uniform sampling as a baseline.

Data Generation Processes

To thoroughly evaluate our proposed method across diverse dependency structures, we implemented 14 different data generation processes. Each process was designed to test specific aspects of dependency modeling. For all processes, we generated datasets with $n = 10\,000$ samples and evaluated coreset performance at sizes 30 and 100. Below, we describe each process mathematically.

1. Bivariate Normal Distribution

[7] The code to reproduce our experiments is available at https://github.com/zeyudsai/mctmcoreset/.

The standard bivariate normal distribution with correlation parameter ρ:

$$(Y_1, Y_2) \sim \mathcal{N}\left(\begin{bmatrix}0\\0\end{bmatrix}, \begin{bmatrix}1 & \rho\\ \rho & 1\end{bmatrix}\right),$$

where $\rho = 0.7$ represents the correlation between variables. This baseline case tests performance on linear dependency structures.

2. Non-linear Correlation

A non-linear correlation structure where the correlation coefficient varies with the predictor:

$$\begin{aligned} &Y_1 = X^2 + \epsilon_1, \quad \epsilon_1 \sim \mathcal{N}(0, 0.5^2),\\ &Y_2 \sim \mathcal{N}(0,1), \text{ with correlation } \rho(X) = \sin(X) \text{ to } Y_1, \end{aligned}$$

where $X \in [-3, 3]$. This tests the ability to capture location-dependent correlation.

3. Bivariate Normal Mixture

A mixture of two bivariate normal distributions with different means and correlation structures:

$$\begin{aligned} (Y_1, Y_2) \sim\ & 0.5 \cdot \mathcal{N}\left(\begin{bmatrix}0\\0\end{bmatrix}, \begin{bmatrix}1 & 0.8\\ 0.8 & 1\end{bmatrix}\right)\\ & + 0.5 \cdot \mathcal{N}\left(\begin{bmatrix}3\\-2\end{bmatrix}, \begin{bmatrix}1.5 & -0.5\\ -0.5 & 1.5\end{bmatrix}\right). \end{aligned}$$

This tests performance on multimodal data with different local dependency structures.

4. Geometric Mixed Distribution

A distribution combining two distinct geometric patterns:

$$(Y_1, Y_2) \sim 0.5 \cdot \text{Circular} + 0.5 \cdot \text{Cross},$$

where the Circular component generates points along a circle with radius $r \sim \mathcal{N}(2, 0.2^2)$ and uniform angle $\theta \sim \text{Uniform}(0, 2\pi)$, while the Cross component generates points along two perpendicular lines with controlled variance. This distribution tests the ability to capture multiple geometric structures simultaneously, with continuous circular features intersecting with linear patterns. The sharp differences in geometric structure and local data density create a particularly challenging scenario for coreset selection.

5. Skewed t-Distribution

A heavy-tailed distribution with skewness:

$$(Y_1, Y_2) \sim \text{Skew-}t_\nu\left(\xi, \Omega, \alpha\right),$$

where $\xi = [0,0]^T$ is the location parameter, $\Omega = \begin{bmatrix} 1 & 0.5 \\ 0.5 & 1 \end{bmatrix}$ is the scale matrix, $\alpha = [5,-3]^T$ is the skewness parameter, and $\nu = 4$ specifies the degrees of freedom. This tests performance on asymmetric, heavy-tailed distributions.

6. Heteroscedastic Distribution

A distribution where the variance depends on the location:

$$Y_1 \sim \mathcal{N}(X^2, \sigma_1^2(X)), \text{ where } \sigma_1(X) = e^{0.5X}.$$
$$Y_2 \sim \mathcal{N}(\sin(X), \sigma_2^2(X)), \text{ where } \sigma_2(X) = \sqrt{|X|}.$$

with $X \in [-3,3]$. This tests the ability to capture variance heterogeneity.

7. Copula Complex Distribution

A Clayton copula-based dependency structure with gamma and log-normal marginals:

$$\begin{aligned}(U_1, U_2) &\sim C_{\text{Clayton}}(\theta = 2),\\ Y_1 &= F^{-1}_{\text{Gamma}(2,1)}(U_1),\\ Y_2 &= F^{-1}_{\text{LogNormal}(0,1)}(U_2).\end{aligned}$$

This tests performance on complex dependency structures with non-normal marginals.

8. Spiral Dependency

A spiral pattern with added noise:

$$\begin{aligned}t &\in [0, 3\pi],\\ r &= 0.5t,\\ Y_1 &= r\cos(t) + \epsilon_1, \quad \epsilon_1 \sim \mathcal{N}(0, 0.5^2),\\ Y_2 &= r\sin(t) + \epsilon_2, \quad \epsilon_2 \sim \mathcal{N}(0, 0.5^2).\end{aligned}$$

This tests performance on complex geometric dependencies.

9. Circular Dependency

A circular pattern with radius variation:

$$\begin{aligned}\theta &\sim \text{Uniform}(0, 2\pi),\\ r &\sim \mathcal{N}(5, 1^2),\\ Y_1 &= r\cos(\theta),\\ Y_2 &= r\sin(\theta).\end{aligned}$$

This tests the ability to capture circular dependencies where linear correlation measures fail.

10. t-Copula Dependency

A t-copula with t and exponential marginals:

$$\begin{aligned}(U_1, U_2) &\sim C_t(\rho = 0.7, \nu = 3),\\ Y_1 &= F_{t_5}^{-1}(U_1),\\ Y_2 &= F_{\text{Exp}(1)}^{-1}(U_2).\end{aligned}$$

This tests performance on tail dependencies with asymmetric marginals.

11. Piecewise Dependency

A piecewise function with different correlation regimes:

$$\begin{aligned}Y_1 &\sim \mathcal{N}(0, 2^2)\\ Y_2 &= \begin{cases} 1.5Y_1 + \epsilon_1, & \text{if } Y_1 < -1,\\ -0.5Y_1 + \epsilon_2, & \text{if } -1 \le Y_1 < 1,\\ -2Y_1 + \epsilon_3, & \text{if } Y_1 \ge 1,\end{cases}\end{aligned}$$

where $\epsilon_1, \epsilon_3 \sim \mathcal{N}(0, 0.5^2)$ and $\epsilon_2 \sim \mathcal{N}(0, 0.8^2)$. This tests ability to capture regime-dependent correlations.

12. Hourglass Dependency

A heteroscedastic pattern with variance increasing quadratically from center:

$$\begin{aligned}&Y_1 \sim \mathcal{N}(0, 2^2),\\ &Y_2 \sim \mathcal{N}(0, \sigma^2(Y_1)), \text{ where } \sigma^2(Y_1) = 0.2 + 0.3Y_1^2.\end{aligned}$$

This tests ability to capture complex variance structures.

13. Bimodal Clusters

Two distinct clusters with opposing correlation structures:

$$\begin{aligned}(Y_1, Y_2) \sim\ & 0.5 \cdot \mathcal{N}\left(\begin{bmatrix}-2\\ 2\end{bmatrix}, \begin{bmatrix}1 & 0.8\\ 0.8 & 1\end{bmatrix}\right)\\ & + 0.5 \cdot \mathcal{N}\left(\begin{bmatrix}2\\ 2\end{bmatrix}, \begin{bmatrix}1 & -0.7\\ -0.7 & 1\end{bmatrix}\right),\end{aligned}$$

This tests the ability to capture cluster-specific dependency structures.

14. Sinusoidal Dependency

A sinusoidal relationship with noise:

$$\begin{aligned}&Y_1 \in [-3, 3],\\ &Y_2 = 2\sin(\pi Y_1) + \epsilon, \quad \epsilon \sim \mathcal{N}(0, 0.5^2).\end{aligned}$$

This tests performance on periodic dependencies.

These 14 data generation processes provide a comprehensive evaluation framework for testing our coreset method across a wide range of dependency structures,

from simple linear correlations to complex geometric patterns and regime-dependent relationships.

Simulation Data Visualization

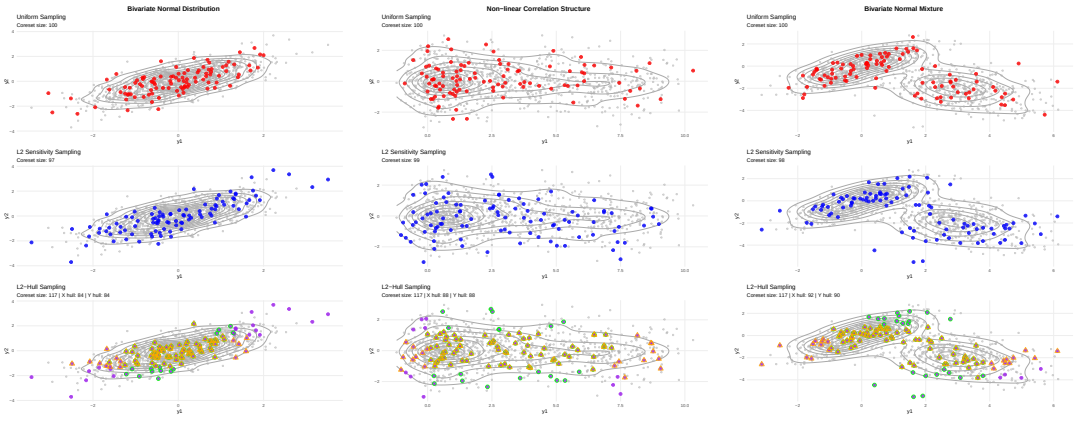
Coreset Construction Methods We compare three sampling strategies:

1. **Uniform Subsampling**: k points are taken with equal probability from all n samples without replacement, and the weights are set to $\frac{n}{k}$.

2. ℓ_2 **-Only Sensitivity Sampling**: leverage scores (or upper bounds on sensitivity) are calculated for $a(x_i)$ and $a(y_i)$, respectively, and subsamples are combined and reweighted after sampling them with the probability $p_i \propto \ell_2$ leverage score; and the weight is then determined using *merge probability* to compute the final weights and do the normalization.

3. ℓ_2**-Hull Hybrid**: on the basis of the above ℓ_2 sensitivity sampling, we also perform *convex hull* (or ε-kernel) approximation sampling on its derivative matrix $a'(\cdot)$ by adding an extra batch of points located at the extreme geometric boundaries, thus avoiding the logarithmic term $\log(a'(x_i)^T\vartheta)$ in the likelihood function on which the values are unstable or near zero. The *sensitivity subsample* and *convex hull subsample* are eventually summarized together in a joint index.

We visualize each data generation process with coresets created using three different sampling methods: Uniform Sampling, ℓ_2 Sensitivity Sampling, and our proposed ℓ_2-Hull Sampling method. Each visualization shows how well the different coreset methods preserve the underlying data distribution structure.



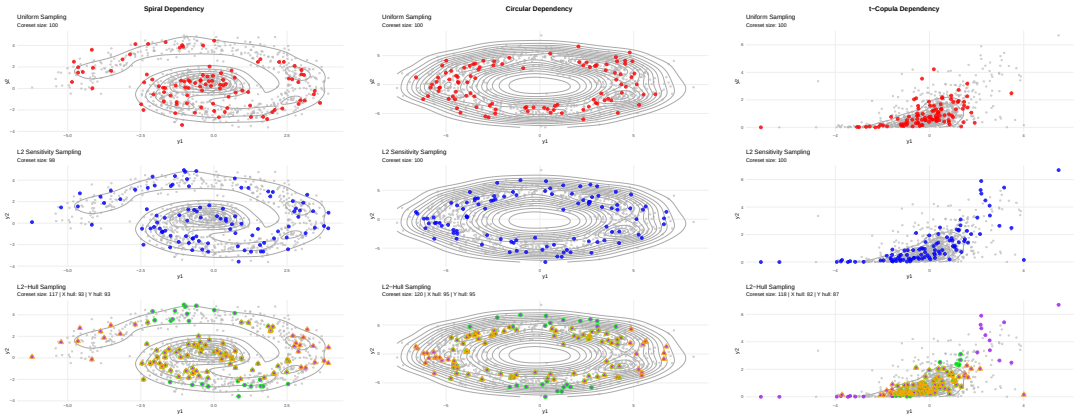
(A) Bivariate Normal (B) Non-linear Correlation (C) Bivariate Normal Mixture

FIGURE 3.5: Coreset visualization for basic probability distributions. Each row shows a different sampling method: Uniform (top), ℓ_2 Sensitivity (middle), and ℓ_2-Hull (bottom).

To visually demonstrate the advantages of our proposed method, we present coreset visualizations for representative data generation processes. In each figure, we display coresets of approximately 100 points generated from original samples

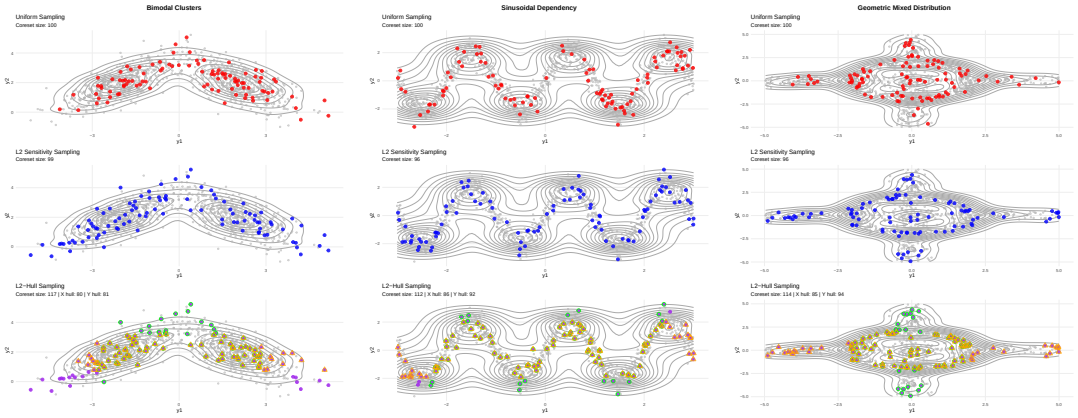
(A) Skew-t Distribution (B) Heteroscedastic Distribution (C) Copula Complex Distribution

FIGURE 3.6: Coreset visualization for complex probability distributions. Each column shows a different sampling method: Uniform (top), ℓ_2 Sensitivity (middle), and ℓ_2-Hull (bottom).



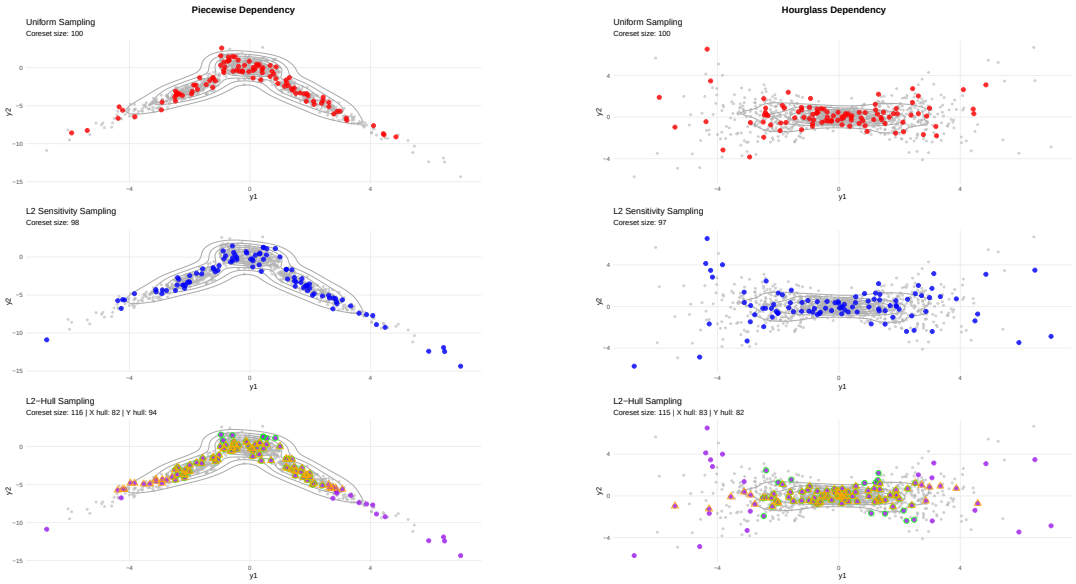
(A) Spiral Dependency (B) Circular Dependency (C) t-Copula Dependency

FIGURE 3.7: Coreset visualization for geometric dependency structures. Each column shows a different sampling method: Uniform (top), ℓ_2 Sensitivity (middle), and ℓ_2-Hull (bottom).



(A) Bimodal Clusters (B) Sinusoidal Dependency (C) Mixture Distribution

FIGURE 3.8: Coreset visualization for additional dependency structures. Each column shows a different sampling method: Uniform (top), ℓ_2 Sensitivity (middle), and ℓ_2-Hull (bottom).



(A) Piecewise Dependency (B) Hourglass Dependency

FIGURE 3.9: Coreset visualization for functional dependency structures. Each column shows a different sampling method: Uniform (top), ℓ_2 Sensitivity (middle), and ℓ_2-Hull (bottom).

of 1 000 points using three different sampling methods: Uniform Sampling (top), ℓ_2 Sensitivity Sampling (middle), and our proposed ℓ_2-Hull Sampling (bottom).

As shown in Figures 3.5–3.8, our ℓ_2-Hull method effectively captures the entire shape of the underlying data distribution, particularly in complex scenarios such as the Hourglass Dependency, Piecewise Dependency, Spiral Dependency, Copula Complex Distribution, and Bimodal Clusters. The visualization highlights a key limitation of uniform sampling: it often fails to include points that are distant from the central mass of the distribution but are nevertheless critical for accurately representing the overall data structure.

For instance, in the Bimodal Clusters visualization (Figure 3.8a), our ℓ_2-Hull method ensures comprehensive coverage of both clusters including their boundaries, while uniform sampling misses several critical points that define the extent of the distribution. Similarly, in the Piecewise Dependency case (Figure 3.9a), the non-linear relationship is better preserved by our method, particularly at the extremes of the distribution.

It is worth noting that in many scenarios, the performances of ℓ_2 Sensitivity Sampling and our ℓ_2-Hull method appear similar, especially as the coreset size increases. This is expected, as the primary purpose of adding convex hull points is to safeguard against extreme cases that may arise during optimization. As the sample size grows, the probability of encountering such extreme cases diminishes. Nevertheless, the ℓ_2-Hull method provides a theoretical guarantee that important boundary points are always included, regardless of the particular dataset instance.

The enhanced coverage provided by our method translates directly to the improved empirical performance metrics observed in Table 3.1 and Table 3.2, particularly for complex dependency structures where capturing the overall shape of the distribution is crucial for accurate statistical inference.

Simulation Experiments Results

In all the above scenarios, we will evaluate under the same sampling/modeling process and can adjust the sample size n as well as the correlation parameters (e.g., ρ) as needed.

Main Workflow For each data generation method, we follow the following flow of experiments:

1. **Data Generation:** Call the corresponding function (e.g. `generate_mixture`) to generate the samples, which can be randomly initialized multiple times.
2. **Full Data Baseline:** Perform MCTM fitting with full data, record its log-likelihood ℓ_{full} with the estimated coefficients $\hat{\theta}_{full}$, as a baseline.
3. **Coreset Sampling & Fitting:** Use *Uniform Sampling*, ℓ_2*-Only*, and ℓ_2*-Hull*, respectively, for multiple subset sizes from the set $k \in \{\text{min\_size}, \ldots, \text{max\_size}\}$ to obtain a subsample and compute the weights; subsequently, the MCTM is fitted using only this subsample to obtain the log-likelihood $\ell_{coreset}$ and estimated coefficients $\hat{\theta}_{coreset}$.
4. **Evaluation Metrics:**
   - **Likelihood Ratio:** compare $\ell_{coreset}/\ell_{full}$, if it is close to 1;
   - **Parameter Error:** calculate the value of $\|\hat{\vartheta}_{coreset} - \hat{\vartheta}_{full}\|_2^2$ and convergence trend under different k;
   - **Lambda Error:** Measure the absolute difference in the dependency parameter $|\lambda_{coreset} - \lambda_{full}|$ to evaluate how well the coreset preserves the dependency structure;
   - **Time Cost:** Record the sampling time consumption and optimization time consumption respectively for evaluating the efficiency of the algorithm.

Summary With such multi-scenario simulations, we can systematically examine the proposed MCTM coreset scheme under different distributions. Detailed comparative graphs and numerical analyses showing the combined advantages of the new method in terms of likelihood approximation accuracy, parameter estimation bias, and time efficiency are presented in the experimental results below.

Figure 3.10 visualizes the performance specifically for three distributions: bivariate normal mixture, non-linear correlation, and bimodal cluster distribution. In

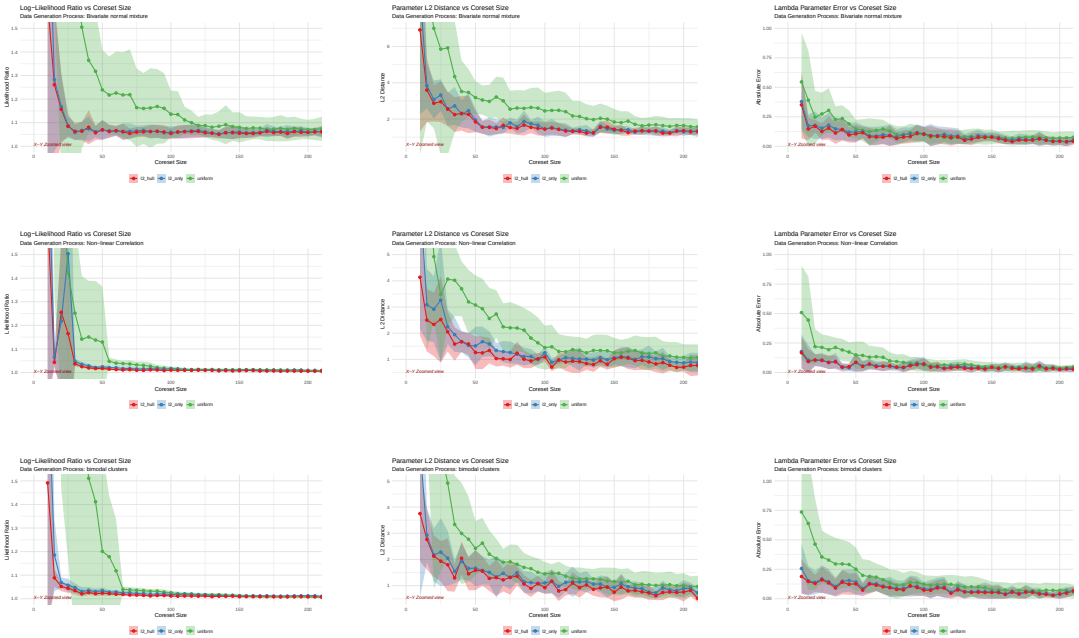


FIGURE 3.10: Convergence of the likelihood ratio, parameter error, and λ error as coreset size increases. First row: Bivariate mixture data of two Gaussian distributions with different means and variances. Second row: Non-linear correlation. Third row: Bimodal clusters with two distinct clusters with opposing correlation structure.

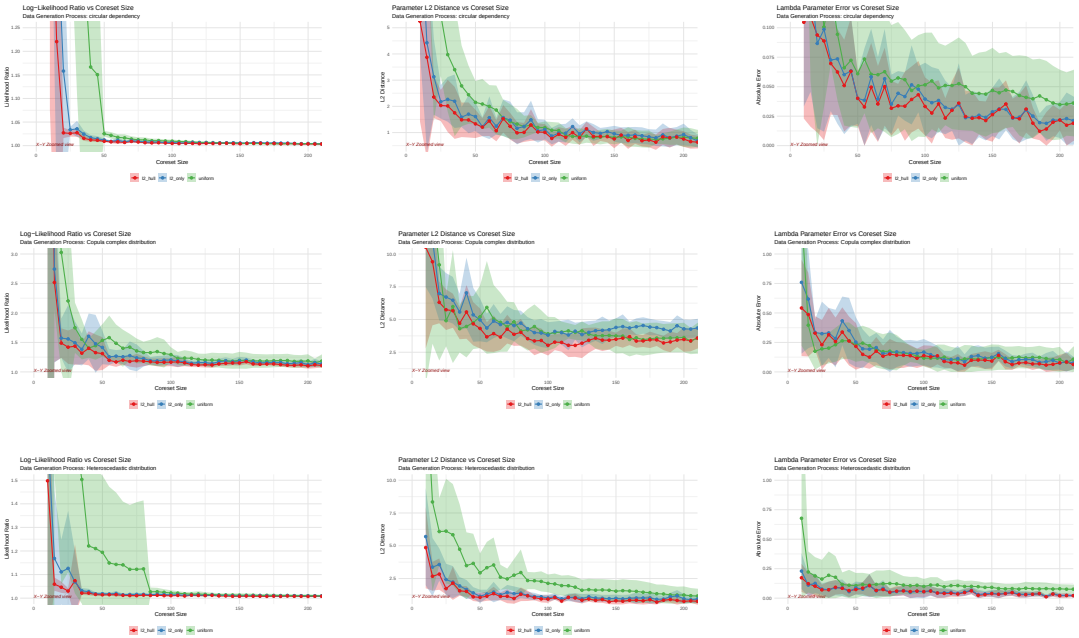


FIGURE 3.11: Convergence of the likelihood ratio, parameter error, and λ error as coreset size increases. First row: circular-dependency. Data points are distributed along circular trajectories, demonstrating a circular dependency structure between variables. Second row: copula complex. A copula construct with tail dependence and non-linear correlation is used. Third row: heteroscedastic distribution. The conditional variance of the data varies with the level of the independent variable, reflecting heteroscedasticity.

TABLE 3.1: Performance comparison of different coreset methods for various data generation processes (coreset Size = 30)

Data Gen. Process	Method	Param. ℓ_2 dist.	λ error	Likelihood ratio	Rel. Impr.(%)	Total time(s)
Bivariate normal	ℓ_2-hull	2.56 ± 0.76	0.44 ± 0.16	1.54 ± 0.29	12.8	0.21 ± 0.03
	ℓ_2-only	$\mathbf{2.54 \pm 0.62}$	0.51 ± 0.13	1.65 ± 0.33	1.6	0.20 ± 0.02
	uniform	4.91 ± 4.74	$\mathbf{0.29 \pm 0.26}$	1.94 ± 1.13	baseline	$\mathbf{0.13 \pm 0.03}$
Non-linear correlation	ℓ_2-hull	$\mathbf{1.76 \pm 0.88}$	$\mathbf{0.09 \pm 0.09}$	$\mathbf{1.03 \pm 0.01}$	**49.8**	0.19 ± 0.02
	ℓ_2-only	2.18 ± 0.82	0.10 ± 0.10	1.05 ± 0.02	38.8	0.19 ± 0.01
	uniform	3.08 ± 0.91	0.11 ± 0.07	1.21 ± 0.46	baseline	$\mathbf{0.12 \pm 0.01}$
Bivariate normal mixture	ℓ_2-hull	$\mathbf{2.60 \pm 1.00}$	$\mathbf{0.14 \pm 0.07}$	$\mathbf{1.07 \pm 0.03}$	**49.6**	0.20 ± 0.03
	ℓ_2-only	2.76 ± 0.94	0.16 ± 0.10	1.08 ± 0.05	43.7	0.21 ± 0.04
	uniform	4.14 ± 1.80	0.20 ± 0.15	1.42 ± 0.34	baseline	$\mathbf{0.15 \pm 0.04}$
Geometric Mixed Distribution	ℓ_2-hull	$\mathbf{2.51 \pm 2.61}$	$\mathbf{0.06 \pm 0.04}$	$\mathbf{1.33 \pm 0.53}$	**41.4**	0.20 ± 0.04
	ℓ_2-only	4.09 ± 4.35	0.07 ± 0.03	1.58 ± 0.58	9.0	0.20 ± 0.02
	uniform	4.11 ± 2.49	0.14 ± 0.09	1.47 ± 0.53	baseline	0.13 ± 0.03
Skew-t distribution	ℓ_2-hull	$\mathbf{3.55 \pm 1.46}$	0.61 ± 0.27	$\mathbf{1.90 \pm 0.70}$	0	0.23 ± 0.03
	ℓ_2-only	3.72 ± 1.36	0.70 ± 0.26	2.14 ± 0.91	0	0.23 ± 0.09
	uniform	4.14 ± 2.04	$\mathbf{0.36 \pm 0.31}$	2.17 ± 0.96	baseline	$\mathbf{0.15 \pm 0.07}$
Heteroscedastic distribution	ℓ_2-hull	$\mathbf{2.07 \pm 0.69}$	$\mathbf{0.08 \pm 0.05}$	$\mathbf{1.07 \pm 0.12}$	**64.8**	0.20 ± 0.07
	ℓ_2-only	2.65 ± 0.88	$\mathbf{0.08 \pm 0.05}$	1.09 ± 0.13	58.4	0.20 ± 0.02
	uniform	4.47 ± 3.38	0.16 ± 0.14	1.63 ± 1.19	baseline	$\mathbf{0.13 \pm 0.02}$
Copula complex distribution	ℓ_2-hull	$\mathbf{4.56 \pm 1.57}$	$\mathbf{0.18 \pm 0.13}$	$\mathbf{1.23 \pm 0.20}$	**58.0**	0.26 ± 0.04
	ℓ_2-only	5.30 ± 1.48	0.20 ± 0.15	1.36 ± 0.29	51.1	0.28 ± 0.08
	uniform	11.05 ± 7.80	0.27 ± 0.22	2.35 ± 0.56	baseline	$\mathbf{0.18 \pm 0.06}$
Spiral dependency	ℓ_2-hull	$\mathbf{1.86 \pm 0.55}$	$\mathbf{0.04 \pm 0.03}$	$\mathbf{1.04 \pm 0.01}$	**79.5**	0.21 ± 0.03
	ℓ_2-only	2.38 ± 0.80	0.06 ± 0.05	1.07 ± 0.02	71.9	0.22 ± 0.07
	uniform	6.16 ± 3.50	0.15 ± 0.09	2.05 ± 0.60	baseline	$\mathbf{0.21 \pm 0.22}$
Circular dependency	ℓ_2-hull	$\mathbf{1.51 \pm 0.39}$	$\mathbf{0.06 \pm 0.04}$	$\mathbf{1.02 \pm 0.01}$	**59.3**	0.20 ± 0.02
	ℓ_2-only	1.95 ± 0.62	$\mathbf{0.06 \pm 0.04}$	1.12 ± 0.29	46.3	0.21 ± 0.03
	uniform	4.11 ± 3.24	0.08 ± 0.08	1.33 ± 0.73	baseline	$\mathbf{0.14 \pm 0.03}$
t Copula	ℓ_2-hull	$\mathbf{5.77 \pm 1.61}$	0.51 ± 0.30	$\mathbf{1.87 \pm 0.62}$	0	0.26 ± 0.03
	ℓ_2-only	7.18 ± 1.63	0.60 ± 0.29	1.98 ± 0.63	0	0.24 ± 0.03
	uniform	6.58 ± 3.35	$\mathbf{0.23 \pm 0.22}$	2.56 ± 0.92	baseline	$\mathbf{0.23 \pm 0.25}$
Piecewise dependency	ℓ_2-hull	$\mathbf{2.20 \pm 0.99}$	$\mathbf{0.30 \pm 0.13}$	$\mathbf{1.11 \pm 0.19}$	**58.5**	0.21 ± 0.04
	ℓ_2-only	2.38 ± 0.83	0.35 ± 0.15	$\mathbf{1.11 \pm 0.15}$	53.4	0.19 ± 0.02
	uniform	5.48 ± 4.11	0.46 ± 0.41	1.53 ± 0.74	baseline	$\mathbf{0.13 \pm 0.03}$
Hourglass dependency	ℓ_2-hull	1.99 ± 1.19	$\mathbf{0.14 \pm 0.07}$	$\mathbf{1.04 \pm 0.02}$	**55.8**	0.19 ± 0.03
	ℓ_2-only	$\mathbf{1.85 \pm 1.04}$	0.15 ± 0.08	1.06 ± 0.03	51.7	0.18 ± 0.01
	uniform	5.00 ± 3.21	0.16 ± 0.15	1.65 ± 0.69	baseline	$\mathbf{0.12 \pm 0.01}$
Bimodal clusters	ℓ_2-hull	$\mathbf{2.14 \pm 0.47}$	$\mathbf{0.15 \pm 0.09}$	$\mathbf{1.04 \pm 0.01}$	**58.5**	0.19 ± 0.02
	ℓ_2-only	2.61 ± 0.66	0.17 ± 0.09	1.06 ± 0.02	51.7	0.18 ± 0.02
	uniform	4.43 ± 3.05	0.22 ± 0.16	1.61 ± 0.74	baseline	$\mathbf{0.14 \pm 0.04}$
Sinusoidal dependency	ℓ_2-hull	$\mathbf{1.42 \pm 0.48}$	$\mathbf{0.05 \pm 0.05}$	$\mathbf{1.02 \pm 0.01}$	**38.6**	0.18 ± 0.03
	ℓ_2-only	1.66 ± 0.50	0.09 ± 0.08	1.03 ± 0.01	16.8	0.19 ± 0.02
	uniform	1.91 ± 0.74	0.10 ± 0.06	1.04 ± 0.01	baseline	$\mathbf{0.12 \pm 0.01}$

Note: The results in the table are the mean ± 1 standard deviation of independent simulations. The Relative Improvement is calculated as the average percentage improvement across all metrics, where improvement for parameter ℓ_2 distance and λ error is $(\text{baseline} - \text{method})/\text{baseline} \times 100\%$, and for likelihood ratio it is $(|\text{baseline} - 1| - |\text{method} - 1|)/|\text{baseline} - 1| \times 100\%$. The parameters ℓ_2 distance and λ error are better when smaller, and the log-likelihood ratio is better when closer to 1. Negative relative improvements are shown as 0. The best performance for each metric in each data generation process is highlighted in **bold**.

TABLE 3.2: Performance comparison of different coreset methods for various data generation processes (coreset Size = 100)

Data Gen. Process	Method	Param. ℓ_2 dist.	λ error	Likelihood ratio	Rel. Impr.(%)	Total time(s)
Bivariate normal	ℓ_2-hull	$\mathbf{1.80 \pm 0.42}$	0.30 ± 0.07	1.32 ± 0.11	0	0.23 ± 0.02
	ℓ_2-only	2.01 ± 0.36	0.39 ± 0.09	1.48 ± 0.17	0	0.24 ± 0.04
	uniform	1.98 ± 0.66	$\mathbf{0.11 \pm 0.06}$	$\mathbf{1.19 \pm 0.08}$	baseline	$\mathbf{0.13 \pm 0.02}$
Non-linear correlation	ℓ_2-hull	$\mathbf{1.05 \pm 0.42}$	$\mathbf{0.03 \pm 0.03}$	$\mathbf{1.01 \pm 0.00}$	**52.6**	0.20 ± 0.02
	ℓ_2-only	1.20 ± 0.53	$\mathbf{0.03 \pm 0.03}$	$\mathbf{1.01 \pm 0.01}$	38.8	0.22 ± 0.02
	uniform	1.79 ± 0.79	0.08 ± 0.05	1.02 ± 0.01	baseline	$\mathbf{0.11 \pm 0.02}$
Bivariate normal mixture	ℓ_2-hull	1.54 ± 0.42	$\mathbf{0.04 \pm 0.04}$	$\mathbf{1.06 \pm 0.01}$	**34.6**	0.30 ± 0.22
	ℓ_2-only	$\mathbf{1.53 \pm 0.45}$	0.07 ± 0.04	$\mathbf{1.06 \pm 0.02}$	26.4	0.21 ± 0.02
	uniform	1.99 ± 0.58	0.09 ± 0.09	1.09 ± 0.04	baseline	$\mathbf{0.13 \pm 0.02}$
Geometric Mixed Distribution	ℓ_2-hull	$\mathbf{1.19 \pm 0.51}$	$\mathbf{0.02 \pm 0.02}$	$\mathbf{1.01 \pm 0.00}$	**49.4**	0.20 ± 0.02
	ℓ_2-only	1.27 ± 0.60	0.03 ± 0.02	1.01 ± 0.00	42.1	0.21 ± 0.03
	uniform	1.75 ± 0.66	0.06 ± 0.04	1.02 ± 0.01	baseline	0.13 ± 0.03
Skew-t distribution	ℓ_2-hull	$\mathbf{2.04 \pm 0.42}$	0.26 ± 0.14	$\mathbf{1.24 \pm 0.10}$	8.1	0.22 ± 0.01
	ℓ_2-only	$\mathbf{1.99 \pm 0.33}$	0.33 ± 0.16	1.28 ± 0.12	0	0.22 ± 0.03
	uniform	2.67 ± 1.11	$\mathbf{0.20 \pm 0.15}$	1.34 ± 0.25	baseline	$\mathbf{0.12 \pm 0.02}$
Heteroscedastic distribution	ℓ_2-hull	$\mathbf{1.06 \pm 0.27}$	$\mathbf{0.04 \pm 0.03}$	$\mathbf{1.01 \pm 0.00}$	**27.6**	0.24 ± 0.08
	ℓ_2-only	1.34 ± 0.40	$\mathbf{0.04 \pm 0.02}$	1.02 ± 0.00	11.3	0.21 ± 0.02
	uniform	1.46 ± 0.84	0.09 ± 0.05	1.01 ± 0.01	baseline	$\mathbf{0.13 \pm 0.02}$
Copula complex distribution	ℓ_2-hull	$\mathbf{3.53 \pm 0.70}$	$\mathbf{0.12 \pm 0.09}$	$\mathbf{1.15 \pm 0.10}$	**32.0**	0.26 ± 0.04
	ℓ_2-only	4.05 ± 0.78	0.13 ± 0.08	1.16 ± 0.08	24.1	0.26 ± 0.03
	uniform	3.75 ± 1.42	0.18 ± 0.15	1.34 ± 0.27	baseline	$\mathbf{0.18 \pm 0.04}$
Spiral dependency	ℓ_2-hull	$\mathbf{1.37 \pm 0.43}$	$\mathbf{0.02 \pm 0.02}$	$\mathbf{1.01 \pm 0.01}$	**34.4**	0.25 ± 0.08
	ℓ_2-only	1.52 ± 0.37	0.03 ± 0.02	1.03 ± 0.01	0	0.27 ± 0.19
	uniform	1.88 ± 0.57	0.04 ± 0.03	1.02 ± 0.01	baseline	$\mathbf{0.13 \pm 0.01}$
Circular dependency	ℓ_2-hull	$\mathbf{0.94 \pm 0.36}$	$\mathbf{0.02 \pm 0.01}$	$\mathbf{1.01 \pm 0.00}$	**54.2**	0.21 ± 0.01
	ℓ_2-only	1.05 ± 0.20	0.03 ± 0.02	$\mathbf{1.01 \pm 0.00}$	43.9	0.21 ± 0.01
	uniform	1.83 ± 0.66	0.05 ± 0.04	1.01 ± 0.01	baseline	$\mathbf{0.12 \pm 0.01}$
t Copula	ℓ_2-hull	3.86 ± 1.03	0.28 ± 0.25	1.39 ± 0.38	0	0.27 ± 0.04
	ℓ_2-only	4.71 ± 1.26	0.32 ± 0.27	1.49 ± 0.50	0	0.28 ± 0.15
	uniform	$\mathbf{2.43 \pm 1.04}$	$\mathbf{0.13 \pm 0.14}$	$\mathbf{1.16 \pm 0.16}$	baseline	$\mathbf{0.19 \pm 0.11}$
Piecewise dependency	ℓ_2-hull	$\mathbf{1.06 \pm 0.36}$	$\mathbf{0.09 \pm 0.08}$	$\mathbf{1.01 \pm 0.01}$	**49.8**	0.22 ± 0.03
	ℓ_2-only	1.29 ± 0.27	0.12 ± 0.11	$\mathbf{1.01 \pm 0.01}$	29.8	0.21 ± 0.02
	uniform	1.62 ± 0.90	0.15 ± 0.16	1.03 ± 0.04	baseline	$\mathbf{0.12 \pm 0.02}$
Hourglass dependency	ℓ_2-hull	$\mathbf{0.96 \pm 0.33}$	$\mathbf{0.08 \pm 0.09}$	$\mathbf{1.01 \pm 0.00}$	**46.0**	0.21 ± 0.02
	ℓ_2-only	0.97 ± 0.40	0.09 ± 0.09	1.02 ± 0.01	38.6	0.20 ± 0.01
	uniform	1.72 ± 0.54	0.12 ± 0.09	1.04 ± 0.02	baseline	$\mathbf{0.11 \pm 0.01}$
Bimodal clusters	ℓ_2-hull	$\mathbf{0.95 \pm 0.24}$	$\mathbf{0.05 \pm 0.05}$	$\mathbf{1.01 \pm 0.00}$	**46.5**	0.20 ± 0.01
	ℓ_2-only	1.03 ± 0.32	0.06 ± 0.04	1.02 ± 0.00	34.9	0.20 ± 0.01
	uniform	1.63 ± 0.53	0.11 ± 0.10	1.02 ± 0.01	baseline	$\mathbf{0.12 \pm 0.01}$
Sinusoidal dependency	ℓ_2-hull	$\mathbf{1.24 \pm 0.39}$	$\mathbf{0.03 \pm 0.03}$	$\mathbf{1.01 \pm 0.00}$	**42.0**	0.20 ± 0.01
	ℓ_2-only	1.28 ± 0.44	0.07 ± 0.04	$\mathbf{1.01 \pm 0.01}$	15.6	0.21 ± 0.01
	uniform	1.36 ± 0.40	0.07 ± 0.04	1.02 ± 0.01	baseline	$\mathbf{0.11 \pm 0.01}$

Note: The results in the table are the mean ± 1 standard deviation of independent simulations. The Relative Improvement is calculated as the average percentage improvement across all metrics, where improvement for parameter ℓ_2 distance and λ error is $(\text{baseline} - \text{method})/\text{baseline} \times 100\%$, and for likelihood ratio it is $(|\text{baseline} - 1| - |\text{method} - 1|)/|\text{baseline} - 1| \times 100\%$. The parameters ℓ_2 distance and λ error are better when smaller, and the log-likelihood ratio is better when closer to 1. Negative relative improvements are shown as 0. The best performance for each metric in each data generation process is highlighted in **bold**.

Figure 3.10, the red line corresponds to our proposed method combining ℓ_2 subsampling with convex hull approximation, the blue line denotes the ℓ_2 subsampling method alone, and the green line indicates the traditional uniform subsampling method. All simulations are based on an original dataset size of 10 000 points.

FIGURE 3.12: Computation time for 9 simulation distributions

To further illustrate the advantages of our coreset approach, we show the effect of reconstructing the univariate marginal densities after constructing the coreset with three different sampling strategies on simulated data from a binary normal distribution, with realistic marginal distribution predictions for X and Y in Figure 3.13 and Figure 3.14, respectively. Each column corresponds to a different coreset size $k = 50, 100, 500$, while the three rows show uniform sampling, ℓ_2 sampling, and ℓ_2-hull sampling, respectively. To reflect the stability of the methods, each subplot of the figure plots a single estimate from 10 replicate trials (thin light-gray colored line) and shows their average result as a solid black line, while the red line gives the true theoretical density. It can be seen that when the coreset size is small (e.g., $k = 50$), uniform sampling is too random and often fails to cover both ends of the distribution, leading to significant deviations in the predicted curves; using only ℓ_2 leverage sampling, with a relatively small coreset size, there is still a relatively high risk of failure.; and introduction of the convex hull can effectively ensure fits the theoretical curves better at the minimum size. As k increases, the average predictions of all three methods gradually converge to the red true density - but at any scale, the ℓ_2 + convex hull scheme reproduces the main shape of the distribution earliest and most consistently, fully reflecting its ability to efficiently capture the distribution with small samples.

Based on the full experimental results in Tables 3.1 and 3.2, the experimental results clearly show that even in the extreme case of fitting the model using only a few dozen points, our proposed coreset method achieves a satisfactory approximation. To

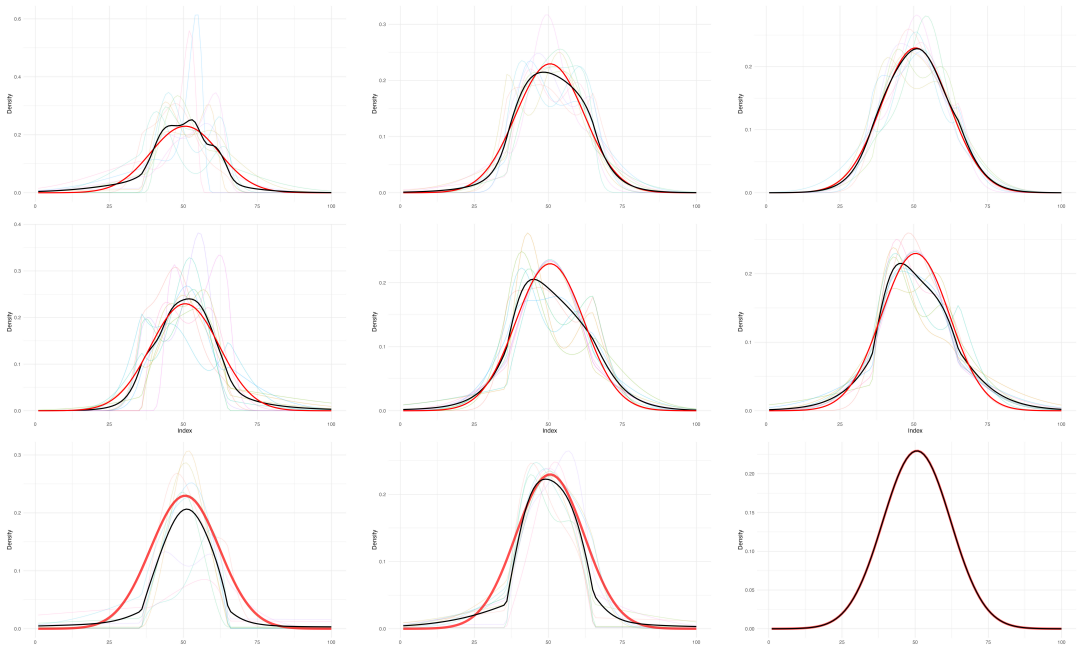


FIGURE 3.13: Predicted marginal density of x for the normal distribution of different coreset size, $k = 50, 100, 500$. First Row: uniform subsampling. Second row: Only ℓ_2 sampling. Third row: ℓ_2 with ε-kernel convex hull

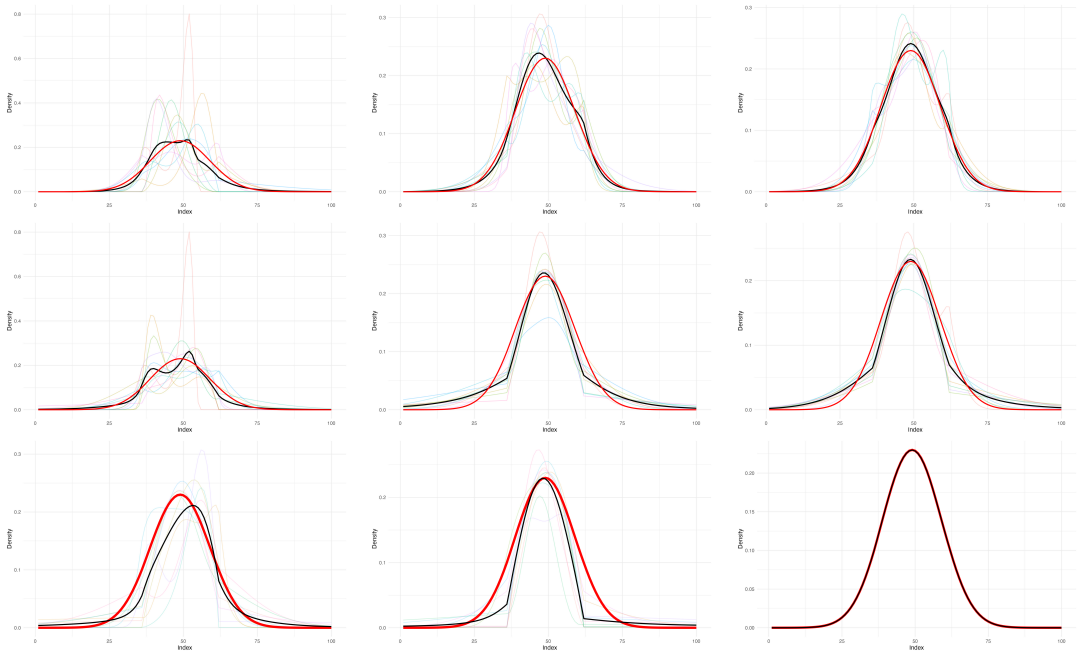


FIGURE 3.14: Predicted marginal density of y of different coreset size, $k = 50, 100, 500$. First Row: uniform subsampling. Second row: Only ℓ_2 sampling. Third row: ℓ_2 with ε-kernel convex hull

ensure the reliability of our experimental results, we performed 10 independent repetitions of each simulation, every time using a different random seed, thus introducing randomness required for obtaining the standard deviations of results. Therefore, we conclude that out of all 14 DGPs, our proposed coreset method (ℓ_2-hull) significantly

outperforms uniform subsampling in 12 scenarios, and fails to show an advantage only in two scenarios, `t-copula` and `skew-t distribution`. This suggests that our coreset method has a robust and effective performance across a wide range of data dependence structures, and is particularly suitable for dealing with non-linearities, heteroscedasticity, complex dependence structures, and multimodality. Our results thus highlight the stability and universality of our method.

Real-World Data Experiments

We now evaluate our method on two large-scale and high-dimensional datasets from different fields of real-world applications.

- **Forest cover data (Covertype).** The UCI Covertype dataset (Blackard, 1998) contains $n = 581{,}012$ samples and 54 variables describing terrain attributes; we focus on the 10 continuous variables (e.g., elevation, slope, hillshade, distances) and a subsample of size $n = 300\,000$. We report comparisons of ℓ_2-hull versus uniform sampling on parameters ϑ error, λ error, log-likelihood ratio, and running time in Table 3.3 and Figure 3.16.

- **Equity returns.** We consider two stock-return datasets: one comprising 10 major stocks' forty-year daily returns ($n \approx 10\,000$), and another with 20 stocks. Performances at varying coreset sizes $k \in \{50, 100, 200, 300\}$ are summarized in Tables 3.4 and 3.5, and Figure 3.17 visualizes the log-likelihood ratio, parameter ℓ_2 distance and λ distance.

Covertype Dataset

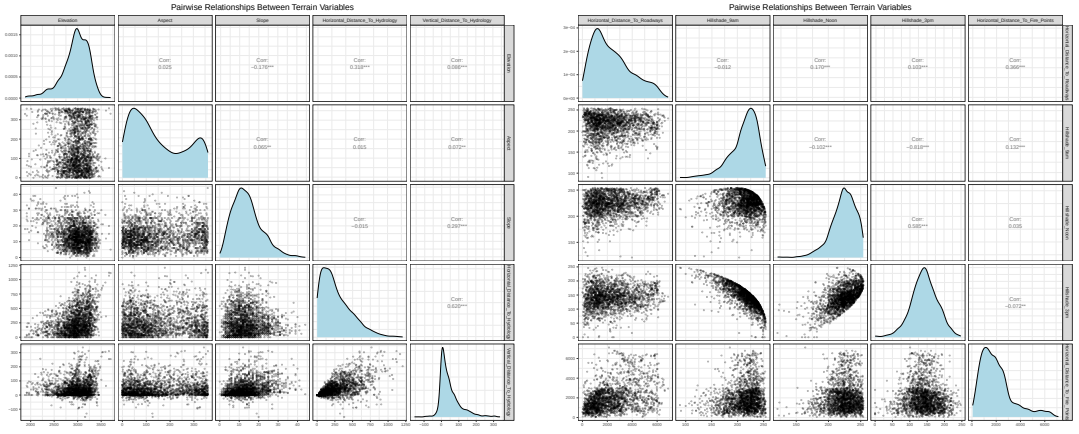
We demonstrate our method on the widely used UCI Covertype dataset (Blackard, 1998), originally intended for forest cover type classification. The full dataset comprises $n = 581\,012$ observations and 54 variables, among which 10 continuous variables represent various terrain features, such as elevation, slope, aspect, distances to hydrological features, and hillshade indices.

This dataset presents several practical challenges motivating the use of MCTM. Specifically, the continuous variables exhibit complex joint dependency structures characterized by multimodality, heavy skewness, and highly non-linear pairwise interactions. Standard Gaussian or parametric copula approaches typically fail to capture these features adequately, thus justifying the adoption of MCTM, which flexibly models the multivariate distribution through non-linear transformations.

However, fitting a full MCTM on large-scale datasets is computationally intensive and sometimes fails. For example, in our case, on the Covertype dataset, when we attempted to fit the 10-dimensional MCTM model to the original $n = 581\,012$ size dataset, hardware limitations caused this fit to simply crash. Therefore, we selected a subsample of $n = 300\,000$ as the original model, and directly fitting an MCTM with only 10 continuous variables on $n = 300\,000$ observations already requires several

hours of computation time. The computational challenge rapidly intensifies with higher dimensions or larger sample sizes, thus providing a natural and compelling scenario for demonstrating the practical benefits of our coreset approach.

We use the Covertype dataset to empirically assess whether coresets can efficiently approximate the full-data MCTM while substantially reducing computational costs, thereby enabling scalable probabilistic modeling of large, high-dimensional datasets.



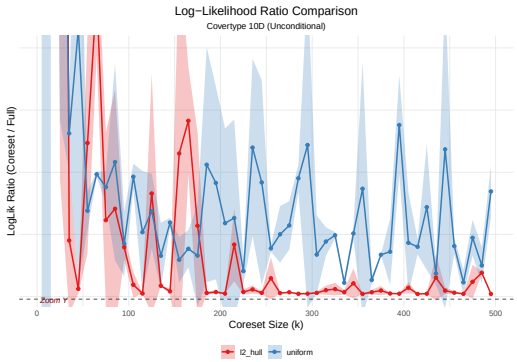
(A) Elevation, Aspect, Slope, Horizontal and Vertical Distance to Hydrology.

(B) Roadways, Hillshade at different times, and Fire Points.

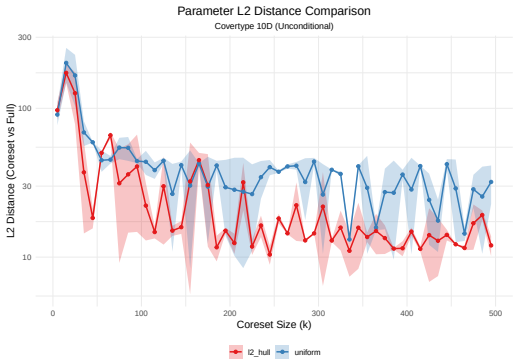
FIGURE 3.15: Pairwise relationships between different sets of terrain variables from the Covertype dataset.

As can be seen from the Figure 3.16, our proposed ℓ_2-`hull` method significantly outperforms the uniform sampling (`uniform`) in all the four evaluation metrics. Specifically, ℓ_2-`hull` is closer to 1 in the log-likelihood ratio, which indicates a more accurate approximation of the original model; it maintains a smaller error in the ℓ_2 distance between the ϑ parameters and λ as well, which proves that it is able to better preserve the distributional structure of the original data in the high-dimensional parameter space; and at the same time, its overall running time is comparable to that of uniform sampling, which does not introduce any additional significant overheads. It can be seen that for high-dimensional large-scale datasets, the ℓ_2-`hull` method, which combines ℓ_2 subsampling with convex hull techniques, outperforms simple uniform sampling in terms of both performance and efficiency.

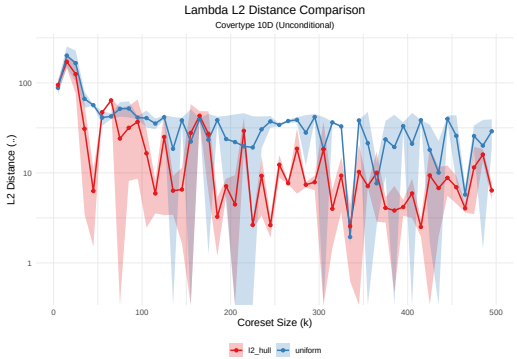
Through experiments on the real-valued features of the Covertype dataset (as shown in Table 3.3), we find that the proposed coreset method (especially the hybrid sampling strategy combining ℓ_2 sensitivity with convex hull approximation) significantly outperforms the uniform subsampling baseline at different coreset sizes $k \in \{50, 100, 200, 500\}$. In particular, when the coreset size is small (e.g., $k = 50$), where the performance of uniform sampling deteriorates, our method can effectively preserve the approximation accuracy of the model, demonstrating the ability to capture important key data points. In addition, our coreset method remains stable and maintains its advantage over uniform sampling as the subset size increases. This indicates that our proposed method not only effectively reduces the computational cost



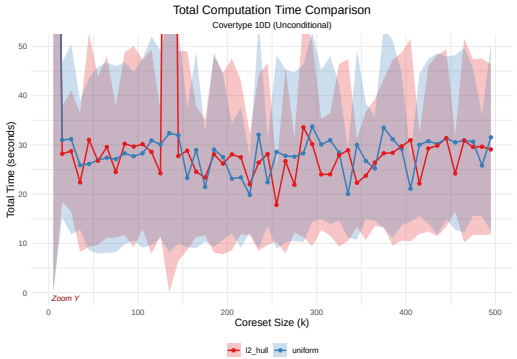
(A) Log-Likelihood Ratio Comparison



(B) Parameter ℓ_2 Distance Comparison



(C) Lambda ℓ_2 Distance Comparison



(D) Total Computation Time Comparison

FIGURE 3.16: Covertype dataset (10-dimensional, unconditional model) on ℓ_2-`hull` versus uniform sampling (`uniform`) compared with respect to four evaluation metrics: (a) log-likelihood ratio, (b) parameter-space ℓ_2 distances, (c) ℓ_2 distances dependent on parameter λ, (d) Total computation time.

on real-world datasets from several hours to a few seconds, but also accurately approximates the Benchmark MCTM model fitted to the complete data, thus achieving efficient and accurate large-scale and high-dimensional probabilistic modeling.

TABLE 3.3: Relative performance comparison on Covertype data for different coreset sizes. Results are mean ± 1 standard deviation over 5 valid trials. Relative improvement is calculated as the average percentage improvement across parameter ℓ_2 distance, λ error, and Log-Likelihood relative to the uniform baseline. Bold indicates the best performance for each metric and coreset size combination.

Coreset Size	Method	Param. ℓ_2 dist.	λ error	Likelihood ratio	Rel. Impr.(%)	Total time(s)
$k = 50$	ℓ_2-hull	$\mathbf{18.332 \pm 2.731}$	$\mathbf{6.290 \pm 4.779}$	$\mathbf{3.144 \pm 1.924}$	**84.7**	31.03 ± 21.75
	ℓ_2-only	40.215 ± 29.062	34.588 ± 33.338	56.179 ± 77.498	39.3	$\mathbf{21.52 \pm 10.58}$
	uniform	59.083 ± 0.280	56.399 ± 0.521	107.654 ± 101.987	baseline	26.13 ± 17.52
$k = 100$	ℓ_2-hull	40.672 ± 25.897	36.896 ± 28.263	19.863 ± 4.852	8.3	29.67 ± 20.42
	ℓ_2-only	$\mathbf{17.719 \pm 1.607}$	$\mathbf{9.381 \pm 2.634}$	$\mathbf{1.208 \pm 0.051}$	**75.7**	28.00 ± 21.17
	uniform	44.206 ± 1.418	41.173 ± 1.630	21.304 ± 13.110	baseline	$\mathbf{27.71 \pm 17.07}$
$k = 200$	ℓ_2-hull	$\mathbf{15.059 \pm 0.621}$	$\mathbf{7.104 \pm 1.542}$	1.859 ± 0.702	**71.0**	$\mathbf{26.21 \pm 18.42}$
	ℓ_2-only	16.277 ± 1.718	9.080 ± 1.645	$\mathbf{1.211 \pm 0.049}$	67.3	27.48 ± 20.50
	uniform	29.472 ± 15.590	23.747 ± 19.244	45.708 ± 38.764	baseline	27.54 ± 16.81
$k = 500$	ℓ_2-hull	$\mathbf{11.992 \pm 1.826}$	$\mathbf{6.383 \pm 1.415}$	$\mathbf{1.090 \pm 0.012}$	**78.6**	$\mathbf{29.09 \pm 17.27}$
	ℓ_2-only	15.749 ± 0.885	9.502 ± 1.200	1.138 ± 0.015	71.0	29.86 ± 18.08
	uniform	31.998 ± 9.277	28.978 ± 10.357	42.306 ± 10.473	baseline	31.53 ± 19.05
Benchmark MCTM	$n = 100\,000$	0	0	1	0	4 837
	$n = 300\,000$	0	0	1	0	17 432

Equity Return Dataset

In finance, returns on multiple stocks often have complex non-linear and time-varying dependence structures. To capture these dependencies, copula models are widely used in risk management and asset allocation. However, traditional copula mostly need to specify the marginal distributions and assume fixed dependence parameters, which makes it difficult to simultaneously model the dynamic changes of the marginals and conditional correlations. MCTM can more comprehensively reflect the characteristics of the joint distribution of financial asset returns by jointly estimating the marginal distributions and the dependence structures through a flexible transformation function, and thus has a good prospect of being applied in the modeling of stock returns. Therefore, we select 10 and 20 representative stock returns data, see Table 3.6 and Table 3.7, respectively, construct their joint distributions based on the MCTM, and combine the coreset method proposed in this paper to improve the computational efficiency of model fitting.

In Table 3.4 (10 stocks) and Table 3.5 (20 stocks) we can see that in most of the experimental scenarios, the ℓ_2-hull method achieves excellent performance, especially in the two metrics of ℓ_2 distance and log-likelihood ratio in the parameter space, which are comparable compared to the pure ℓ_2 sampling scheme. This may be due to the fact that in these scenarios, there are not many extreme points and the convex hull approximation does not add a significant advantage. In addition to this, the ℓ_2-hull is far superior to uniform subsampling in terms of parameter error and log-likelihood ratios in the vast majority of scenarios, and is no slouch in terms

TABLE 3.4: Performance comparison on 10 stock return series for different coreset sizes (1985-2025)

Coreset Size	Method	Param. ℓ_2 dist.	λ error	Log-likelihood ratio	Rel. Impr. L2 / λ / LL (%)	Total time (s)
$k = 50$	ℓ_2-hull	$\mathbf{45.518 \pm 1.643}$	2.992 ± 0.530	1.683 ± 0.234	12.0 / 0.0 / 57.5	8.21 ± 2.30
	ℓ_2-only	45.784 ± 0.554	2.599 ± 0.629	$\mathbf{1.351 \pm 0.082}$	11.5 / 0.0 / 78.2	8.45 ± 2.53
	uniform	51.745 ± 2.474	$\mathbf{1.515 \pm 0.303}$	2.610 ± 0.101	baseline	$\mathbf{7.98 \pm 1.88}$
$k = 100$	ℓ_2-hull	$\mathbf{39.888 \pm 0.643}$	1.613 ± 0.106	1.399 ± 0.147	22.1 / 0.0 / 89.3	9.93 ± 2.58
	ℓ_2-only	41.183 ± 1.503	1.404 ± 0.114	$\mathbf{1.153 \pm 0.044}$	19.6 / 0.0 / 96.1	9.80 ± 2.91
	uniform	51.195 ± 2.616	$\mathbf{0.913 \pm 0.140}$	4.926 ± 0.390	baseline	$\mathbf{8.94 \pm 2.10}$
$k = 200$	ℓ_2-hull	$\mathbf{33.722 \pm 1.242}$	1.050 ± 0.133	1.236 ± 0.145	29.6 / 0.0 / 80.9	11.37 ± 2.89
	ℓ_2-only	38.031 ± 0.282	1.147 ± 0.140	$\mathbf{1.085 \pm 0.004}$	20.6 / 0.0 / 93.1	11.40 ± 3.51
	uniform	47.881 ± 1.217	$\mathbf{0.621 \pm 0.108}$	2.242 ± 0.149	baseline	$\mathbf{10.13 \pm 2.69}$
$k = 300$	ℓ_2-hull	$\mathbf{29.814 \pm 1.363}$	0.905 ± 0.100	1.127 ± 0.081	33.6 / 0.0 / 92.1	13.95 ± 4.48
	ℓ_2-only	35.107 ± 0.916	0.851 ± 0.091	$\mathbf{1.070 \pm 0.007}$	21.8 / 0.0 / 93.5	$\mathbf{12.63 \pm 3.13}$
	uniform	44.915 ± 3.172	$\mathbf{0.488 \pm 0.050}$	2.079 ± 0.171	baseline	13.01 ± 3.91

Results are mean ± 1 standard deviation over 5 valid trials. Bold indicates the best (lowest for error metrics, likelihood ratio closer for 1 for better) per coreset size and metric. Relative improvement (%) over the uniform baseline shown per metric (Param L2 / Lambda / LL

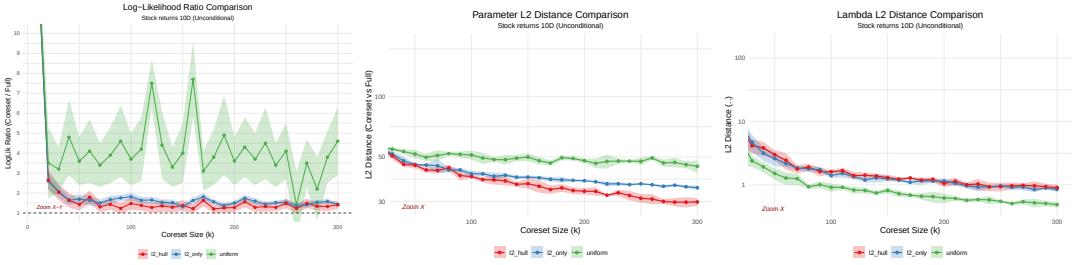
TABLE 3.5: Performance comparison on 20 stock return series for different coreset sizes (1985-2025)

Coreset Size	Method	Param. ℓ_2 dist.	λ error	Log-likelihood ratio	Rel. Impr. L2 / λ / LL (%)	Total time (s)
$k = 50$	ℓ_2-hull	54.254 ± 1.367	3.944 ± 0.563	5.208 ± 0.210	3.6 / 0.0 / 0.0	30.95 ± 5.06
	ℓ_2-only	$\mathbf{53.922 \pm 1.280}$	3.920 ± 0.344	30.303 ± 0.426	4.2 / 0.0 / 0.0	31.10 ± 6.00
	uniform	56.263 ± 1.042	$\mathbf{3.490 \pm 0.260}$	$\mathbf{4.504 \pm 0.200}$	baseline	$\mathbf{29.00 \pm 7.55}$
$k = 100$	ℓ_2-hull	$\mathbf{46.280 \pm 1.887}$	2.270 ± 0.138	$\mathbf{1.675 \pm 0.154}$	13.4 / 0.0 / 89.6	35.56 ± 7.94
	ℓ_2-only	47.385 ± 0.788	2.304 ± 0.370	1.718 ± 0.141	11.4 / 0.0 / 88.9	$\mathbf{34.95 \pm 6.53}$
	uniform	53.469 ± 2.072	$\mathbf{2.006 \pm 0.151}$	7.518 ± 0.305	baseline	35.22 ± 6.65
$k = 200$	ℓ_2-hull	40.225 ± 3.690	1.511 ± 0.138	1.574 ± 0.147	20.4 / 0.0 / 62.3	49.73 ± 7.74
	ℓ_2-only	$\mathbf{38.852 \pm 0.925}$	1.458 ± 0.079	$\mathbf{1.131 \pm 0.044}$	23.1 / 0.0 / 91.4	52.99 ± 12.33
	uniform	50.531 ± 1.513	$\mathbf{1.224 \pm 0.158}$	2.525 ± 0.067	baseline	$\mathbf{45.91 \pm 7.39}$
$k = 300$	ℓ_2-hull	$\mathbf{35.819 \pm 3.552}$	1.165 ± 0.099	1.400 ± 0.107	23.6 / 0.0 / 75.2	$\mathbf{60.99 \pm 6.86}$
	ℓ_2-only	35.036 ± 0.286	1.137 ± 0.132	$\mathbf{1.109 \pm 0.033}$	25.3 / 0.0 / 93.2	61.41 ± 10.14
	uniform	46.906 ± 3.669	$\mathbf{1.028 \pm 0.044}$	2.617 ± 0.195	baseline	63.89 ± 12.15

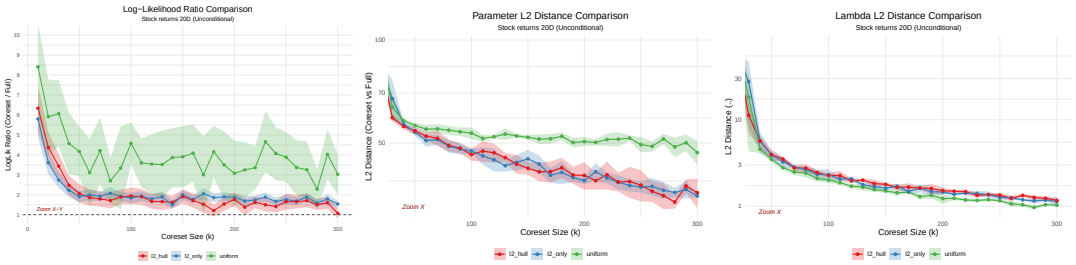
Results are mean ± 1 standard deviation over 5 valid trials. Bold indicates the best (lowest for error metrics, likelihood ratio closer for 1 for better) per coreset size and metric. Relative improvement (%) over the uniform baseline shown per metric (Param L2 / Lambda / LL

of estimation accuracy dependent on the structural parameter λ. In summary, the combination of sensitivity sampling and convex hull technique not only maintains the comparable effect with simple ℓ_2 sampling, but also outperforms uniform sampling when facing sparse extremes and high-dimensional complex structures, and achieves a good balance between high accuracy and high efficiency.

Across both real-world applications, our proposed ℓ_2-hull strategy consistently outperforms uniform subsampling, achieving much tighter log-likelihood approximations and smaller parameter errors, at comparable running times. These results confirm that coresets enable scalable, accurate MCTM fitting on truly large and high-dimensional datasets.



(A) 10-stock log-likelihood ratio (B) 10-stock parameter ϑ distance (C) 10-stock parameter λ distance



(D) 20-stock log-likelihood ratio (E) 20-stock parameter ϑ distance (F) 20-stock λ distance

FIGURE 3.17: Coreset performance comparisons on stock-return datasets. Top row: results for 10 stocks; bottom row: results for 20 stocks. From left to right: (a) log-likelihood ratio, (b) parameter ϑ ℓ_2 distance, (c) parameter λ ℓ_2 distance. Shaded bands indicate ± 1 standard deviation over multiple independent repetitions, and solid lines show the averages over the repetitions.

3.3.6 Summary and Conclusion

This study focuses on the *coreset* approach to enhance efficiency and scalability of fitting MCTM models to large datasets. Despite MCTM's advantages in flexibly modeling complex multivariate joint distributions, its computational burden increases significantly with increasing sample size and output dimension, which hinders its direct application to real-world Big Data scenarios. To this end, we develop a novel coreset construction algorithm, which integrates subsampling based on ℓ_2 leverage scores with convex hull selection. On the one hand, the ℓ_2 leverage scores ensure that

TABLE 3.6: List of 10 Selected Stocks

Ticker	Company Name
JNJ	Johnson & Johnson
PG	Procter & Gamble Co.
KO	The Coca-Cola Company
XOM	Exxon Mobil Corporation
WMT	Walmart Inc.
IBM	International Business Machines Corporation
GE	General Electric Company
MMM	3M Company
MCD	McDonald's Corporation
PFE	Pfizer Inc.

TABLE 3.7: List of 20 Selected Stocks

Ticker	Company Name
JNJ	Johnson & Johnson
PG	Procter & Gamble Co.
KO	The Coca-Cola Company
XOM	Exxon Mobil Corporation
WMT	Walmart Inc.
IBM	International Business Machines Corporation
GE	General Electric Company
MMM	3M Company
MCD	McDonald's Corporation
PFE	Pfizer Inc.
AAPL	Apple Inc.
MSFT	Microsoft Corporation
INTC	Intel Corporation
CSCO	Cisco Systems Inc.
AMGN	Amgen Inc.
CMCSA	Comcast Corporation
COST	Costco Wholesale Corporation
GILD	Gilead Sciences Inc.
SBUX	Starbucks Corporation
TOT	TotalEnergies SE

the samples focus on particularly informative observations. On the other hand, the convex hull captures the extreme value patterns of the distribution. In experiments, it has been demonstrated that the ℓ_2-`hull` algorithm reduces the data to a negligible proportion of their original size. Moreover, the fitting accuracy (log-likelihood ratio, parameter ϑ, λ distances) are virtually unchanged in contrast to pure ℓ_2 sampling and the simplest uniform sampling. Our work comprehensively evaluates fourteen 2-dimensional simulation scenarios and two real-world datasets, with high-dimension and large sample sizes. Our findings demonstrate that ℓ_2-`hull` outperforms uniform sampling in 12 out of 14 simulated scenarios. Furthermore, it achieves significantly superior statistical performance in the context of high-dimensional real-world data. Notably, the running time of ℓ_2-`hull` is comparable to that of uniform sampling, yet it is neither inferior to nor merely comparable to pure ℓ_2 sampling.

The main contribution of this paper is the first introduction of the coreset method to the complex semi-parametric MCTM framework. Unlike previous coresets, which were limited to parametric models such as (generalized) linear regression, we propose a new sampling scheme that naturally combines ℓ_2 leverage score sampling with iterative convex hull selection, and provide a rigorous theoretical proof of the error bounds, which ensures that downstream statistical performance measures are retained in the large-scale data regime. Furthermore, we discuss the intrinsic connection between MCTMs and normalizing flows (NFs), which have become popular in machine learning recently: both of them map complex distributions to simple reference distributions through invertible transformations, thus achieving highly flexible distribution modeling. This not only provides a new perspective on the connection of semi-parametric transformation models with deep generative models (Herp et al., 2025), but also opens up new research directions for inserting coreset techniques into flow models or combining MCTMs with NFs more closely in the future. Future research directions of our coreset approach include extensions to semi-parametric additive or mixed models or to Bayesian settings. Online streaming updating, and distributed calculation of our coreset construction can be explored to adapt to time- and location-varying data scenarios. Finally, it is an intriguing question how far our methods can be generalized to build coresets for NFs.

Chapter 4

Efficient Computation Methods for Distribution Distances

4.1 Introduction

Building upon the insights developed in the previous chapters, this chapter shifts focus from model construction to the principled evaluation of probabilistic outputs—particularly those arising from Monte Carlo simulations. In Chapter 3, we proposed coreset-based approaches to scalable Bayesian inference, where the quality of posterior approximation was theoretically characterized using Wasserstein-based bounds. This naturally raises a broader question central to modern statistics and machine learning: how should we measure the discrepancy between two probability distributions, especially when the only access to them is through samples?

In contemporary Bayesian computation and generative modeling, Monte Carlo methods remain the cornerstone for approximating high-dimensional integrals and posterior distributions. Yet, the reliability of such approximations heavily depends on the quality of the generated samples. As a result, assessing the fidelity of Monte Carlo samples becomes a fundamental task—not merely from the viewpoint of convergence diagnostics, but also in quantifying how close the empirical distribution is to the true target. This chapter thus aims to formalize distributional discrepancy metrics such as the Wasserstein distance, MMD, and related kernel-based methods, and investigates their use in evaluating the performance of Monte Carlo procedures under both theoretical and practical lenses.

By establishing connections between these metrics and the statistical properties of Monte Carlo estimates, we aim to provide a rigorous foundation for comparing different sampling strategies and developing more effective sample quality diagnostics in probabilistic modeling workflows.

While theoretical metrics for comparing distributions are well-established, their practical implementation and evaluation in applied research settings often remain ad hoc and inconsistent. In scientific domains, Monte Carlo-based inference pipelines are widespread, yet researchers frequently rely on heuristic visual checks, marginal histograms, or convergence diagnostics that do not quantitatively capture the multivariate structure of the target distribution.

Our development of the `MCbench` benchmark suite is motivated by a practical gap in standardized, quantitative, and reproducible methods for evaluating Monte Carlo sampling quality. We observed that in many realistic applications, such as posterior simulation via MCMC or evidence estimation via Nested Sampling, the absence of effective diagnostic tools often leads to unreliable inference—particularly when working with high-dimensional or multi-modal distributions where visual inspections fail. Moreover, across collaborations with physicists and engineers, we found a consistent demand for lightweight and extendable frameworks that can compare empirical samples against ground-truth IID samples using scalable discrepancy metrics such as sliced Wasserstein and MMD.

By offering a modular `Julia`-based framework, `MCbench` serves both as a practitioner-oriented tool and a research prototype for sample diagnostics. It allows users to input external samples—e.g., from `Stan`, `PyMC`, or custom samplers—and obtain quantitative feedback about how these samples compare to the ideal benchmark. This aligns with a growing emphasis in modern scientific computing: not only do we require Monte Carlo estimates, but also transparent justifications of their fidelity.

MCBench thus bridges a methodological gap between theory and practice. It offers concrete implementation of advanced distance metrics, optimized for computational efficiency, while remaining agnostic to the underlying application domain. This makes it a practical contribution not just in Bayesian computation, but also in applied disciplines where probabilistic sampling is central.

4.2 Background

In this paper, we present `MCbench`, a benchmark suite designed to assess the quality of Monte Carlo (MC) samples. The benchmark suite enables quantitative comparisons of samples by applying different metrics, including basic statistical metrics as well as more complex measures, in particular the sliced Wasserstein distance and the maximum mean discrepancy. We apply these metrics to point clouds of both independent and identically distributed (IID) samples and correlated samples generated by MC techniques, such as Markov Chain Monte Carlo or Nested Sampling. Through repeated comparisons, we evaluate test statistics of the metrics, allowing to evaluate the quality of the MC sampling algorithms.

Our benchmark suite offers a variety of target functions with different complexities and dimensionalities, providing a versatile platform for testing the capabilities of sampling algorithms. Implemented as a Julia package, `MCbench` enables users to easily select test cases and metrics from the provided collections, which can be extended as needed. Users can run external sampling algorithms of their choice on these test functions and input the resulting samples to obtain detailed metrics that quantify the quality of their samples compared to the IID samples generated by our package. This approach yields clear, quantitative measures of sampling quality and allows for informed decisions about the effectiveness of different sampling methods.

By offering such a standardized method for evaluating MC sampling quality, our benchmark suite provides researchers and practitioners from many scientific fields, such as the natural sciences, engineering, or the social sciences with a valuable tool for developing, validating and refining sampling algorithms.

A large number of scientific fields use numerical methods to address mathematical problems which, in most cases, cannot be solved analytically. A common variant is the Monte Carlo (MC) method. It is based on drawing random numbers from (probability) distributions defined by the problem at hand and the calculation of suitable quantities, e.g., expected values. The MC method is mostly used for integration, optimization and uncertainty propagation. An example for an intensive application of the MC method in the natural sciences is the simulation of scattering processes in particle physics, where – experimentally – two types of particles are repeatedly collided head-on under the same conditions and the resulting interactions are measured by large detectors. The processes and their resulting experimental signatures can be accurately modeled, but only be calculated numerically. Another example stems from the engineering sciences, particularly from civil engineering, where the tilting of the foundation in consolidation problems of soil is the quantity of interest. There, the tilting is calculated exactly by a finite element method for specific, fixed values of its inputs, the soil parameters. However, the soil parameters are subject to uncertainties, which leads to an uncertain result. To model this uncertainty, MC sampling in combination with a Kriging model is employed in frequentist as well as Bayesian analyses, see e.g. Williams and Rasmussen (2006) and Meegen et al. (2025).

A key challenge when using the MC method is to obtain a sampling distribution that represents the target function sufficiently well. While drawing random numbers for some specific distributions is trivial, a variety of methods have been developed since the late 1940s to obtain samples from arbitrary distributions. Besides the brute force approach of the good old "hit & miss" algorithm, some of the most notable examples of refined methods include importance sampling and other variance-reduction techniques, as well as variants of random walk-inspired Markov Chain Monte Carlo (MCMC), e.g. the traditional Metropolis algorithm (Metropolis et al., 1953). For most applications, it is desirable that the resulting sample of random numbers is independently and identically distributed (IID), so that all random numbers are mutually independent. Samples generated by MCMC algorithms, for example, are not IID and often show a significant amount of autocorrelation.

In specific applications, the choice of the sampling algorithm depends on the characteristics of the probability distribution under investigation, in particular its dimension and general topology. Conveniently, a variety of software packages and tools have been made available in recent years that offer sampling algorithms either in a specific context, e.g. MCMC methods in conjunction with statistical inference such as `Stan` (Carpenter et al., 2017) or `BAT.jl` (Schulz et al., 2021), or as stand-alone numerical methods. Regardless of whether one uses existing tools or develops one's own software code, it is important to evaluate the quality of the generated samples. In

particular, with high-dimensional distributions it is often not easy to assess whether the generated samples represent the target function in all corners of the phase space.

Although the need to compare samples from high-dimensional distributions has been recognized in various fields, e.g. in applications in particle physics (Grossi et al., 2025), there is as yet no general, domain-neutral tool that is modular in terms of test metrics, flexible in the definition of test cases and easy to use. This paper attempts to close this gap by introducing the `MCbench` package as a general benchmark suite developed specifically to evaluate the quality of MC sampling algorithms. The suite provides a selection of concrete test functions – with strong variation in dimension and complexity – from which the user can generate samples. The resulting point clouds are read into the software package and then compared with a large set of IID-produced samples of the same distributions. For this purpose, several uni- and multidimensional metrics with sensitivities to a wide range of different features are evaluated. The paper is structured as follows: after an outline of the basic design idea of the benchmark suite (Section 4.3), the different test functions (Section 4.4) and the various implemented metrics (Section 4.5) are described, followed by the implementation details (Section 4.6). Finally, selected problems are discussed in detail to demonstrate the application of the benchmark suite (Section 4.7).

4.3 Design Ideas and Workflow

The benchmark suite evaluates the quality of a sampling algorithm by comparing sample distributions drawn from well-defined test functions using (a) the algorithm under study and (b) algorithms that produce IID-samples. Several metrics are used for the comparison. The general workflow of the benchmark suite is illustrated in Figure 4.1 and comprises three steps:

Choice of Test Function and Generation of Samples Users select a specific test function from the available list and implement it as a target function in a sampling code of their choice that provides an implementation of the sampling algorithm under test. The test functions vary in dimensionality and complexity. A detailed list of the available test functions is provided in Section 4.4. Example implementations of all test functions included in the list to be used with `Stan`, `PyMC` and `BAT.jl` are available at the GitHub repository of the project[1]. Users generate samples from the target function which are then saved into a file.

Comparison of Distributions The user-generated samples are compared against automatically generated IID samples from the same target function using the metrics detailed in Section 4.5. For the basic statistical measures (Section 4.5.1), both the IID and user-generated samples are partitioned into m batches, each with an (effective) sample size of n. Evaluating these metrics across the batches yields histograms of

[1] https://GitHub.com/tudo-physik-e4/MCBench

the expected values, where the distribution widths capture the statistical variation inherent to the finite sample size. Note that the computation of the effective sample size depends on the specific sampling method and can be implemented individually. As a default, the ratio of the squared sum of the weights divided by the sum of their squares is used as an estimate. The metrics are again evaluated for each of these batches, enabling a direct comparison between the distributions derived from the user-generated samples and those obtained from the IID samples. For the advanced statistical measures described in Section 4.5.2 that involve two-sample comparisons, we evaluate the metrics by performing IID vs. IID comparisons to generate the expected distribution, as well as IID vs. user-generated sample comparisons to assess the performance of the sampling algorithm under investigation.

Visualization of the Results In the final comparison plots, we present the distributions of the metrics in a normalized way, see e.g. Fig. 4.1 (bottom right). The mean value of each metric calculated from the set of IID-sampled distributions are centered around 0 (black vertical line). The regions corresponding to the 1 σ-, 2 σ-, and 3 σ-ranges of each metric calculated from the IID-sampled distribution are colored green, yellow, and red, respectively. By overlaying the mean value and the standard deviation of the metric evaluated on the user-generated samples (marker with horizontal error bars), we provide a straightforward visual comparison between the IID and the user-generated samples.

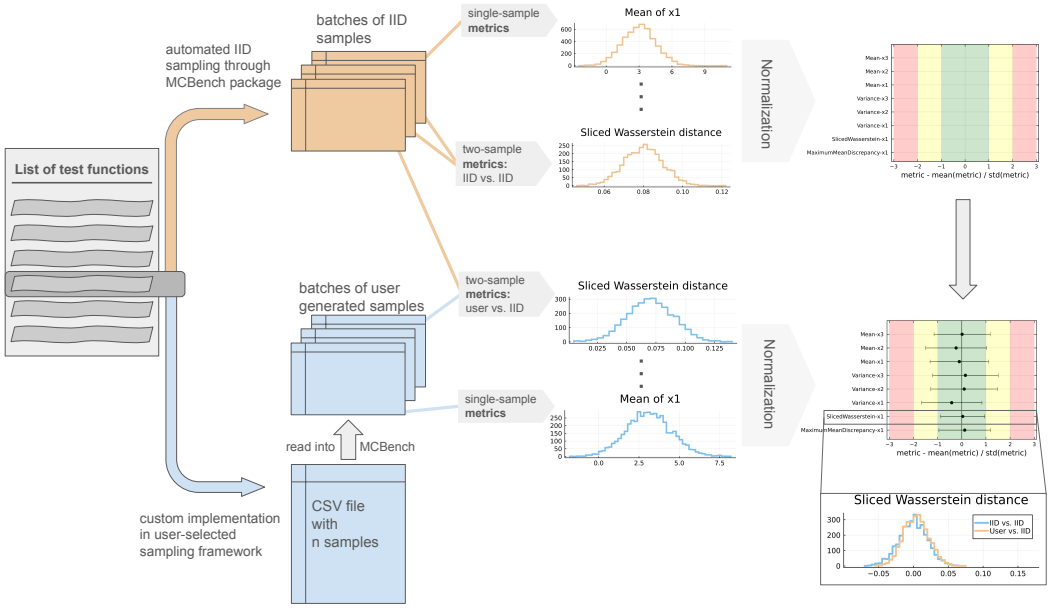


FIGURE 4.1: Illustration of the `MCbench` benchmark suite workflow.

4.4 Target Functions

The benchmark suite provides a number of target functions that cover a wide range of complexities, including varying dimensionality, multi-modality, and correlations. They are chosen to represent practical applications or show specific features that challenge sampling algorithms. All target functions provided are IID-sampleable, which is a requirement for the benchmarking process. The list of target functions used is shown in Table 4.1. We present two detailed walkthrough examples in Section 4.7, namely the one-dimensional Gaussian case and a three-dimensional bimodal distribution.

The target functions include standard normal distributions for multiple dimensionalities, correlated normal distributions, Cauchy distributions and mixture models which are used to create multimodal distributions.

In addition to the standard functions, we also include the eight schools example from the `posteriorDB` database (Magnusson et al., 2024), which is a model used to estimate the effects of coaching programs on student performance. The model consists of three hierarchical prior distributions:

$$\begin{aligned} \mu &= \mathcal{N}(0, 5) \\ \tau &= \text{Cauchy}(0, 5) \cdot \mathbb{I}(\tau > 0) \\ \theta_i &= \mathcal{N}(\mu, \tau) \quad \text{for} \quad i = 1, \ldots, 8\,. \end{aligned}$$

Given σ_i and θ_i, the likelihood is given by:

$$L(\theta, \sigma, y) = \prod_{i=1}^{8} \mathcal{N}(y_i | \theta_i, \sigma_i)\,. \tag{4.1}$$

In order to compare the performance of the samplers, IID samples are drawn using a simple accept-reject algorithm for the eight schools example. These test functions provide a basis for evaluating the performance of the samplers across a range of different distributions and complexities. In addition to the implementation of the test functions in Julia, we also provide examples of how to use the test functions in PyMC and Stan. These examples as well as supplementary information about the test cases, such as evaluation of the function for specific test points, are provided in the GitHub repository for the benchmark suite [2].

4.5 Metrics

Our benchmark suite offers a collection of statistical distance measures, or metrics, to assess discrepancies between two sets of samples and to evaluate if they are drawn from the same target distribution. These metrics vary in their sensitivity to specific features of the distributions and their robustness in detecting discrepancies.

[2] https://GitHub.com/tudo-physik-e4/MCBench

Name	Equation	Properties
Standard Normal 1D	$f(x \mid \mu, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} \cdot e^{-\frac{(x-\mu)^2}{2\sigma^2}}$	Unimodal
Standard Normal kD uncorrelated	$\mathcal{N}_k(\mu, \Sigma) = (2\pi)^{-k/2} \det(\Sigma)^{-1/2} \cdot \exp\left(-\frac{1}{2}(x-\mu)^\top \Sigma^{-1}(x-\mu)\right)$	Unimodal, $k = 2, 3, 10, 100$
Standard Normal kD weakly/strongly correlated	$\mathcal{N}_k(\mu, \Sigma)$ $\Sigma = r \cdot \mathbf{J} + (1-r) \cdot \mathbf{I}$	$r = 0.2, 0.9$ $k = 2, 10, 100$
Mixture Normal kD strongly correlated	$f(x \mid \mu, \Sigma) = 0.25\, \mathcal{N}_k(\mu, \Sigma) + 0.75\, \mathcal{N}_k(-\mu, \Sigma)$ $\Sigma = r \cdot \mathbf{J} + (1-r) \cdot \mathbf{I}$	Multimodal $r = 0.9, k = 3, 10$
Cauchy 1D	$f(x \mid x_0, \gamma) = \frac{1}{\pi\gamma} \cdot \frac{1}{1 + \left(\frac{x-x_0}{\gamma}\right)^2}$	Unimodal
Eight schools example	$L(\theta \mid y) = \prod_{i=1}^{8} \frac{1}{\sqrt{2\pi\sigma_i^2}} \exp\left(-\frac{(y_i - \theta_i)^2}{2\sigma_i^2}\right)$	For more details see Section 4.4

TABLE 4.1: List of test cases used in the benchmark suite including names, equations, and properties. The **J** refers to a matrix filled with ones while **I** refers to the identity matrix. More details about the test cases can be found in the documentation of the benchmark suite.

We employ both basic, well-known metrics and more advanced statistical distance measures, which we briefly summarize in the following section.

4.5.1 Basic Statistical Distance Measures

Traditional descriptive statistics, such as the mean, variance, and chi-square test statistics provide fundamental insights into the characteristics of a data set. These basic measures allow users to easily grasp the distribution's key features, including its central tendency (mean) and variability (variance). They are particularly well-suited for analyzing low-dimensional data or projections of high-dimensional data, offering a preliminary and intuitive understanding of the data distribution. However, these basic statistical measures reach their limitations when analyzing high-dimensional or multimodal data. The overall structure and distributional characteristics of such data are often inadequately captured due to the so-called *curse of dimensionality*. Therefore, introducing more complex statistical distance measures becomes necessary to examine high-dimensional data at a finer level of detail.

In general, a statistical distance should satisfy several properties as follows (Munkres, 2000):

Definition 4.5.1 (Distance Function / Metric)**.** *Let X be a set. A function $d : X \times X \longrightarrow \mathbb{R}$ is called a metric (or distance function) on X if it satisfies the following conditions for all $x, y, z \in X$:*

- ***Non-negativity:***

$$d(x, y) \geq 0.$$

- ***Identity of indiscernibles:***

$$d(x, y) = 0 \iff x = y.$$

- ***Symmetry:***

$$d(x, y) = d(y, x).$$

- ***Triangle inequality:***

$$d(x, z) \leq d(x, y) + d(y, z).$$

4.5.2 Advanced Statistical Distance Measures

Complex distance metrics are specifically designed to measure the similarity of two distributions in high-dimensional and complex cases. In particular, such metrics not only refer to the overall shape of data distributions but also to particular differences between them, and hence, provide a more expressive method of comparison. In general, statistical distance measures quantify the divergence between two probability distributions and are widely used in many data analysis and machine learning tasks. So far, we have implemented two advanced statistical distance measures: the (sliced) Wasserstein distance (SWD) and the maximum mean discrepancy (MMD). Further statistical distance metrics are planned to be implemented in the future.

The Sliced Wassertein Distance

The Wasserstein distance, or Earth Mover's distance, is a statistical measure of dissimilarity between two probability distributions. In the context of optimal transport theory, the Wasserstein distance can be understood as the minimum cost required to transport a mass from one probability distribution to another. This theory was first proposed by the French mathematician Gaspard Monge in 1781 (Monge, 1781) and later further popularized and developed by the Russian mathematician Leonid Kantorovich in the 20th century (Kantorovich, 1942). Recently, the Wasserstein distance has been widely used in different fields of machine learning and statistics. Examples include Wasserstein-Generative-Adversarial-Networks (Arjovsky et al., 2017) and the approximation of the posterior distributions in Bayesian statistics (Ding et al., 2024).

According to Panaretos and Zemel (2019), the p-Wasserstein distance between two probability measures μ and ν is given by

$$W_p(\mu, \nu) = \inf_{\substack{X \sim \mu \\ Y \sim \nu}} (\mathbb{E}\|X - Y\|^p)^{1/p} = \left(\inf_{\gamma \in \Gamma(\mu,\nu)} \int_{\mathcal{X} \times \mathcal{X}} \|x - y\|^p \, d\gamma(x, y) \right)^{1/p}, \quad p \geq 1.$$

Here, the element $\gamma \in \Gamma(\mu, \nu)$ can be viewed as a coupling of μ and ν, which is a joint distribution with the given marginal distributions μ and ν on each axis. In the discrete case, this analytical expression can be interpreted as follows: for $\gamma \in \Gamma(\mu, \nu)$ and any pair of positions (x, y), the value of $\gamma(x, y)$ represents the proportion of the mass of μ that is transferred from x to y in order to transform μ into ν.

Although the Wasserstein distance can accurately measure the distance of two probability distributions, especially for its consideration of the geometric structure of the distributions, the computational complexity is generally large with $O(n^3 \log n)$. Also, it suffers the curse of dimensionality, where the sample complexity increases with the dimension d by $O(n^{-1/d})$, see Nguyen and Ho (2022). This means that when d increases, an exponential increase in sample size is required to maintain the same estimation accuracy.

To overcome the problematic scaling for high-dimensional test cases, the benchmark suite uses the sliced Wasserstein distance. It approximates the difference between high-dimensional distributions by first projecting them onto one dimension along various random directions, then computing the one-dimensional Wasserstein distance for each projection, and finally averaging these one-dimensional distances. This method was first proposed in Rabin et al. (2012) using a random projection approach, and was later extended using the Radon transformation method, see Bonneel et al. (2015). The random-projection-based SWD is defined as:

Definition 4.5.2 (Sliced Wasserstein Distance)**.** *Let μ and ν be probability distributions defined on $\mathbb{R}^d$ with the sliced Wasserstein distance defined as:*

$$SWD_p(\mu, \nu) = \left(\int_{\mathbb{S}^{d-1}} W_p^p \left(P_\theta \mu, P_\theta \nu \right) \, d\theta \right)^{1/p}. \tag{4.2}$$

Here, $\mathbb{S}^{d-1}$ is the unit sphere in $\mathbb{R}^d$. $P_\theta \mu$ and $P_\theta \nu$ denote the one-dimensional projected distributions obtained by projecting μ and ν onto the direction θ respectively. The $W_p \left(P_\theta \mu, P_\theta \nu \right)$ is the p-Wasserstein distance projected onto the direction θ.

Since it is not possible (and not necessary) to project onto all possible θ directions, the Monte Carlo method is used to approximate the above integrals with a finite number of randomly sampled directions $\{\theta_i\}_{i=1}^L$:

$$\mathrm{SWD}_p(\mu, \nu) \approx \left(\frac{1}{L} \sum_{i=1}^{L} W_p^p \left(P_{\theta_i} \mu, P_{\theta_i} \nu \right) \right)^{1/p}. \tag{4.3}$$

The process of implementing the sliced Wasserstein distance can be summarized as follows:

1. Random projection: Project the high dimensional data $x \in \mathbb{R}^d$ into one dimension, i.e., compute $\theta^\top x$, where θ is a random unit vector on the unit sphere $\mathbb{S}^{d-1}$.

2. One-dimensional Wasserstein distance: computing the Wasserstein distance between two distributions in a one-dimensional space has an analytic solution, which can be efficiently accomplished by a sorting operation.

3. Composite result: By computing the one-dimensional Wasserstein distance in multiple random directions and averaging or taking the expected value, the sliced Wasserstein distance is obtained, which serves as an approximation to the high-dimensional Wasserstein distance.

We show here some basic properties of the SWD. The majority of our proofs follow the seminal work established in Bonnotte (2013), which provides a comprehensive theoretical foundation for optimal transportation methods. The following theoretical framework establishes the fundamental characteristics of SWD, with particular emphasis on its metric properties and convergence behavior in probability spaces.

Theorem 4.5.3 (Metric Property)**.** *The Sliced Wasserstein Distance SWD_p defines a proper metric on the space of probability measures.*

Proof. To prove that SWD_p is a metric, we need to verify four properties: non-negativity, symmetry, triangular inequality, and positive definiteness.

Non-negativity and symmetry: Since W_p is non-negative and symmetric, and the integral maintains non-negativity and symmetry, SWD_p naturally satisfies these two properties.

Triangle inequality: Using the triangle inequality for W_p and the linearity of the integral, it can be shown that SWD_p satisfies the triangle inequality.

Positive definiteness: the key is to show that $SWD_p(\mu,\nu) = 0$ implies $\mu = \nu$. Assume $SWD_p(\mu,\nu) = 0$, then for every $\theta \in \mathbb{S}^{d-1}$, we have

$$W_p(\theta\mu, \theta\nu) = 0.$$

This means that for all θ, the projective distributions of μ and ν in the direction θ are the same.

Using the Fourier slice theorem, the Fourier transforms of μ and ν are the same in all directions:

$$\mathcal{F}\mu(s\theta) = \mathcal{F}(\theta\mu)(s) = \mathcal{F}(\theta\nu)(s) = \mathcal{F}\nu(s\theta). \tag{4.4}$$

Due to the invertibility of the Fourier transform on $\mathbb{R}^d$, this shows that $\mu = \nu$. □

Lemma 4.5.4 (Equivalence Bounds, Bonnotte, 2013)**.** *For $\mu,\nu \in \mathcal{P}p(\mathbb{R}^d)$, there exists a constant $c_{d,p}$ such that:*

$$SWD_p(\mu,\nu)^p \leq c_{d,p} W_p(\mu,\nu)^p, \tag{4.5}$$

where $c_{d,p} = \frac{1}{d}\int_{S^{d-1}} |\theta|_p^p d\theta \leq 1$.

Further, regarding the convergence rate, Bonnotte (2013) showed the following Lemma.

Lemma 4.5.5 (Convergence Rate). *For measures supported in $B(0, R)$, there exists a constant $C_d > 0$ such that:*

$$W_1(\mu, \nu) \leq C_d R^{d/(d+1)} SW_1(\mu, \nu)^{1/(d+1)}. \tag{4.6}$$

For detailed proofs of Lemma 1 and Lemma 2, see Bonnotte (2013).

Next, we give some illustrations of how the projection process of the sliced Wasserstein distance works.

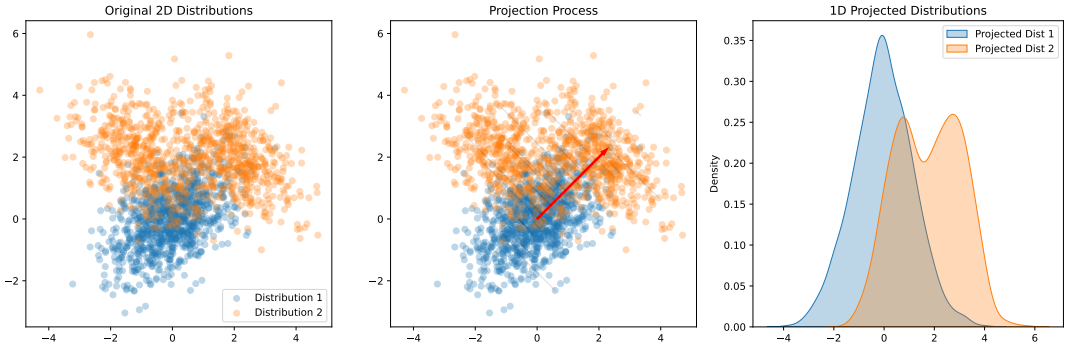


FIGURE 4.2: Illustration of the sliced Wasserstein distance computation process with 2D Gaussian distributions.

The left panel shows two original distributions in 2D space: a single Gaussian distribution (blue points) and a mixture of two Gaussians (orange points). The key insight of SWD is to project these high-dimensional distributions onto one-dimensional spaces. In the specific illustration here, we show a two-dimensional case, as shown by the red projection direction. The middle panel demonstrates this projection process, where each point is mapped onto the chosen direction. The right panel reveals the resulting one-dimensional probability kernel estimation after projection, effectively transforming our original distributional comparison problem into a simpler one-dimensional optimal transport problem.

The effectiveness of the SWD relies on using multiple random projections to capture different aspects of the distributional differences. As illustrated in Figure 4.3, different projection angles reveal distinct characteristics of the distributions. When $\theta = 0$, the projection highlights the horizontal spread of the distributions, while larger angles progressively capture the vertical structural differences. This demonstrates why using multiple random projections is crucial for accurately computing the SWD, as each projection angle may capture different aspects of the distributional discrepancy.

Figure 4.4 provides an empirical analysis of the SWD **in a low-dimensional setting**. The convergence plot (a) demonstrates that the SWD rapidly approaches the true Wasserstein distance within the first 25 projections and stabilizes thereafter. The

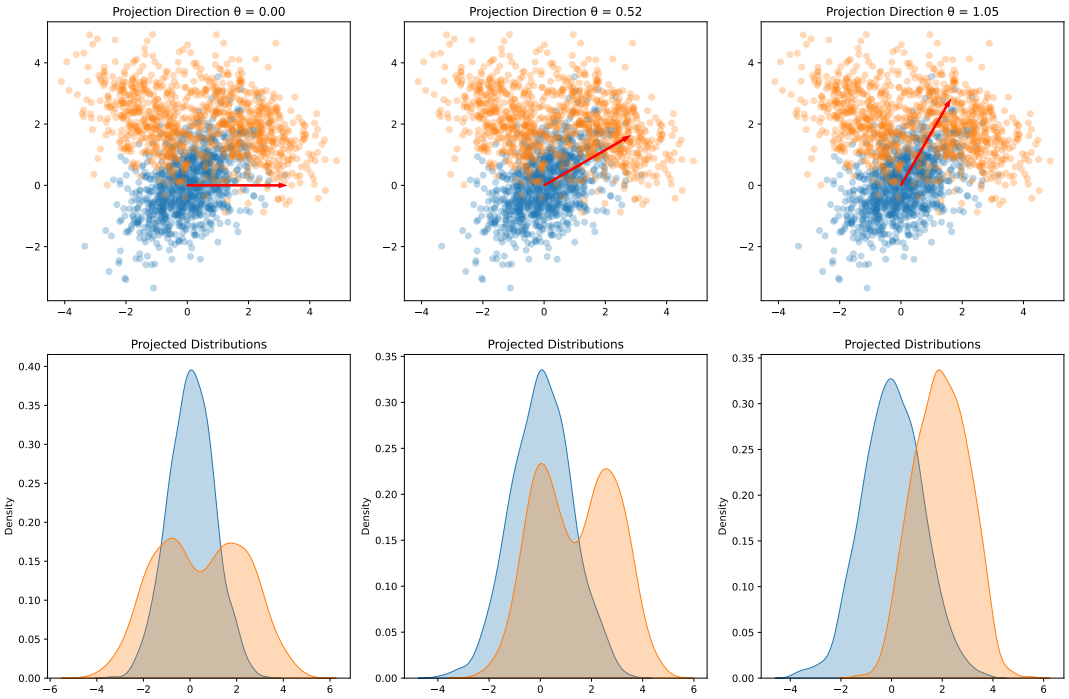
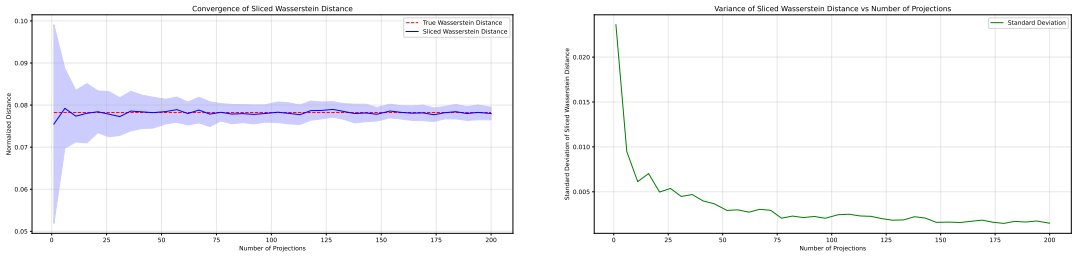


FIGURE 4.3: Effect of different projection angles ($\theta = 0, \pi/6, \pi/3$) on sliced Wasserstein distance computation. These directions are for visual illustrations only.

variance analysis (b) confirms this observation by showing a significant reduction in standard deviation as the number of projections increases. **For this specific low-dimensional scenario**, these results suggest that approximately 50 projections are sufficient for obtaining a reliable approximation while maintaining the computational efficiency of $O(Ln \log n)$, a substantial improvement over the $O(n^3 \log n)$ complexity of the exact Wasserstein computation (Xie et al., 2020; Le et al., 2024).

However, it is crucial to note that this rapid convergence to the true Wasserstein distance does not generalize to high-dimensional spaces. Theoretically, the number of random projections L required to approximate the exact Wasserstein distance within a small error ϵ scales exponentially with the dimension d, suffering from the curse of dimensionality with error bounds roughly proportional to $(1/\epsilon^2)^d$. Consequently, in high-dimensional applications, using a small, fixed number of projections (e.g., $L = 50$) will fail to yield a tight approximation of the original Wasserstein distance. Nevertheless, from a practical standpoint, the SWD remains a highly valuable computational tool in modern scalable inference. In high-dimensional regimes, the objective shifts from strictly approximating the exact Wasserstein distance to utilizing the SWD as an efficient, standalone discrepancy measure. Since the SWD inherently defines a valid distance metric between distributions, using a tractable number of projections is empirically sufficient to capture structural differences and drive model convergence, circumventing the prohibitive $O(n^3 \log n)$ computational bottleneck.



(a) Convergence behavior of SWD

(b) Variance reduction analysis

FIGURE 4.4: Empirical analysis of sliced Wasserstein distance: (a) shows the convergence to true Wasserstein distance as the number of projections increases, with the blue shaded area representing one standard deviation; (b) demonstrates the decreasing trend of standard deviation with more projections, indicating improved stability of the estimation.

Maximum Mean Discrepancy

The maximum mean discrepancy (MMD) measures the distance between two distributions by comparing moments in a high-dimensional feature space. It maps the data into a Reproducing Kernel Hilbert Space (RKHS) and computes the distance between the mean embeddings of the two distributions to measure their divergence. The advantage of MMD is its effectiveness in handling high-dimensional data and its good performance with complex distributions. Gretton et al. (2012a) defines it as

Definition 4.5.6 (Maximum Mean Discrepancy in RKHS). *Let $(\mathcal{H}, k)$ be a reproducing kernel Hilbert space with kernel function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$. For two probability measures p and q defined on a metric space $\mathcal{X}$, assuming $\mathbb{E}_x[k(x,x)] < \infty$ and $\mathbb{E}_y[k(y,y)] < \infty$, the maximum mean discrepancy is defined as:*

$$MMD[p,q] = \|\mu_p - \mu_q\|_{\mathcal{H}} \tag{4.7}$$

where μ_p and μ_q are the mean embeddings of distributions p and q respectively in $\mathcal{H}$, defined as:

$$\mu_p = \mathbb{E}_{x\sim p}[k(x,\cdot)] \quad \textit{and} \quad \mu_q = \mathbb{E}_{y\sim q}[k(y,\cdot)]. \tag{4.8}$$

The squared MMD can be expanded in terms of kernel functions as:

$$\begin{aligned} MMD^2[p,q] = \mathbb{E}_{x,x'\sim p}[k(x,x')] + \mathbb{E}_{y,y'\sim q}[k(y,y')] \\ -2\mathbb{E}_{x\sim p, y\sim q}[k(x,y)]. \end{aligned} \tag{4.9}$$

MMD has been widely applied in numerous machine learning tasks, including two-sample tests, domain adaptation, and generative model evaluation. Compared

to the Wasserstein distance, MMD has an advantage in high-dimensional data processing because it can flexibly capture the higher-order structural features of the data through the choice of kernel functions.

The choice of kernel function in MMD significantly affects its performance and ability to characterize differences between distributions. Commonly used kernels include the Gaussian, Laplacian, Polynomial, Linear, and Sigmoid kernels. Notably, generalized kernel families such as the Matérn kernel or the powered exponential kernel provide a flexible interpolation between the Laplacian and Gaussian kernels (Williams and Rasmussen, 2006). This generalization mathematically mirrors the shape-adaptive nature of the p-generalized normal distribution utilized in our Bayesian p-probit model (discussed in Chapter 2). Recognizing this connection offers a unified perspective on controlling tail behavior and smoothness across both regression modeling and distribution comparison.

Despite the existence of such generalized families, our benchmark suite employs the ubiquitous Gaussian kernel $k_G(x, y)$ as the default choice for evaluation, which is defined as:

$$k_G(x, y) = \exp\left(-\frac{\|x - y\|^2}{2\sigma^2}\right). \tag{4.10}$$

The parameter σ (the kernel width) plays a critical role in determining its sensitivity. A commonly used approach for selecting σ is the *median heuristic*, which sets σ to the median of pairwise distances among data points. As noted by Gretton et al. (2012a), using the median heuristic for bandwidth selection can improve the performance and stability of MMD-based statistical tests. For a more detailed discussion on the characteristic kernels and the parameter choices of the MMD, we refer to Sriperumbudur et al. (2010) and Gretton et al. (2012b).

In practice, the empirical squared MMD can be computed as follows:

$$\begin{aligned}\mathrm{MMD}^2(X, Y) =& \frac{1}{n^2}\sum_{i,j=1}^{n} k(x_i, x_j) + \frac{1}{m^2}\sum_{i,j=1}^{m} k(y_i, y_j) \\ &- \frac{2}{nm}\sum_{i=1}^{n}\sum_{j=1}^{m} k(x_i, y_j),\end{aligned} \tag{4.11}$$

where $k(\cdot, \cdot)$ represents the kernel function, $X = x_1, x_2, \ldots, x_n$ contains n data points, and $Y = y_1, y_2, \ldots, y_m$ contains m data points.

Despite its theoretical elegance and ability to detect differences between distributions, the practical application of MMD faces significant computational challenges, especially when dealing with large-scale datasets.

Therefore, in practice, a common approach is to approximate the kernel calculations. A popular method is to use the Random Fourier Features (RFF). The theoretical justification for RFF comes from Bochner's theorem (Bochner, 1959).

Both, the original MMD calculation for the Gaussian kernel as well as its corresponding RFF approximation are implemented in the benchmark suite. It can be shown that the computational complexity for the original Gaussian MMD is $O(d \cdot mn)$. If $n \approx m$, the computational complexity becomes $O(d \cdot n^2)$. This is because computing all pairwise kernel evaluations requires $O(n^2)$ operations, and each kernel computation involves $O(d)$ time due to the dimensionality of the data.

For large sample sizes, the memory and computational requirements would be substantial. Therefore, we recommend using the RFF to obtain the approximated MMD for large sample sizes, with a computational complexity of $O(D(n+m)d)$, where D is the number of Random Fourier Features. In practice, if $D \ll n$, the computational time and memory requirements would be significantly reduced.

The maximum mean discrepancy (MMD) was first introduced by Gretton et al. (2006), in which the authors introduced MMD as a measure to compare two distributions. It was then introduced in Gretton et al. (2012a), which includes detailed theoretical introduction and its statistical properties.

In reproducing kernel Hilbert space (RKHS), the computation of MMD can actually be performed directly via the kernel function without explicitly computing the embedding in the high-dimensional feature space. The essence of the kernel trick is to implicitly compute in high-dimensional feature spaces by defining the kernel function $k(x, y)$ without having to explicitly compute the feature map $\phi(x)$. This allows us to compute efficiently in high-dimensional or even infinite-dimensional feature spaces.

For two datasets of size n in d dimensions, computing the empirical MMD with a Gaussian kernel requires evaluating $O(n^2)$ pairwise distances, leading to an overall time complexity of $O(n^2 d)$. In our empirical analysis, the memory requirements for MMD computation pose significant constraints, particularly when processing high-dimensional samples. When dealing with samples of dimension $d = 100$, the computation becomes computationally intractable as n exceeds $10,000$ samples, primarily due to the quadratic memory complexity O(n^2) required for storing the kernel matrix.

Therefore, some approximation methods that can be used for fast computation and with lower complexity have been proposed.

For the Laplacian kernel, Bodenham and Kawahara (2023) presents an efficient computational method, `euMMD`, for the computation of MMD statistics on one-dimensional data, reducing the complexity from $O(n^2)$ to $O(n \log n)$. In the high-dimensional case, the authors further borrowed the idea of the slicing idea of the SWD to map the high-dimensional data to one-dimensional by random projection, which is utilized for the approximation computation so as to maintain the computational efficiency.

Random Fourier Features for MMD approximation

For the Gaussian kernel, a popular approach is to use the Random Fourier Features (RFF) to approximate the kernel calculation. The theoretical justification for RFF

comes from Bochner's theorem (Bochner, 1959), which can be summarized as follows:

Theorem 4.5.7. *Bochner's Theorem*

A continuous, shift-invariant kernel function $k(\delta) = k(x - y)$ is positive definite if and only if it is the Fourier transform of some non-negative finite Borel measure μ, i.e.:

$$k(\delta) = \int_{\mathbb{R}^d} e^{i\omega^\top \delta} \, d\mu(\omega),$$

where $\delta = x - y$, $\omega \in \mathbb{R}^d$, and i are imaginary units.

Bochner's theorem shows that any continuous, shift-invariant, positive definite kernel function can be expressed as the Fourier transform of a non-negative finite Borel measure. For a detailed proof of this, see Bochner (1959). Based on Bochner's theorem, we can represent the shift-invariant kernel function as a Fourier transform form. For example, for the Gaussian kernel function we have

$$k(x, y) = \exp\left(-\frac{\|x - y\|^2}{2\sigma^2}\right).$$

It can be expressed as:

$$k(\delta) = \int_{\mathbb{R}^d} e^{i\omega^\top \delta} p(\omega) \, d\omega,$$

where $p(\omega)$ is the Fourier transform of a Gaussian kernel $p(\omega) = \mathcal{N}(\omega; 0, 2\sigma^{-2}\mathbf{I})$ and let

$$p(\omega) = (2\pi\sigma^{-2})^{-d/2} \exp(-\|\omega\|^2(\sigma^2/2)).$$

RFF approximates the frequency $\{\omega_i\}$ by randomly sampling the frequency from the distribution $p(\omega)$ for ω, thus approximating the otherwise infinite-dimensional integral representation to a finite-dimensional inner product representation.

In order to convert the complex exponential representation to the cosine representation, we sample a shift $\{b_i\}$ from the uniform distribution $\mathcal{U}(0, 2\pi)$, which can be verified by:

$$\mathbb{E}_b[\cos(\omega^T x + b)\cos(\omega^T y + b)] = \frac{1}{2}\cos(\omega^T(x - y)).$$

By taking the real part and using Euler's formula $e^{i\theta} = \cos\theta + i\sin\theta$, we have:

$$k(x, y) = \int_{\mathbb{R}^d} 2\cos\left(\omega^\top x + b\right)\cos\left(\omega^\top y + b\right) p(\omega) \, d\omega.$$

Using the Monte Carlo integral, we have

$$k(x, y) \approx \frac{1}{D}\sum_{i=1}^{D} \cos(\omega_i^\top (x - y)).$$

Finally, define the mapping $z : \mathbb{R}^d \to \mathbb{R}^D$:

$$z(x) = \sqrt{\frac{2}{D}} \left[\cos(\omega_1^\top x + b_1), \cos(\omega_2^\top x + b_2), \ldots, \cos(\omega_D^\top x + b_D) \right]^\top .$$

With the above mapping, the kernel function can be approximated as:

$$k(x, y) = \langle z(x), z(y) \rangle,$$

where $\langle \cdot, \cdot \rangle$ denotes the Euclidean inner product. In this way, the computation of the Gaussian kernel is transformed from the explicit $\exp(\frac{\|x-y\|^2}{2\sigma^2})$ to the computation of the inner product of two D-dimensional feature vectors.

To demonstrate the effectiveness and computational efficiency of MMD and RFF MMD in distributional difference measures, we designed a small illustration comparison experiment.

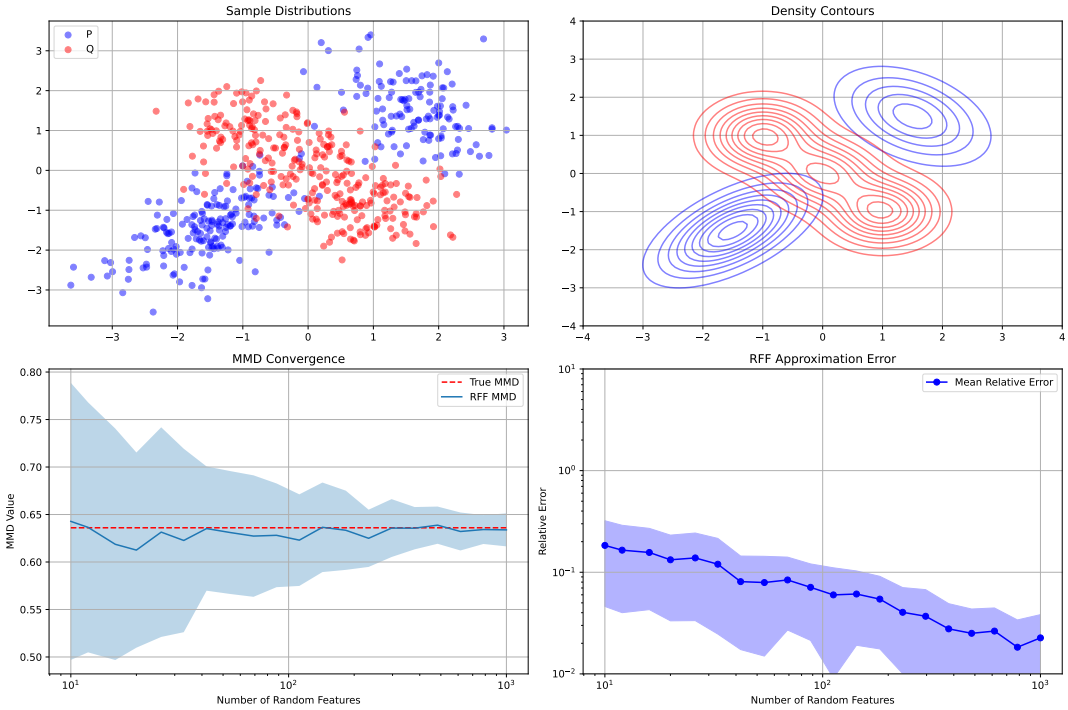


FIGURE 4.5: Illustration of the MMD and RFF-MMD comparison using two multi-modal Gaussian distributions.

As shown in the Figure 4.5, the experiments use different multi-modal distributions P and Q that have clear spatial separation characteristics in the two-dimensional plane. The results show that the RFF MMD estimates exhibit good convergence properties as the random feature dimension D is increased from 10 to 1000, with the relative error decreasing monotonically from 0.2 to 0.03. In particular, the confidence interval continues to narrow after $D > 100$, although the rate of error reduction tends to slow down, confirming that the RFF method is able to provide a reliable approximation of the MMD while maintaining computational efficiency.

4.6 Implementation Details

We have developed our benchmark suite in the Julia programming language, which combines high performance with user-friendly syntax. Julia uses a just-in-time compiler that ensures efficient execution of computationally intensive tasks such as MC sampling. Its syntax is similar to that of the Python programming language, featuring a script-like structure and dynamic typing, making it easy to learn and straightforward to use for scientific applications. Julia's rich ecosystem of packages for numerical calculations and statistical analysis makes it an ideal choice for implementing our benchmark suite.

The benchmark suite comes in the form of a Julia package and can be installed via the Julia package manager. It offers three main objects, corresponding to the three essential components needed for benchmarking: *test case*, *sampler*, and *metrics*. The implementation details of these components are discussed in the following.

The *Test case* objects describe the selected IID-sampleable target distributions. These objects are designed to be compatible with the `Distributions.jl` package (Besançon et al., 2021; Lin et al., 2019), which makes them easy to use as parts of *test cases*. Typically, the user will choose a test case from the list described in Section 4.4. But also custom test cases can be implemented using the `TestCase` datatype and linking to distributions or mixture model distributions from the `Distributions.jl` together with the chosen boundaries of the free parameters. For more complex or custom target functions that are not part of the `Distributions.jl` package, users can also define custom `Target` datatypes. In such a case, the user needs to implement the `Base.rand` and `Distributions.logpdf` functions for these types, which are needed for generating IID samples. Most of the target functions presented in Section 4.4 are implemented using the `Distributions.jl` package. The *Eight Schools* problem is implemented with a custom target function based on `posteriorDB` derived using `Stan` and `BridgeStan.jl` (Roualdes et al., 2023).

The *sampler* object consists of information about the selected sampling algorithm and its properties. Samplers can be added by defining new structs as subtypes of the `SamplingAlgorithm` datatype and providing a specific implementation of the `sample` functions. These types provide the functionality to define a `TestCase`, a sampler and a number of samples, and they return a set of samples as a `DensitySampleVector` object using the desired sampler.

As an alternative to sampling the distributions on the fly by directly interfacing to a sampling framework, our benchmark suite allows to read in pre-sampled data using the `FileBasedSampler` datatype and a user-defined path to either a directory or single file. The samples should be stored in a file format that can be read by the `FileBasedSampler` object, e.g. the CSV or HDF5 format. Samples can also be provided directly as a `DensitySampleVector` object. This allows the user to use custom formats

supported by any other Julia package, such as `JLD2.jl` [3] , and `HDF5.jl` [4].

The *metric* objects contain the result of the evaluated metric and its properties. Each of the metrics introduced in Section 4.5 is its own type and extends the abstract type `TestMetric`. The implementation of the metric requires a `calcmetric` function that takes a `DensitySampleVector` and returns a vector of metrics itself. In the case of two sample tests, the metric type is `TwoSampleTestMetric` and the function receives two `DensitySampleVector` instances. The evaluation of a metric on the sample batches is performed in parallel using the `Folds.jl` [5] package, which allows for efficient computations on multiple threads. The test statistics for each metric and test case can be built iteratively and are written into a `JSON` file for further analysis. The calculation of the Wasserstein Distance is based on the `R` implementation of the `transport` package (Schuhmacher et al., 2024). For the MMD, the `IPMeasures.jl` [6] package is used while the `Distances.jl` [7] package provides the calculation for the kernel width.

As an example, we can use the following code snippet to test the Metropolis-Hastings sampler of the `BAT.jl` package on a standard normal distribution in three dimensions which is provided as a test case. The `marginal_mean`, `marginal_variance` and `sliced_wasserstein_distance` metrics are used for the evaluation.

```
testcase = Standard_Normal_3D_Uncorrelated
metrics = [marginal_mean(),marginal_variance(),sliced_wasserstein_distance()]
sampler = BATMH()
```

Alternatively, the user can use the `FileBasedSampler` to read in pre-sampled data:

```
sampler = FileBasedSampler("samples.csv")
```

After defining the test case, the set of metrics used, and the sampler, the user can build the test statistic and plot the metrics using the following commands:

```
build_teststatistic(testcase, metrics, n=100, n_samples=10^5)
build_teststatistic(testcase, metrics, n=100, n_samples=10^5, s=sampler)
plot_metrics(testcase,metrics,BATMH())
```

The resulting plot will show the distribution of the metrics for the given test case and sampler compared to IID samples from the target distribution as will be demonstrated in the following section.

[3] https://GitHub.com/JuliaIO/JLD2.jl
[4] https://GitHub.com/JuliaIO/HDF5.jl
[5] https://GitHub.com/JuliaFolds/Folds.jl
[6] https://GitHub.com/JuliaFolds/IPMeasures.jl
[7] https://GitHub.com/JuliaFolds/Distances.jl

4.7 Walkthrough Example

We demonstrate the usage of the benchmark suite by evaluating the performance of the default Metropolis-Hastings sampler provided by the `BAT.jl` package. The evaluation is carried out on the basis of two test cases of varying complexity: A basic example using a standard normal distribution in three dimensions and a more complex, multimodal mixture model of two correlated normal distributions in three dimensions.

4.7.1 Basic Example: Standard Normal Distribution in 3D

The test case is a three-dimensional normal distribution without correlation. It is expressed using the `Distributions.jl` package as follows, which allows the benchmark suite to generate IID samples on the fly:

```
f = MvNormal(zeros(3), I(3))
bounds = NamedTupleDist(x = [-10..10 for i in 1:3])
Standard_Normal_3D_Uncorrelated = Testcases(f,bounds,3,"Normal-3D-Uncorrelated")
```

Then, the metrics and the sampler to be used for this test case are defined:

```
metrics = [marginal_mean(),marginal_variance(),
          sliced_wasserstein_distance(),maximum_mean_discrepancy()]
sampler = BATMH(n_steps=10^5, nchains=10)
```

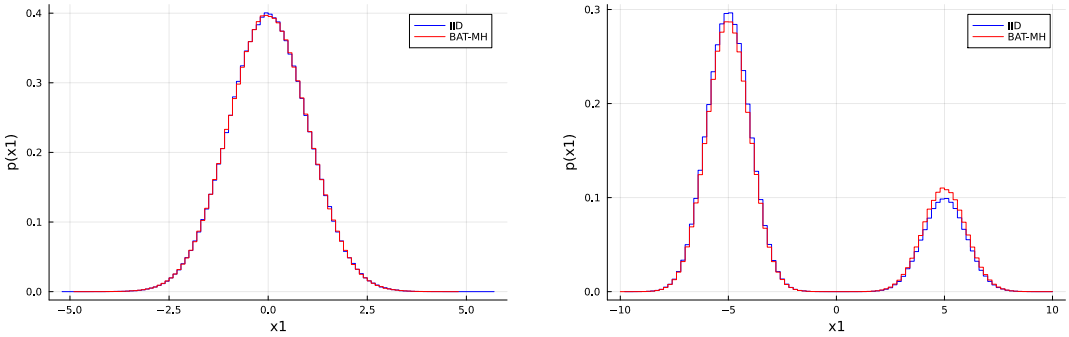
In this case, the sampler is directly interfaced to the `BAT.jl` package. Alternatively, when using a different sampling software package, it is possible to load the sample using the `FileBasedSampler` as follows :

```
sampler = FileBasedSampler("samples.csv")
```

After defining the test case, metrics, and sampler, the test statistic for the IID samples and the samples generated by the Metropolis-Hastings sampler are generated:

```
teststatistics_IID = build_teststatistic(Standard_Normal_3D_Uncorrelated, metrics,
                    n=100, n_steps=10^5, n_samples=10^5)
teststatistics_MH = build_teststatistic(Standard_Normal_3D_Uncorrelated, metrics,
                    n=100, n_steps=10^5, n_samples=10^5, s=sampler)
```

As an example, a one-dimensional marginalized distribution for the IID-samples (blue) and the samples using the Metropolis-Hastings algorithm (red) are shown in Figure 4.6a. The two distributions are visually indistinguishable. For the metrics, 100 batches of 10^5 samples each are used to evaluate the mean values and variances of all marginal distributions as well as the SWD and the MMD. Finally, an overview of the metrics is generated comparing the samples from the Metropolis-Hastings sampler with the IID samples:



(A) Uncorrelated 3D standard normal distribution.

(B) Mixture model with two 3D correlated normal distributions.

FIGURE 4.6: One-dimensional marginalized distributions for both test functions used in the examples using both IID sampling (blue) and Metropolis-Hastings (red).

```
plot_metrics(Standard_Normal_3D_Uncorrelated,metrics,sampler)
```

Figure 4.7 shows the expected mean values (black vertical line at 0) and standard deviations (colored regions for one, two, and three standard deviations) of each metric evaluated with the IID samples and compares them with the mean values (marker) and standard deviations (horizontal error bars) of the metrics evaluated with the samples provided by the Metropolis-Hasting sampler. In this concrete example, the metrics derived with the Metropolis-Hastings sampler roughly match the expectations in a sense that the mean values and standard deviations are close. A slight overestimation of the variance of the metrics is observed and is due to an overestimation of the effective sample size. The effective sample size is smaller than the actual sample size because of the autocorrelation of the samples that is intrinsically introduced by the Metropolis-Hastings algorithm.

For a closer look on the metrics, the distribution of the metrics for the samples generated by the Metropolis-Hastings sampler and the IID sampler can be plotted:

```
plot_teststatistic(Standard_Normal_3D_Uncorrelated,metrics[1],sampler,nbins=20)
plot_teststatistic(Standard_Normal_3D_Uncorrelated,metrics[3],sampler,nbins=20)
```

The distributions for the mean of the first dimension (left) and the SWD (right) are shown in Figure 4.8 and provide a detailed view of the distribution of the metrics for the Metropolis-Hastings sampler (red) compared to the IID samples (blue).

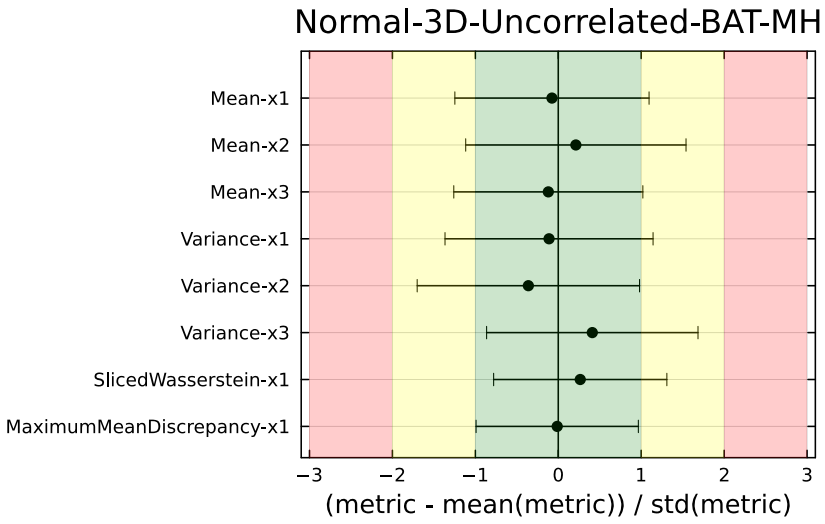
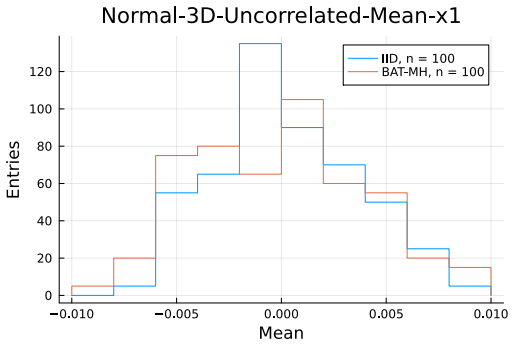
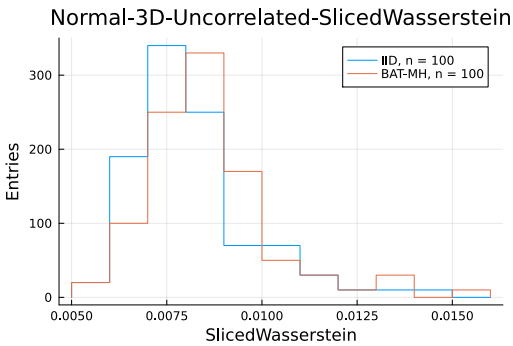


FIGURE 4.7: Mean values and standard deviations of several metrics calculated with IID samples (colored bands) and samples produced with the Metropolis-Hastings sampler implemented in BATjl. The samples are drawn from an uncorrelated standard normal distribution in three dimensions. The metrics are normalized to the mean and variance of the IID samples from the target distribution.



(a) Mean of the marginalized distribution of the first dimension.



(b) Sliced Wasserstein distance for the 3D distribution.

FIGURE 4.8: Example of the distributions of the metrics for the Metropolis-Hastings sampler and the IID samples for a standard normal distribution in 3D.

4.7.2 Complex Example: Mixture Model of Correlated Normal Distributions in 3D

The second test case is a mixture model of two correlated normal distributions in 3D. The definition of the test case can be expressed using the `Distributions.jl` package as follows:

```
r = 5
f1 = MvNormal(r*ones(10), ones(10,10)*0.9 + I(10)*0.1)
f2 = MvNormal(-r*ones(10), ones(10,10)*0.9 + I(10)*0.1)
f = MixtureModel([f1,f2], [0.25, 0.75])
bounds = NamedTupleDist(x = [-100..100 for i in 1:10])
normal_3d_multimodal_10std = Testcases(f,bounds,10,"Normal-3D-Multimodal-10std")
```

The generation of samples, the evaluation of the metrics, and the generation of the test statistics are performed analogously to the basic example. As an example, a one-dimensional marginalized distribution of the samples for both methods is shown in Figure 4.6b. While the samples might look similar at first glance the relative strength of the magnitude of the modes is significantly different. This effect is observed in all one-dimensional marginals. The resulting distribution of the metrics for the example are shown in Figure 4.9. The mean value and standard deviations of almost all metrics derived from samples obtained with the Metropolis-Hastings algorithm clearly deviate from those derived from the IID samples. This shows that the mixture model represents a challenge for the sampling algorithm compared to the basic example. This is to be expected, as the Metropolis-Hastings sampler is known to have difficulties sampling multimodal distributions, as it can get stuck in local modes. As a result, the sampler often does not meet the convergence criteria and therefore draws a set of samples that does not represent the target distribution. This example demonstrates the capability of the benchmark suite to evaluate the performance of samplers on a wide range of test cases with varying complexities.

4.8 Conclusion

In this paper, we have proposed a flexible and easy-to-use benchmark suite to evaluate the quality of MC sampling algorithms. The software package `MCbench` enables quantitative comparisons of samples drawn from well-defined multidimensional distributions by evaluating a variety of metrics. In addition to basic statistics, more sophisticated statistical measures such as the SWD and the MMD are used. The benchmark suite also features visualization and detailed information on the comparison of the samples.

In the future, we will increase the sensitivity of the benchmark suite by expanding the list of target functions and by increasing the number of metrics. In particular, we plan to include target functions with special features, such as the Rosenbrock function and the Himmelblau function, which serve as proxies for more complex scenarios and

multimodal functions. In addition, we plan to explore more sophisticated distance metrics to further improve the accuracy and scope of sample quality assessment. In particular, we will explore other MMD kernels to test their effectiveness and applicability in different sampling algorithms. These additions will complete our benchmark suite and help to qualitatively improve the development and optimization of MC sampling algorithms.

4.9 Algorithms

Algorithm 3 Calculating the sliced Wasserstein distance

Require: Two sample sets: $X = \{x_j\}_{j=1}^N \sim \mu$, $Y = \{y_j\}_{j=1}^N \sim \nu$; Number of projections L; Order of distance p

Ensure: Approximate sliced Wasserstein distance $\mathrm{SWD}_p(\mu, \nu)$

1: Initialize accumulation variable $S \leftarrow 0$

2: **for** $i = 1$ to L **do**

3: Randomly sample a unit vector from the unit sphere $\mathbb{S}^{d-1}$ θ_i

4: Calculate the projection:

$$\alpha_j = \theta_i^\top x_j, \quad \beta_j = \theta_i^\top y_j, \quad \text{for } j = 1, 2, \ldots, N$$

5: Sort $\{\alpha_j\}$ and $\{\beta_j\}$ in ascending order, respectively, to obtain the sorted sequences $\{\alpha_{(j)}\}$ and $\{\beta_{(j)}\}$

6: Compute the one-dimensional Wasserstein distance:

$$W^{(i)} = \left(\frac{1}{N} \sum_{j=1}^{N} \left| \alpha_{(j)} - \beta_{(j)} \right|^p \right)^{1/p}$$

7: Update accumulation variable $S \leftarrow S + \left(W^{(i)} \right)^p$

8: **end for**

9: Calculate SWD:

$$\mathrm{SWD}_p(\mu, \nu) = \left(\frac{S}{L} \right)^{1/p}$$

10: **return** $\mathrm{SWD}_p(\mu, \nu)$

Algorithm 4 Calculating maximum mean discrepancy using Gaussian Kernel

Require: Two sample sets: $\{x_i\}_{i=1}^n \sim p(x)$ and $\{y_j\}_{j=1}^m \sim q(y)$; Gaussian kernel bandwidth parameter σ

Ensure: Empirical MMD value MMD; median heuristic calculation of σ is also possible

1: **Step 1: Compute Kernel Matrices**

2: Compute the kernel matrix K_{XX} where $(K_{XX})_{i,j} = \exp\left(-\frac{\|x_i - x_j\|^2}{2\sigma^2}\right)$ for all $i, j \in \{1, \ldots, n\}$

3: Compute the kernel matrix K_{YY} where $(K_{YY})_{i,j} = \exp\left(-\frac{\|y_i - y_j\|^2}{2\sigma^2}\right)$ for all $i, j \in \{1, \ldots, m\}$

4: Compute the cross-kernel matrix K_{XY} where $(K_{XY})_{i,j} = \exp\left(-\frac{\|x_i - y_j\|^2}{2\sigma^2}\right)$ for all $i \in \{1, \ldots, n\}$ and $j \in \{1, \ldots, m\}$

5: **Step 2: Calculate MMD Value**

6: Calculate the MMD squared:

$$\text{MMD}^2 = \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}(K_{XX})_{i,j} + \frac{1}{m^2}\sum_{i=1}^{m}\sum_{j=1}^{m}(K_{YY})_{i,j} - \frac{2}{nm}\sum_{i=1}^{n}\sum_{j=1}^{m}(K_{XY})_{i,j}$$

7: **Step 3: Return Result**

8: **return** $\text{MMD} = \sqrt{\text{MMD}^2}$

Algorithm 5 Calculating maximum mean discrepancy using random Fourier features

Require: Two sample sets: $\{x_i\}_{i=1}^n \sim p(x)$, $\{y_j\}_{j=1}^m \sim q(y)$; feature dimension D; kernel function parameters (e.g., bandwidth of Gaussian kernel σ)

Ensure: approximate MMD value MMD_{RFF}

Step 1: Generate random frequencies and offsets

Sample D random frequency vectors $\{\omega_d\}_{d=1}^D$ independently from the spectral density $p(\omega)$ corresponding to the kernel function. For Gaussian kernel, $\omega_d \sim \mathcal{N}(0, \frac{1}{\sigma^2}\mathbf{I})$.

Sample D random offsets $\{b_d\}_{d=1}^D$ independently from a uniform distribution $\mathcal{U}(0, 2\pi)$

Step 2: Compute random Fourier feature mapping

for $i = 1$ to n **do**

For the ith sample x_i, compute its random Fourier feature vector:

$$z(x_i) = \sqrt{\frac{2}{D}}\left[\cos(\omega_1^\top x_i + b_1), \cos(\omega_2^\top x_i + b_2), \ldots, \cos(\omega_D^\top x_i + b_D)\right]$$

end for

for $j = 1$ to m **do**

For the jth sample y_j, compute its random Fourier feature vector:

$$z(y_j) = \sqrt{\frac{2}{D}}\left[\cos(\omega_1^\top y_j + b_1), \cos(\omega_2^\top y_j + b_2), \ldots, \cos(\omega_D^\top y_j + b_D)\right]$$

end for

Step 3: Calculate the mean embedding

Calculate the mean embedding of the two distributions:

$$\hat{\mu}_p = \frac{1}{n}\sum_{i=1}^n z(x_i), \quad \hat{\mu}_q = \frac{1}{m}\sum_{j=1}^m z(y_j)$$

Step 4: Calculate MMD value

Calculate the approximate MMD value:

$$\text{MMD}_{\text{RFF}} = \left\|\hat{\mu}_p - \hat{\mu}_q\right\|_2$$

return MMD_{RFF}

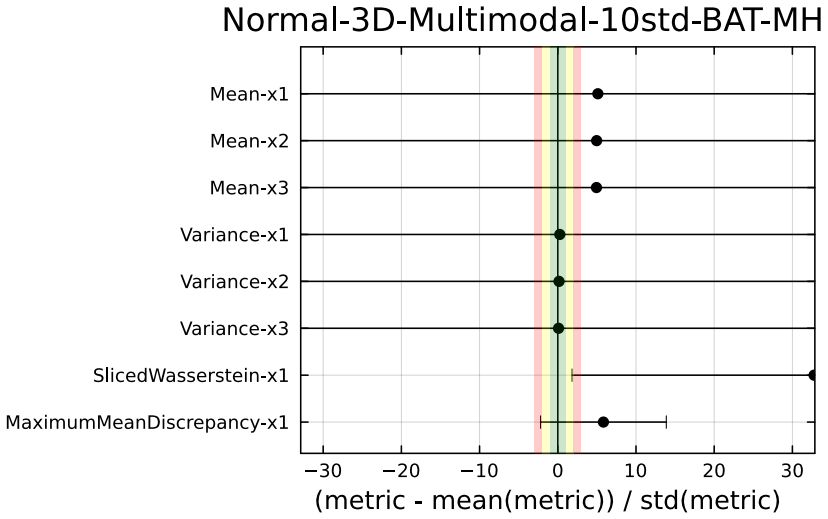


FIGURE 4.9: Example of metric distributions for the Metropolis-Hastings sampler on a mixture model of correlated normal distributions in 3D. The metrics are normalized to the mean and variance of the IID samples from the target distribution.

Chapter 5

Design and Implementation of Statistical Software Packages

In the field of modern statistics and data science, effective software implementations are a key bridge for translating theoretical innovations into practical applications. In this chapter, two statistical software packages developed within the framework of this dissertation's research will be described in detail: the `R` package `BayesPprobit` for Bayesian p-generalized probit regression with *coreset* construction, and the `Julia` package `MCbench` for Monte Carlo method evaluation. These two packages correspond to the main research directions of this thesis: flexible Bayesian modeling and scalable distributional metric computation, respectively.

5.1 R Package: Bayesian p-Probit Model

The `BayesPprobit` package bridges the gap between the theoretical flexibility of p-generalized probit models and the practical demands of big data analysis. While generalized link functions offer superior fit for asymmetric or heavy-tailed data (as shown in Chapter 2), their non-standard posterior distributions have historically hindered efficient computation. `BayesPprobit` solves this by integrating two key innovations: an adaptive Metropolis-within-Gibbs sampler for robust parameter exploration, and a native coreset construction engine that compresses large-scale datasets into weighted subsets, enabling Bayesian inference to scale to millions of observations with minimal accuracy loss.

5.1.1 Software Architecture Design

In designing the architecture of the `BayesPprobit` package, we followed the best practices of `R` language package development while focusing on modularity, extensibility and user-friendliness. We adopt a hierarchical component structure that allows the user to interact with the software at different levels of abstraction, from high-level model fitting functions to direct control of low-level MCMC samplers.

The architecture of the `BayesPprobit` package follows the standard package structure of `R` language, which mainly contains the following core components:

Core Estimation and Sampling Engine: The core modeling module is built around the `multi_chain()` function, providing an end-to-end workflow from data input to posterior sampling. To balance robustness with extensibility, this module employs a Strategy Pattern design (Gamma, 1995). This architecture decouples specific sampling steps (e.g., parameter updating, p-parameter tuning) into interchangeable strategies, allowing advanced users to customize priors or Metropolis-Hastings proposals without rewriting the framework. Internally, the module enforces rigorous validation to handle edge cases like perfectly separated or unbalanced data, while integrating convergence diagnostic tools that extend the `coda` package (Plummer et al., 2006).

Data Compression: A key innovation is the modularization of the data compression engine. Implementing the coreset construction methods from Chapter 3, this module generates high-quality summaries via sensitivity sampling. We designed this functionality as a stand-alone component with a generic interface, making it applicable to any statistical model relying on p-norm structures, not strictly limited to p-probit regression. It also handles essential preprocessing tasks, ensuring clean data ingestion for the subsequent modeling steps.

Object-Oriented Interface and Visualization: The package unifies these components through a consistent S3 class system, defining `pgprobit` and `pgcoreset` classes. Following `R` package development best practices (Wickham and Bryan, 2023), we provide specialized implementations of standard generics—such as `print`, `summary`, `plot`, and `predict`. This design ensures seamless interoperability with the `R` ecosystem while offering tailored visualization tools, particularly for inspecting the non-standard posterior distributions of the shape parameter p.

5.1.2 Core Functionality Implementation

The core functionality of the `BayesPprobit` package embodies several key innovations in modern Bayesian computation, particularly in the handling of large-scale data and flexible link functions.

1. **Multi-chain MCMC sampling**[1]:

   The center of the package is the multi-chain MCMC sampling system, implemented through the `multi_chain` function. Unlike traditional single-chain sampling methods, our implementation runs multiple independent Markov chains simultaneously, a design that provides significant advantages in several ways. First, the multi-chain structure allows us to quantify convergence using the Gelman-Rubin diagnostic statistic (Gelman and Rubin, 1992), providing a more reliable assessment of convergence than single-chain methods. Second, running multiple chains from overdispersed starting values ensures a more robust and comprehensive exploration of the parameter space. This is particularly valuable for inferring the shape parameter p, as it mitigates the risk of

[1] https://github.com/zeyudsai/BayesPprobit/blob/main/R/sampling.R

slow mixing and guarantees a more reliable characterization of the posterior variance. Third, the multi-chain design naturally supports parallel computation, permitting significant speedup of the sampling process on multi-core systems.

```
fit <- multi_chain(
n_sim = 5000, # number of iterations
burn_in = 1000, # warm-up period
X = X, # design matrix
y = y, # binary response variable
initial_theta = rep(0, d), # initial parameter values
initial_p = 2, # Initial p value
mh_iter = 100, # number of iterations per MH update
p_range = c(1, 3), # Range constraint for p
step_size = 0.05, # MH step size
n_chains = 3 # number of Markov chains
)
```

The internal implementation of this function uses a hybrid sampling strategy that combines the efficiency of Gibbs sampling with the flexibility of the Metropolis-Hastings method. For the regression coefficients β, we use data augmentation techniques to introduce latent variables such that the full conditional distribution of β has an analytic form given the latent variables. This approach avoids complex numerical integration while maintaining efficient sampling. For the shape parameter p, since its full conditional distribution does not have a standard form, we implement a Metropolis-Hastings update step designed to support adaptive tuning. The architecture allows the proposal distribution to be dynamically adjusted based on historical acceptance rates to target optimal efficiency, ensuring robust exploration of the shape parameter space across varying datasets.

2. **Data compression function**[2]:

   The data compression functionality is implemented through the `compute_coreset` function, which is an important innovation of the package that allows Bayesian inference to scale to large-scale datasets of millions of observations. The function implements the coreset construction methods described in detail in Chapter 3, including sensitivity sampling based on leverage scores, uniform sampling, and our proposed coreset (oneshot) method. The oneshot method is special in that it can be applied to a range of p parameters at once, rather than a single value, which is particularly important for our Bayesian framework since p itself is also a parameter that needs to be inferred.

```
coreset <- compute_coreset(
  X = X, # original data matrix
  y = y, # response variable
```

[2]https://github.com/zeyudsai/BayesPprobit/blob/main/R/coreset.R

```
  coreset_size = 100, # target coreset size
  method = ''oneshot'', # compression method
  p = 2 # p parameter, can be a range for the oneshot method.
)
```

The coreset construction process involves complex numerical linear algebra operations, including QR decomposition, eigenvalue computation, and efficient matrix sketching techniques. Our implementation leverages the sparse matrix capabilities of the `Matrix` package to ensure that computational efficiency is maintained even for high-dimensional data.

3. **Update of parameter p**[3]: The updating mechanism of the parameter p is central to the flexibility of the model and is implemented through the `update_p` function. This function employs the adaptive Metropolis-Hastings algorithm, in which the proposal distribution is adjusted according to the current state and historical acceptance rates. Unlike the standard MH algorithm with a fixed step size, our implementation monitors the acceptance rate and dynamically adjusts the offer variance to maintain a theoretically optimal acceptance rate (Gelman et al., 1997). This adaptive mechanism is particularly important for the p-parameter, since the shape and scale of its posterior distribution may vary substantially from one data set to another.

4. **Example and Diagnostic Functions**: In addition to these core computational functions, the package provides some diagnostic and visualization tools. The `run_example` function implements a complete workflow example, showing the entire process from data generation to model fitting to interpretation of the results, which is valuable for new users to understand how the package is used. The function generates synthetic data with known parameters, applies different coreset methods and compares the accuracy and computational efficiency of the results, providing the user with visual evidence of the effectiveness of the methods.

   The visualization function is implemented through the dedicated `plot` method, which generates multi-panel diagnostic charts for objects of class `pgprobit`. These plots include MCMC trajectory plots, density plots of posterior distributions, visualisations of Gelman-Rubin diagnostic statistics (Gelman and Rubin, 1992), and comparisons of true versus estimated parameters (in simulation studies with known true values). In particular, we developed specific tools for the visualisation of p-parameters, as understanding their posterior distribution is crucial for model interpretation.

[3] https://github.com/zeyudsai/BayesPprobit/blob/main/R/sampling.R

Performance Optimization

In order for the `BayesPprobit` package to efficiently handle large-scale datasets, several performance optimization strategies have been implemented:

1. **Coreset data compression**: With the previously described coreset method, we were able to reduce the amount of data to a fraction of its original size while maintaining the accuracy of statistical inference. This is the main strategy for dealing with large datasets.

2. **Vectorized computation**: The nature of the `R` language allows efficient vectorized computation. We avoid loop operations as much as possible, and instead use matrix and vector operations, which significantly improves computational efficiency.

3. **Sparse Matrix Support**: Through the integration of the `Matrix` package, we have implemented efficient matrix operations for high-dimensional sparse data. This is currently used for handle high-dimensional sparse data efficiently during the sketching and sensitivity scoring phases.

4. **Multi-chain Architecture**: The underlying `multi_chain` design is intrinsically parallelizable. While the current implementation executes chains sequentially for stability, the independence of chains allows for straightforward extension to parallel execution on multi-core systems.

Performance tests show that the processing time for large datasets ($n > 100,000$) can be reduced from hours to minutes after using the coreset method, achieving an approximate 100 times speedup while maintaining good statistical accuracy.

5.2 Julia Package: MCBench

The `MCBench` package[4] is a high-performance benchmarking suite designed to standardize the evaluation of Monte Carlo sampling algorithms. Written in `Julia`, it implements the rigorous distributional distance metrics established in Chapter 4, addressing the critical need for reproducible quality diagnostics in computational statistics.

5.2.1 Design Philosophy and Architecture

The package leverages `Julia`'s type system and multiple dispatch to achieve a highly modular architecture. As shown in the source structure (e.g., `src/MCBench.jl`), the design separates the *sampling source*, the *target distribution*, and the *evaluation metric* into distinct abstract types. This decoupling allows researchers to mix and match components—comparing any sampler against any target using any metric—without performance overhead.

[4] Available at https://github.com/tudo-physik-e4/MCBench.jl

Type System and Abstractions

The core architecture is built upon three abstract hierarchies defined in `src/testcases.jl`[5] and `src/samplers.jl`[6]:

1. **Test Case Abstraction (`AbstractTestcase`)**: Rather than hard-coding a static library of distributions, the package defines a flexible `Testcases` struct that wraps standard `Distributions.jl` (Lin et al., 2019) objects. This design allows users to instantly convert any continuous distribution—including complex mixtures or heavy-tailed distributions (e.g., Cauchy)—into a standardized benchmark problem with defined bounds and dimension information.

2. **Sampler Interface (`AnySampler`)**: To support language-agnostic benchmarking, we implemented the `FileBasedSampler` and `CsvBasedSampler`. These components allow the suite to ingest chain files generated by external probabilistic programming frameworks, such as `Stan` (Carpenter et al., 2017) or `PyMC`(Abril-Pla et al., 2023), facilitating cross-ecosystem comparisons. Additionally, an `IIDSampler` is provided to generate ground-truth baselines.

3. **Metric System (`TestMetric`)**: Rooted in the `TestMetric` abstract type, this layer implements both scalar summary statistics (e.g., marginal moments) and complex distributional distances. Crucially, the system is designed to handle **weighted samples** natively, ensuring correctness when evaluating algorithms like Importance Sampling or Nested Sampling.

5.2.2 Efficient Metric Implementation

A key contribution of the package is the optimized implementation of high-dimensional distance metrics.

Parallelized Sliced Wasserstein Distance (SWD): Computing the Wasserstein distance is cubic in complexity. To scale to millions of samples, we implemented the Sliced Wasserstein approximation. The implementation in `src/wasserstein.jl` utilizes `Julia`'s multi-threading capabilities (`Threads.@threads`) to parallelize the random projections. This design allows the computationally intensive projection steps to scale close to linearly with the number of available CPU cores.

```
# Simplified core logic from src/wasserstein.jl
function get_sliced_wasserstein_distance(
    sample1, sample2;
    L=1000, p=1, parallel=true
)
    # Projects high-dim samples onto random vectors z
    if parallel
        ds = Vector{Float64}(undef, L)
```

[5] https://github.com/tudo-physik-e4/MCBench.jl/blob/main/src/testcases.jl
[6] https://github.com/tudo-physik-e4/MCBench.jl/blob/main/src/samplers.jl

```
        Threads.@threads for i in 1:L
            z = get_random_projection(dim)

            # Project samples
            W = project_samples(sample1, z)
            V = project_samples(sample2, z)

            # 1D Wasserstein calculation
            ds[i] = wasserstein1d(V, W,
                                  wa=sample1.weight,
                                  wb=sample2.weight)
        end
    end
    return norm(ds, p) / (length(ds)^(1/p))
end
```

Adaptive Maximum Mean Discrepancy (MMD)

For the MMD metric (`src/mmd.jl`[7]), we implemented an adaptive bandwidth selection mechanism based on the median heuristic. The `compute_bandwidth` function automatically estimates the kernel width from the pairwise distance matrix of the combined samples. While users can manually specify the bandwidth parameter `g` if desired, the automatic selection provides a reasonable default that reduces the need for case-by-case tuning.

5.2.3 Workflow and Integration

The benchmarking workflow is orchestrated by the `build_teststatistic` function. This engine manages the generation of reference IID samples, the computation of effective sample sizes (ESS), and the iterative evaluation of metrics.

Integration Capability: The Eight Schools Example: To demonstrate the package's extensibility to non-standard distributions, we provide an implementation of the "Eight Schools" hierarchical model (Gelman et al., 2013). The example uses data formatted according to the `posteriorDB` (Magnusson et al., 2024) standard, allowing users to compare their implementations against our benchmark. By defining a custom `EightSchoolsAcceptReject` struct (in `examples/example_posteriordb.jl`), we show how the package can evaluate samplers on complex posterior geometries beyond standard distributional families. This confirms the software's capability to support realistic workflow scenarios where users can define custom targets by implementing the `Target` interface.

[7] https://github.com/tudo-physik-e4/MCBench.jl/blob/main/src/mmd.jl

5.3 Future Development

These two packages represent important attempts to translate statistical theory into practical tools. The flexibility of the `BayesPprobit` package makes it a tool when dealing with asymmetric or heavy-tailed data, while `MCbench` provides an unprecedented standardized framework for the evaluation of Monte Carlo methods. Based on user feedback and research progress, we have mapped out a comprehensive future development path.

For the `BayesPprobit` package, we plan to extend its functionality through three main directions. First, the flexibility of prior distributions will be enhanced to realize complex structures including sparse induced priors such as Horseshoe and Spike-and-slab, as well as pairwise weighted priors. These priors will enable the model to perform better in high-dimensional sparse scenarios, such as suggested by Piironen and Vehtari (2017b) and Piironen and Vehtari (2017a). Second, we will develop novel coreset construction methods, such as geodesic coreset algorithms as proposed by Campbell and Broderick (2018), and adaptive sampling algorithms optimized for specific data structures. These methods will further improve the efficiency and accuracy of data compression. Third, we will design a distributed computing architecture to support parallel MCMC sampling across multiple computing nodes. This architecture will utilize a parametric server model to enable the model to handle very large data sets with billions of data points while maintaining good a posteriori approximation quality.

The development of `MCBench` envisions enhancing its evaluation capabilities and application scope to broader statistical contexts. Regarding metric expansion, there is significant potential in extending the current library beyond standard Wasserstein and MMD implementations. It would be valuable to explore broader families of Integral Probability Metrics (IPMs) and divergence measures, particularly those specialized for time-series data or latent variable models where temporal dependencies are critical. These additions would provide a more comprehensive view of distributional discrepancies that simple marginal statistics might overlook.

In parallel, expanding the repository of test distributions remains a priority. We plan to incorporate more pathological benchmarks that stress-test sampling algorithms under extreme geometric conditions. A planned addition is the Rosenbrock function, whose narrow, curving valley structure poses well-known convergence challenges for many MCMC algorithms. Furthermore, including distributions derived from actual physical processes or complex hierarchical models would help bridge the gap between abstract benchmarking and realistic scientific applications.

Finally, improving interoperability with the wider data science ecosystem is a direction of great interest. While the current file-based interface allows for language-agnostic comparisons, developing direct bindings for `Python` and `R` environments would be a significant step forward. Such direct integration would allow users to call `MCBench` diagnostics natively within their existing pipelines—whether using `PyMC`,

`Stan`, or base `R`—thereby dramatically lowering the barrier to entry for practitioners who wish to rigorously validate their sampling results.

These development plans will further solidify the position of these two packages as important tools in the field of Bayesian analysis and Monte Carlo method evaluation, and are expected to advance the application of statistical methods in a wider range of disciplines. By continuously improving and extending these open-source tools, we aim to facilitate the bridge between statistical theory and practical applications, making advanced methods available to a wider range of researchers and practitioners.

5.4 Summary

This chapter has detailed two open source software packages developed for Bayesian statistics and distribution comparison: `BayesPprobit` and `MCbench`. These packages are not only practical products of theoretical research, but also important tools to facilitate the application of Bayesian methods in large-scale data analysis. With these tools, we hope to advance the development of flexible Bayesian modeling and efficient distribution distance computation in practical applications.

`BayesPprobit` implements flexible p-generalized probit regression modeling and efficient computation in big data scenarios through coreset technique. `MCbench`, on the other hand, provides a standardized framework for evaluating and comparing the performance of different Monte Carlo sampling algorithms. Together, these two packages form a complete tool chain for statistical analysis, providing comprehensive support for modern data analysis from model construction to performance evaluation.

The experience of developing these tools also reveals key principles of data science software development: a balance of flexibility, scalability, performance optimization, and user-friendliness. These principles have not only guided the design of current software packages, but will continue to guide future development efforts.

Chapter 6

Conclusions and Future Work

This dissertation is dedicated to addressing the core challenges facing Bayesian methods in the context of large-scale data, namely computational efficiency and model scalability. Through in-depth research on data reduction techniques and efficient computation of distributional metrics, this research not only makes advances at the theoretical level, but also translates these theoretical results into practical statistical software packages, which provide analytical tools for both academia and industry. This chapter will first summarize the main findings of this study, describe its theoretical contributions and practical significance, then discuss the limitations of the current study, and on the basis of this, look forward to possible future research directions.

6.1 Major Research Findings

Focusing on the core theme of scalable Bayesian methods in large-scale data environments, this thesis has made progress at three levels: theoretical innovation, method development, and practical applications. Through systematic research, we have successfully addressed the computational challenges of traditional Bayesian inference when facing large-scale data, while providing an efficient computational framework for distance measures between probability distributions.

In terms of model flexibility, our proposed Bayesian p-generalized probit regression model extends the applicability of traditional binary regression. The model, by introducing a shape parameter p, enables the link function to adaptively adjust to the tail characteristics of the data, ranging from the Laplace distribution ($p = 1$) to the standard normal distribution ($p = 2$) to more heavily or lightly tailed forms of the distribution. Empirical studies show that our model outperforms traditional logit and probit models on a wide range of assessment metrics when the data deviate from standard assumptions. In particular, when dealing with highly unbalanced binary classification problems, the adaptive nature of the model allows it to better capture the underlying structure of the data.

In terms of computational efficiency, the coreset method we developed provides a

theoretically guaranteed data compression technique for large-scale Bayesian inference. By combining Wasserstein distance theory and a sensitivity sampling framework, we demonstrate that the constructed coreset is able to reduce the data size to 1-5% of the original size while maintaining statistical accuracy. More importantly, our proposed oneshot coreset method is able to be applied to a range of p parameters simultaneously, which is important for the Bayesian framework where p itself needs to be inferred as well. Experimental results show that the coreset method is able to reduce the computation time from hours to minutes, achieving a nearly 100-fold speedup when dealing with datasets of more than $100,000$ observations.

In the field of semiparametric modeling, we successfully extend the coreset technique to MCTM, which is the first time that coreset data compression techniques are applied to complex semiparametric distribution models. By designing a hybrid sampling strategy that combines ℓ_2-leverage score sampling and convex hull approximation techniques, we effectively address the numerical instability of the logarithmic terms in MCTM. In a comprehensive evaluation of 14 different data generation processes, our method significantly outperforms uniform subsampling in 12 scenarios, especially when dealing with nonlinear, heteroskedastic, and multimodal structures.

In the area of distributional metric computation, we developed the `MCbench` benchmark suite to fill the gap in the field of Monte Carlo method evaluation. By implementing state-of-the-art metrics such as sliced Wasserstein distance, MMD, and its random Fourier feature approximation, the tool provides a standardized platform for evaluating the quality of sampling algorithms. In particular, our implementation takes into account the computational challenges of high-dimensional data, and algorithmic optimizations make it possible to work with millions of sample points and hundreds of dimensions of data.

6.2 Theoretical Contributions and Practical Implications

The theoretical contributions of this research are reflected in several cutting-edge areas of statistics and computational science. First, we provide a complete Bayesian theoretical framework for the p-generalized probit model, including the existence of the posterior distribution, the convergence analysis of the MCMC algorithm, and the proof of the identifiability of the parameter p. This theoretical framework not only extends the Bayesian theory of generalized linear models, but also provides methodological guidance for the Bayesian analysis of other nonstandard link functions.

In terms of data compression theory, the Wasserstein distance bounds we establish provide rigorous theoretical guarantees for Bayesian coreset. This result not only provides a theoretical foundation for the coreset construction of p-probit models, but also opens up a new research direction for data compression of other probabilistic models. More importantly, our oneshot coreset theory solves the technical difficulty of coreset construction on parameter ranges and provides a feasible solution for large-scale inference of adaptive parametric models.

For the MCTM model, our theoretical analysis reveals the intrinsic difficulty of coreset construction for semi-parametric models and proposes a solution based on geometric approximation. The established relative error bound proves that our hybrid sampling strategy is able to maintain the $(1 \pm \varepsilon)$ error approximation of the log-likelihood with high probability, while the lower bound results show that our sampling complexity is theoretically near-optimal.

In terms of distributional metric theory, our theoretical analysis of the sliced Wasserstein distance establishes its mathematical foundations as a true metric and provides precise bounds on the speed of convergence. These theoretical results not only support the design of our algorithm, but also provide a solid theoretical foundation for the comparison of high-dimensional distributions.

The practical significance of this research is reflected in several application areas. In industry, the large-scale binary classification problem is widely used in scenarios such as recommender systems, fraud detection, and medical diagnosis. Our approach makes Bayesian analysis, which would otherwise require huge computational resources, feasible while providing richer quantitative information about uncertainty than point estimation. In academic research, the open-source software package we developed can be adopted by further academic projects, advancing the application of related methods in different disciplines.

It is particularly worth emphasizing that the research paradigm established in this study, which focuses on both theory and practice, provides a valuable reference for statistical methodology research. By organically combining theoretical innovation, algorithm development and software implementation, we have not only advanced the academic frontiers, but also provided usable tools for practical applications, reflecting the trend of modern statistical research.

6.3 Future Research Directions

Despite the series of advances made in this study, there are still some limitations, and these limitations also point the way for future research. During the course of this thesis, we also conducted several additional literature reviews, which have inspired some of the potential future research directions discussed below.

Extension and improvement of Bayesian p-probit modeling

Current implementations of Bayesian p-probit models mainly use an uninformative uniform prior distribution, which is theoretically concise but may not be optimal in practical applications. Especially when dealing with the perfect separation problem in binary classification, the uninformative prior may lead to degradation of the posterior distribution. An important direction for future research is to develop families of informative prior distributions suitable for p-probit modeling.

We plan to focus on the application of several types of prior distributions. The first is the regularized prior, which consists of a hybrid penalty approach that combines ℓ_2 and ℓ_p regularization, such as suggested by Mandt et al. (2017), Li and Ma (2018), and Kneib et al. (2011). This approach can exploit both the stability of ridge regression and the sparsity-inducing properties of ℓ_p regularization, and is particularly effective for feature selection in high-dimensional data. Next is the horseshoe prior and its variants, which perform well in dealing with sparse regression problems and are able to efficiently shrink noisy features while maintaining important features, as suggested by Piironen and Vehtari (2017a) and Piironen and Vehtari (2017b).

Another important direction is to develop adaptive prior selection mechanisms. Different datasets may require different types of prior distributions, and manual selection often relies on expert knowledge and may not be objective enough. We plan to investigate data-driven prior selection methods, including empirical Bayesian methods and automatic prior adjustment techniques in a variational Bayesian framework, such as Kucukelbir et al. (2017) and Blei et al. (2017).

Extension and generalization of coreset techniques

Current coreset techniques in this thesis primarily focus on static datasets. A natural and significant direction for extension is towards dynamic data environments.

In the context of *online* or *streaming* settings where data arrives sequentially, Munteanu et al. (2022) have already demonstrated the effectiveness of combining online leverage score approximations with reservoir sampling for the standard probit model ($p = 2$). However, for general p-probit models, their approach still requires two passes over the data. The theoretical landscape has recently advanced significantly to address this limitation. Woodruff and Yasuda (2023) established the first nearly optimal subspace embeddings for online and sliding window models using Lewis weights for general ℓ_p norms. Incorporating these optimal theoretical bounds into the Bayesian framework is a promising direction to further tighten the sampling complexity and enable fully single-pass streaming coresets for arbitrary p-probit models.

More critically, future research should move beyond insertion-only streams to fully dynamic environments—formally known as the *turnstile* data stream model—where data points can be arbitrarily inserted, deleted, or updated (e.g., for data correction or cross-validation). As demonstrated by Munteanu and Omlor (2024b), handling such arbitrary updates for ℓ_p leverage score sampling is highly non-trivial. While maintaining exact ℓ_p sensitivities or Lewis weights under turnstile updates directly is computationally prohibitive, a promising solution emerges from the ℓ_2-augmentation strategy proposed by Munteanu and Omlor (2024a). They showed that augmenting ℓ_p sensitivities with ℓ_2 sensitivities yields optimal sampling complexity for $p \in [1, 2]$. Since ℓ_2 leverage scores are highly conducive to dynamic updates in turnstile streams via sketching techniques, integrating this augmentation approach paves the way for efficiently constructing dynamic coresets for Bayesian robust p-probit regression.

Recent theoretical advancements have sharpened the bounds for sensitivity sampling in the context of ℓ_p norms, which is intrinsic to the p-probit model. Munteanu and Omlor (2024a) proved that while standard ℓ_p sensitivity sampling may yield suboptimal dependency on the dimension d, augmenting the sampling distribution with ℓ_2 leverage scores achieves the optimal sampling complexity. This result directly benefits our p-probit coreset construction: by employing this hybrid sampling strategy (combining ℓ_p and ℓ_2 scores), we ensure that the size of our coreset remains linear in d, providing theoretical guarantee for data efficiency in this regime.

Furthermore, extending the coreset construction for MCTMs to ultra high-dimensional settings presents a highly promising avenue for future research. In our current approach, the convex hull calculation inherently exploits the block-diagonal structure of the model's likelihood, allowing the geometric approximation to be processed efficiently in batches across different dimensions. This decoupled structure naturally motivates a sparse approximation strategy. In ultra high-dimensional scenarios, it is reasonable to assume that the complex dependence structure is driven by a small subset of k active dimensions ($k \ll d$). By integrating compressed sensing and sparse regression techniques, future algorithms could identify and restrict the convex hull computations solely to these relevant dimensions. As recently demonstrated by Mai et al. (2023) in the context of sparse linear regression, sketching techniques can reduce the geometric and computational dependency on the full dimension d to a significantly smaller factor proportional to $k \log(d/k)$. Adapting these optimal sparse sketching bounds to our block-diagonal convex hull framework would allow us to bypass computations on irrelevant dimensions, thereby enabling ultra-scalable coreset constructions for high-dimensional MCTMs.

Structural Generalization of the MCTM Framework

While the current MCTM framework effectively combines non-parametric marginals with a Gaussian copula, there are still spaces for potential structural generalizations.

First, regarding the dependence structure, the current reliance on the Gaussian copula may limit the model's ability to capture tail dependencies in heavy-tailed data (e.g., financial returns). A promising future direction is to extend the coreset construction to other classes of copulas, such as t-copulas. Although these distributions do not possess the simple quadratic structure of the Gaussian, recent theoretical advances suggest that log-concave copulas with convex level sets can be approximated using John-Ellipsoids (Woodruff and Yasuda, 2022). This would allow us to enclose level sets within an ellipsoid and subsequently apply leverage score sampling induced by this surrogate quadratic form, albeit potentially at the cost of increased coreset dimensions.

Second, regarding the choice of basis functions, our current implementation relies on Bernstein polynomials to strictly enforce monotonicity in the transformation function. However, alternative basis families such as B-splines or Bayesian Conditional Transformation Models (BCTMs) (Carlan et al., 2024) could offer different approximation properties and local control. Future work will investigate the integration of

B-splines into the coreset framework. The challenge lies in efficiently enforcing monotonicity constraints within the sensitivity sampling framework, but this extension would significantly broaden the applicability of the method to scenarios requiring more flexible local adjustments.

Furthermore, addressing the geometric challenges in high-dimensional spaces remains a critical frontier. Our current method utilizes convex hull approximations to bound the logarithmic terms in the likelihood. As the dimension of the feature space increases, the complexity of exact convex hull computations grows exponentially, potentially becoming a bottleneck. Future research should explore approximate geometric methods to reduce dimensionality before hull construction. These techniques could extend the scalability of our coreset method to ultra-high-dimensional regimes while maintaining provable error bounds.

Integration of MCTM with generative modeling

MCTM has a deep theoretical connection with generative models, which have been developing rapidly in recent years, especially in relation to normalizing flows. We plan to study this connection in depth and develop a new modeling framework that combines the advantages of both. Specifically, the semi-parametric properties and interpretability of MCTM can be combined with the expressive power of normalizing flows to create generative models that are both flexible and interpretable.

In terms of generative modeling applications, we plan to investigate the use of MCTM in variational autoencoders (VAE) and GANs. Especially when dealing with high-dimensional complex data, the transform framework of MCTM may provide better potential spatial structures for these models.

Innovations and optimizations in distribution distance metrics

In the context of distribution distance metrics, we plan to develop a family of p-norm-based kernel functions and integrate them into the MMD framework. Such extensions are not only theoretically interesting but may also provide better performance in practical applications. In recent years, attempts to distribution distances metric based on kernel methods and applying them to various generative models have gained a lot of attention, such as Hagemann et al. (2023), Altekrüger et al. (2023), and Hertrich et al. (2023). We are particularly interested in the application of such kernel functions in models where kernel methods are heavily applied, such as Support Vector Machines (SVM) and GANs, where preliminary studies have shown that flexible kernels may outperform traditional Gaussian kernels in some cases. To address the efficiency of MMD computation with high-dimensional data, we plan to investigate more advanced approximation methods, including the Nyström method, and other variants based on random features and slicing methods. Theoretical analysis and practical performance comparison of these methods will be an important part of the research.

While Random Fourier Features provide a widely adopted and effective randomized approximation for scaling up Maximum Mean Discrepancy computations

by bypassing $O(n^2)$ pairwise kernel evaluations, incorporating alternative explicit feature maps presents a valuable direction for the future development of the `MCbench` framework. For instance, deterministic approximations based on the Taylor expansion of the Gaussian kernel can be utilized to construct explicit polynomial feature maps (Cotter et al., 2011). This deterministic approach eliminates the inherent variance of Monte Carlo-based methods like RFF, making it particularly advantageous when high-precision MMD estimations are required.

Furthermore, to accelerate the randomized dimensionality reduction process in high-dimensional settings, structured random projections leveraging the Fast Walsh-Hadamard Transform (FWHT) can be employed. As rigorously analyzed by Ailon and Liberty (2009), replacing dense Gaussian random matrices with Hadamard-based transformations effectively reduces the projection time complexity to $O(n \log d)$ using compositions of size-2 discrete Fourier transforms (DFTs) and Rademacher series on dual BCH codes. Integrating these deterministic expansions and fast-transform-based approximations into `MCbench` would provide researchers with a highly comprehensive suite of scalable MMD estimators, allowing for flexible trade-offs between computational speed, approximation variance, and memory efficiency.

Software Development and Maintenance Commitment

At a practical level, we are committed to continuing to maintain and improve the software packages that have been developed to ensure that they remain usable and state-of-the-art in an ever-changing computing environment. This includes regular bug fixes, performance optimizations, new feature additions, and compatibility maintenance with the latest `R` and `Julia` releases.

Through the exploration of these future research directions, it is expected to further promote the development of Bayesian methods under large-scale data and provide more powerful and reliable statistical tools and theoretical support for solving complex problems in the real world.

Appendix A

Mathematical Proofs

A.1 Main mathematical proofs of Ding et al. (2024)

A.1.1 Wasserstein bound

In this section, we aim to bound the Wasserstein distance between the original likelihood and the approximated likelihood obtained from the coreset. The mathematical framework and proof techniques in this appendix build upon the coreset theory for ℓ_p linear regression models developed in Munteanu (2018) (chapters 3), particularly the sensitivity sampling framework and Wasserstein distance bounds for Bayesian ℓ_p-regression.

Definition A.1.1 (ℓ_p Wasserstein distance, Villani, 2009)**.** *Given two probability measures γ, ν on $\mathbb{R}^d$ the ℓ_p Wasserstein distance between γ and ν is defined as*

$$\mathcal{W}_p(\gamma,\nu) = \left(\inf_{\lambda\in\Lambda(\gamma,\nu)} \int_{\mathbb{R}^d\times\mathbb{R}^d} \|x-y\|_p^p \ \mathrm{d}\lambda(x,y)\right)^{\frac{1}{p}} = \inf_{\lambda\in\Lambda(\gamma,\nu)} \mathbb{E}_\lambda[\|x-y\|_p^p]^{\frac{1}{p}},$$

where $\Lambda(\gamma,\nu)$ denotes the set of all joint probability distributions on $\mathbb{R}^d \times \mathbb{R}^d$ with marginals γ and ν, respectively

We split the distance into two parts that can be bounded separately.

Observation 1. Let $Z_1, Z_2 \in \mathbb{R}^d$ be random variables and let m_1, m_2 be values that they can attain. Let $Z_1^m = Z_1 - m_1$, respectively, $Z_2^m = Z_2 - m_2$ be their centered counterparts. Then for any fixed $p \in [1,\infty)$ it holds that

$$\mathbb{E}[\|Z_1 - Z_2\|_p^p] \leq 2^{p-1}\left(\|m_1 - m_2\|_p^p + \mathbb{E}[\|Z_1^m - Z_2^m]\|_p^p\right).$$

Proof. By convexity of the p-norm we can apply Jensen's inequality, which yields for arbitrary $x, y \in \mathbb{R}^d$ a generalized triangle inequality,

$$\|x+y\|_p^p = 2^p \left\|\frac{x+y}{2}\right\|_p^p \leq 2^p \frac{\|x\|_p^p + \|y\|_p^p}{2} = 2^{p-1}(\|x\|_p^p + \|y\|_p^p).$$

We thus have

$$\begin{aligned}\mathbb{E}\left[\|Z_1 - Z_2\|_p^p\right] &= \mathbb{E}\left[\|Z_1 - m_1 + m_1 - Z_2 + m_2 - m_2\|_p^p\right] \\ &= \mathbb{E}\left[\|Z_1^m - Z_2^m + m_1 - m_2\|_p^p\right] \\ &\leq 2^{p-1}\,\mathbb{E}\left[\|Z_1^m - Z_2^m\|_p^p + \|m_1 - m_2\|_p^p\right] \\ &= 2^{p-1}\left(\mathbb{E}\left[\|Z_1^m - Z_2^m\|_p^p\right] + \|m_1 - m_2\|_p^p\right).\end{aligned}$$

□

Next, we will need a p-norm generalization of the extreme singular values.

Definition A.1.2 (ℓ_p singular values, cf. Golub and Loan (2013) and Clarkson et al. (2016))**.** *Let $p \in [1, \infty)$. We define for a matrix $M \in \mathbb{R}^{n\times d}$*

$$\sigma_p^{\max}(M) = \sup_{x\in\mathbb{R}^d\setminus\{0\}} \frac{\|Mx\|_p}{\|x\|_p} \quad \textit{and} \quad \sigma_p^{\min}(M) = \inf_{x\in\mathbb{R}^d\setminus\{0\}} \frac{\|Mx\|_p}{\|x\|_p}.$$

We need a technical lemma, that bounds the distance of vectors, given that their negative log likelihood is close.

Lemma A.1.3. *Fix $p \in [1, \infty)$. Let $\alpha, \beta \in \mathbb{R}^d$ be such that $f(X\beta) \leq (1+\varepsilon)f(X\alpha)$. Then it holds that*

$$\|\alpha - \beta\|_p \leq \frac{(p\cdot(2+\varepsilon)(1+\mu))^{1/p}}{\sigma_p^{\min}(X)} \cdot f(X\alpha)^{1/p}.$$

Proof. By Lemma C.3 of Munteanu et al. (2022), we have for all $r \geq 0$ that $r^p \leq p \cdot f(r)$. This yields

$$\begin{aligned}\sigma_p^{\min}(X)\|\alpha - \beta\|_p &\leq \|X\alpha - X\beta\|_p \\ &\leq \|X\alpha\|_p + \|X\beta\|_p \\ &= (\|X\alpha\|_p^p)^{1/p} + (\|X\beta\|_p^p)^{1/p} \\ &\leq (1+\mu)^{1/p}((\|(X\alpha)^+\|_p^p)^{1/p} + (\|(X\beta)^+\|_p^p)^{1/p}) \\ &\leq (1+\mu)^{1/p}((p\cdot\textstyle\sum_{x_i\alpha>0} f(x_i\alpha))^{1/p} + (p\cdot\sum_{x_i\beta>0} f(x_i\beta))^{1/p}) \\ &\leq (1+\mu)^{1/p}((p\cdot f(X\alpha))^{1/p} + (p\cdot f(X\beta))^{1/p}) \\ &\leq (1+\mu)^{1/p}((p\cdot f(X\alpha))^{1/p} + (p\cdot(1+\varepsilon)f(X\alpha))^{1/p}) \\ &= (1+\mu)^{1/p}p^{1/p}(1+(1+\varepsilon)^{1/p})f(X\alpha)^{1/p} \\ &\leq (p(2+\varepsilon)(1+\mu))^{1/p}f(X\alpha)^{1/p}\end{aligned}$$

□

We apply the previous lemma with the respective MLE (modes) to bound the location component measured by their p-norm distance.

Corollary A.1.4. *Let X' be a coreset of X for the p-probit regression loss f and let μ, ν minimize $f(X'\eta), f(X\eta)$, respectively, over $\eta \in \mathbb{R}^d$. Then*

$$\|\nu - \mu\|_p \leq \frac{(p \cdot (2+\varepsilon)(1+\mu))^{1/p}}{\sigma_p^{\min}(X)} f(X\nu)^{1/p}.$$

Proof. By rescaling ε in Corollary 2.4 of Munteanu et al. (2022), we have that $f(X\mu) \leq (1+\varepsilon)f(X\nu)$. The result follows directly by invoking Lemma A.1.3 with $\alpha = \nu$ and $\beta = \mu$. □

Next we turn our focus to bounding the variation in p-norm of random variables drawn from the respective distributions, where by Observation 1 we can assume w.l.o.g. that they are centered at zero.

Lemma A.1.5. *Given $X \in \mathbb{R}^{n \times d}$, let X' be an ε-coreset for the p-probit or logistic loss (in which case the lemma applies with $p = 1$). Let $q \propto \exp(-f(X\alpha))$ and $q' \propto \exp(-f(X'\beta))$. Let Z_1^m, Z_2^m be the mode-centered versions of the random variables $Z_1 \sim q$ and $Z_2 \sim q'$ that are distributed according to q and q' respectively. Then we have*

$$\inf_{\lambda' \in \Lambda(q,q')} \mathbb{E}_{\lambda'} \|Z_1^m - Z_2^m\|_p^p \leq \frac{p \cdot (2+\varepsilon)(1+\mu)}{(\sigma_p^{\min}(X))^p} \mathbb{E}_q \left[f(XZ_1^m) \right].$$

Proof. We can upper bound the quantity by defining an arbitrary coupling λ between the distributions. We choose to construct an deterministic coupling, i.e., a one-to-one mapping between the two domains. To this end fix an arbitrary vector $\alpha \in \mathbb{R}^d$ and let $\beta \in \mathbb{R}^d$ be such that $\beta = \tau\alpha$, for $\tau \geq 0$ and $f(X\alpha) = f(X'\beta) := \rho$, for $\rho \geq 0$. The fact that the pair evaluates to the same value on the original data and on the coreset, respectively, means that they have the same density up to normalization. Moreover, since f is convex, the distributions have convex level sets, which means that in each direction (from the common mode 0) there is exactly one pair α and β with both properties. Now, coupling those pairs defines the the one-to-one mapping that we denote as λ in the remainder.

It remains to bound the distance of each pair of points α, β mapped to one another. To this end, we first show that their cost with respect to f are close to each other using the coreset property.

We have

$$f(X\beta) \leq (1+\varepsilon)f(X'\beta) = (1+\varepsilon)f(X\alpha).$$

Then we use this fact to invoke Lemma A.1.3 which yields the required bound

$$\|\alpha - \beta\|_p \leq \frac{(p \cdot (2+\varepsilon)(1+\mu))^{1/p}}{\sigma_p^{\min}(X)} \cdot f(X\alpha)^{1/p}.$$

We can thus conclude

$$\inf_{\lambda' \in \Lambda(q,q')} \mathbb{E}_{\lambda'} \|Z_1^m - Z_2^m\|_p^p \leq \mathbb{E}_{\lambda} \|Z_1^m - Z_2^m\|_p^p \\ \leq \frac{p \cdot (2+\varepsilon)(1+\mu)}{(\sigma_p^{\min}(X))^p} \mathbb{E}_{\lambda} \left[f(XZ_1^m) \right] = \frac{p \cdot (2+\varepsilon)(1+\mu)}{(\sigma_p^{\min}(X))^p} \mathbb{E}_{q} \left[f(XZ_1^m) \right].$$

□

We are now ready to prove the Wasserstein bound given in Theorem 3.2.3.

Proof of Theorem 3.2.3. The lemma follows from Observation 1, Corollary A.1.4 and Lemma A.1.5. Using the notation of those results, we have

$$\begin{aligned} \mathcal{W}_p(q,q') &:= \inf_{\lambda' \in \Lambda(q,q')} \mathbb{E}_{\lambda'} \left[\|Z_1 - Z_2\|_p^p \right]^{\frac{1}{p}} \\ &\leq 2^{1-\frac{1}{p}} \inf_{\lambda' \in \Lambda(q,q')} \mathbb{E}_{\lambda'} \left[\|m_1 - m_2\|_p^p + \|Z_1^m - Z_2^m\|_p^p \right]^{\frac{1}{p}} \\ &\leq 2 \left(\|\nu - \mu\|_p^p + \inf_{\lambda' \in \Lambda(q,q')} \mathbb{E}_{\lambda'} \left[\|Z_1^m - Z_2^m\|_p^p \right]^{\frac{1}{p}} \right) \\ &\leq 2 \cdot \frac{(p \cdot (2+\varepsilon)(1+\mu))^{\frac{1}{p}}}{\sigma_p^{\min}(X)} \left(f(X\nu) + \mathbb{E}_q \left[f(XZ_1^m) \right] \right)^{\frac{1}{p}}. \end{aligned}$$

□

We note that the last term $\mathbb{E}_q \left[f(XZ_1^m) \right]$ equals the entropy up to normalizing constants. By the maximum entropy property of the normal distribution among all distributions with the same variances, this term can be estimated to be of order $\operatorname{tr}((X^T X)^{-1})$. This holds up to constant factors depending on μ (to relate to the p-GGD) and further up to the constant $\tau(p) = p^{\frac{2}{p}} \frac{\Gamma(3/p)}{\Gamma(1/p)}$ that translates to the corresponding variance.

A.1.2 Details on one-shot coresets

We first need a technical result similar to Lemma 2 in Bachem et al. (2018) that allows us to interpolate between two consecutive values of p.

Lemma A.1.6. *For any $\theta \in [0,1]$, $\Delta \in \mathbb{R}_{>0}$ and $p \in [1,\infty)$ it holds that*

$$u_i^{(p(1+\theta\Delta))} \leq n^{\theta\Delta} \left((1-\theta) u_i^{(p)} + \theta u_i^{(p(1+\Delta))} \right)$$

Proof. Our goal will be to show that for any $\beta \in \mathbb{R}^d \setminus \{0\}$, thus in particular β maximizing $\frac{|x_i\beta|^{p(1+\theta\Delta)}}{\|X\beta\|_p^{p(1+\theta\Delta)}}$, it holds that

$$\begin{aligned}\frac{|x_i\beta|^{p(1+\theta\Delta)}}{\|X\beta\|_{p(1+\theta\Delta)}^{p(1+\theta\Delta)}} &\leq n^{\theta\Delta}\left((1-\theta)\frac{|x_i\beta|^p}{\|X\beta\|_p^p} + \theta\frac{|x_i\beta|^{p(1+\Delta)}}{\|X\beta\|_{p(1+\Delta)}^{p(1+\Delta)}}\right)\\ &\leq n^{\theta\Delta}\left((1-\theta)u_i^{(p)} + \theta u_i^{(p(1+\Delta))}\right).\end{aligned}$$

Here the last inequality follows by definition of the ℓ_p-leverage scores. Now fix β and set $z = X\beta$. Notice that we can assume without loss of generality that $p = 1$ as substituting z_i with $z_i' = z_i^p$ gives us the desired result for any p. Thus our goal for the remaining part of the proof is to show that

$$\frac{|x_i\beta|^{(1+\theta\Delta)}}{\|X\beta\|_{(1+\theta\Delta)}^{(1+\theta\Delta)}} \leq n^{\theta\Delta}\left((1-\theta)\frac{|x_i\beta|}{\|X\beta\|_1} + \theta\frac{|x_i\beta|^{(1+\Delta)}}{\|X\beta\|_{(1+\Delta)}^{(1+\Delta)}}\right).$$

Since $n \geq 1$ the lemma follows trivially for $\theta \in \{0,1\}$. Now for any positive real a, b, c, α, β with $\frac{1}{\alpha} + \frac{1}{\beta} = 1$. Young's inequality states that

$$abc \leq c\left(\frac{a^\alpha}{\alpha} + \frac{b^\beta}{\beta}\right)$$

We set $\alpha = \frac{1}{1-\theta}$ and $\beta = \frac{1}{\theta}$. Then $\frac{1}{\alpha} + \frac{1}{\beta} = 1 - \theta + \theta = 1$. Further we set $a = \left(\frac{|z_i|}{\|z\|_1}\right)^{1-\theta}$, $b = \left(\frac{|z_i|^{(1+\Delta)}}{\|z\|_{(1+\Delta)}^{(1+\Delta)}}\right)^\theta$ and $c = \frac{\|z\|_1^{(1-\theta)}\|z\|_{(1+\Delta)}^{\theta(1+\Delta)}}{\|z\|_{(1+\theta\Delta)}^{(1+\theta\Delta)}}$. Then we have that $abc = \frac{|z_i|^{(1+\theta\Delta)}}{\|z\|_{(1+\theta\Delta)}^{(1+\theta\Delta)}}$ and

$$\left(\frac{a^\alpha}{\alpha} + \frac{b^\beta}{\beta}\right) = (1-\theta)\left(\frac{|z_i|}{\|z\|_1}\right) + \theta\left(\frac{|z_i|^{(1+\Delta)}}{\|z\|_{(1+\Delta)}^{(1+\Delta)}}\right).$$

Now Hoelder's inequality implies that

$$\|z\|_{(1+\theta\Delta)}^{(1+\theta\Delta)} = \sum_{j=1}^n |z_j|^{(1+\theta\Delta)} \geq \frac{1}{n^{\theta\Delta}}\left(\sum_{j=1}^n |z_j|\right)^{(1+\theta\Delta)} = \frac{\|z\|_1^{(1+\theta\Delta)}}{n^{\theta\Delta}}.$$

We thus get that

$$c = \frac{\|z\|_1^{(1-\theta)}\|z\|_{(1+\Delta)}^{\theta(1+\Delta)}}{\|z\|_{(1+\theta\Delta)}^{(1+\theta\Delta)}} \leq n^{\theta\Delta} \cdot \frac{\|z\|_{(1+\Delta)}^{\theta(1+\Delta)}}{\|z\|_1^{\theta(1+\Delta)}} \leq n^{\theta\Delta}$$

as $\|z\|_1 \geq \|z\|_{(1+\Delta)}$. Combining everything we we conclude that

$$\frac{|z_i|^{(1+\theta\Delta)}}{\|z\|_{(1+\theta\Delta)}^{(1+\theta\Delta)}} = abc \leq c\left(\frac{a^\alpha}{\alpha} + \frac{b^\beta}{\beta}\right) \leq n^{\theta\Delta}\left((1-\theta)\left(\frac{|z_i|}{\|z\|_1}\right) + \theta\left(\frac{|z_i|^{(1+\Delta)}}{\|z\|_{(1+\Delta)}^{(1+\Delta)}}\right)\right).$$

□

Lemma 2.10 of (Munteanu et al., 2022) states that for any fixed p the sensitivity of x_i is bounded by $\zeta_i \leq c\mu\left(\frac{1}{n} + u_i^{(p)}\right)$. Thus it follows that:

Corollary A.1.7. *We set $\Delta = \frac{1}{\log(n)}$. There is a constant such that if $s_i \geq c\mu\left(\frac{1}{n} + u_i^{(p')}\right)$ for any $p' \in \{1, 1+\Delta, (1+\Delta)^2, \ldots (1+\Delta)^r\}$ then for any $p \in [1, (1+\Delta)^r]$ s_i is an upper bound for the sensitivity $\zeta_i = \max_{\beta\in\mathbb{R}} \frac{f_p(x_i\beta)}{f_p(X\beta)}$. In particular for any $p_m \in [1,\infty)$ if $r \geq \frac{\log(p_m)}{\log(1+\Delta)} = O(\log(p_m)\log(n))$ then for any $p \in [1, p_m]$ s_i is an upper bound for the sensitivity $\zeta_i = \max_{\beta\in\mathbb{R}^d\setminus\{0\}} \frac{f_p(x_i\beta)}{f_p(X\beta)}$.*

Proof of Theorem 3.2.5. We use Algorithm 6 but instead of approximating the ℓ_p-leverage scores of one single p (lines 2-8), we approximate the ℓ_p-leverage scores for all p in $\{p_{\min}, \left(1+\frac{1}{\log(n)}\right)p_{\min}, \left(1+\frac{1}{\log(n)}\right)^2 p_{\min}, \ldots, \left(1+\frac{1}{\log(n)}\right)^r p_{\min}\}$ where $r = \min\{r \in \mathbb{N} \mid (1+\frac{1}{\log(n)})^r p_{\min} \geq p_{\max}\} = O(\log(n)\cdot\log(p_{\max}/p_{\min}))$. Since we need r calculations in each step of the first pass the running time increases from $O(\text{nnz}(X))$ to $O(r\text{nnz}(X))$. Further the running time of (line 9) computing the QR-decompositions for all p takes time $O(r\text{poly}(d))$. Similarly the running time of the second pass is also increased by a factor of r since we need to calculate the ℓ_p leverage scores for *each* of the r values of p and sum them up to obtain the final sensitivity. The new sensitivities are also increased and add up to $S \leq r \cdot S_{p_{\max}}$. Thus we need to sample roughly r times as many rows as before. Finally the sample size of the sensitivity framework (Feldman and Langberg, 2011) has an additive $\log(1/\delta)$ term for achieving success probability $1-\delta$, which is typically set to some absolute constant. Here we choose $\delta = \frac{1}{100m}$. By applying a union bound, the probability that the coreset fails to approximate any of the m values of $p \in Q = \{p_1, \ldots, p_m\}$ is bounded by the constant $m\cdot\delta = 1/100$ which implies the $\log(m)$ term in our size bound. □

A.2 Main mathematical proofs of Ding et al. (2026)

A.2.1 Preliminaries

Considering the MCTM model whose negative log-likelihood function can be defined as in Equation (3.1), the goal of the coreset approach is to obtain a subset $C \subset D$ of data, such that the approximation of the original likelihood function is bounded within a factor $(1 \pm \epsilon)$.

After applying the basis functions a and their derivatives a' to the raw data, we can assume that for $(i,j) \in [n] \times [J]$ we are given data points $a_{ij} \in \mathbb{R}^d$ and $a'_{ij} \in \mathbb{R}^d$. We

use $A, A' \in \mathbb{R}^{nJ \times d}$ to denote the corresponding data matrices. Now for $i \in [n], j \in [J]$ consider $f(a_{ij}, \vartheta, \lambda) = \frac{1}{2}(\sum_{k=1}^{j} \lambda_{j,k}\vartheta_k a_{ij})^2 - \log(\vartheta_j a'_{ij})$ which is the negative logarithm of $g(i,j) = \vartheta_j a'_{ij} \exp(-\frac{1}{2}(\sum_{k=1}^{j} \lambda_{j,k}\vartheta_k a_{ij})^2)$. We assume that for all $i \in [n], j \in [J]$ and all choices of parameters it holds that $g(i,j) \leq c$ for some constant $c \in \mathbb{R}_{\geq 1}$. This is a natural Lipschitz-type restriction ensuring that the distribution function has a smooth transition from 0 to 1 preventing sudden jumps. Note, that this is equivalent to $-\ln(g(i,j)) \geq -\ln(c)$. We further add to the nJ loss contributions a total shift of $\log \mathcal{N} = nJ\ln(c) + n/J$. This corresponds to a normalization term $\mathcal{N}$ that ensures non-negativity and thus allows a relative approximation to be meaningful, but does not affect the optimization because it is independent of the parameters that we optimize. Additionally, we assume that there is an intercept, i.e., that for each (i,j) the first coordinate of both a_{ij} and a'_{ij} is 1 to make it consistent with the presence of intercepts in the transformation functions h defined above. In the following, we show that if we sample with probabilities that are larger than the ℓ_2 leverage scores plus a uniform term and add the convex hull of $\{a'_{ij} \mid i \in [n], j \in [J]\}$, then we get a coreset for $f(A, \vartheta, \lambda)$ to be minimized as in Equation (3.1). Let $w \in \mathbb{R}^{n \times J}$ be a weight vector. We split the weighted version of f into three parts (w_{ij} are omitted in the unweighted case):

$$\begin{aligned}
\text{squared part:} \quad & f_1(A, \vartheta, \lambda, w) = \frac{1}{2}\sum_{(i,j)\in[n]\times[J]} w_{ij}\left(\sum_{k=1}^{j} \lambda_{j,k}\langle \vartheta_k, a_{ij}\rangle\right)^2 \\
\text{positive log part:} \quad & f_2(A, \vartheta, \lambda, w) = \sum_{(i,j)\in[n]\times[J]} w_{ij} \max\{\log(\langle \vartheta_j, a'_{ij}\rangle), 0\} \\
\text{negative log part:} \quad & f_3(A, \vartheta, \lambda, w) = \sum_{(i,j)\in[n]\times[J]} w_{ij} \max\{-\log(\langle \vartheta_j, a'_{ij}\rangle), 0\}.
\end{aligned}$$

A.2.2 Squared part

We let $u_i = \sup_{\|x\|_2=1} \frac{|M_i x|^2}{\|Mx\|_2^2}$ for the i-th row of a matrix M denote their ℓ_2 leverage score. We show that sampling proportionally to the ℓ_2 leverage scores preserves the squared part f_1: given rows $a_{ij} \in \mathbb{R}^d$ where $i \in [n]$ and $j \in [J]$, we are looking for an ε-coreset, which is given by a matrix comprising a subset of rows indexed by $S \subseteq [n] \times [J]$ and corresponding weights $w_{i,j}$ for every $(i,j) \in S$, such that for all $\vartheta_1, \ldots, \vartheta_J \in \mathbb{R}^d$ and $\lambda \in \mathbb{R}^{J \times J}$ it holds that

$$\begin{aligned}
&\left|\sum_{i=1}^{n}\sum_{j=1}^{J}\left(\sum_{k=1}^{j} \lambda_{j,k}\langle \vartheta_k, a_{ij}\rangle\right)^2 - \sum_{(i,j)\in S} w_{i,j}\left(\sum_{k=1}^{j} \lambda_{j,k}\langle \vartheta_k, a_{ij}\rangle\right)^2\right| \\
&\qquad\qquad \leq \varepsilon\left|\sum_{i=1}^{n}\sum_{j=1}^{J}\left(\sum_{k=1}^{j} \lambda_{j,k}\langle \vartheta_k, a_{ij}\rangle\right)^2\right|
\end{aligned}$$

In the following, we arrange data points in a matrix such that sampling rows of this matrix yields a coreset for the function defined above. We set $B \in \mathbb{R}^{nJ \times dJ^2}$ to be the matrix whose rows equal $(b_{iJ+j})_k = a_{il}$ if $k = (j-1)J + l$ for some $l \in [J]$ and $(b_{iJ+j})_k = 0$ otherwise. Then B consists of n vertically stacked blocks. The i-th block,

for $i \in [n]$, is defined by

$$B_i = \begin{bmatrix} b_i & 0 & 0 & 0 & \cdots & 0 \\ 0 & b_i & 0 & 0 & \cdots & 0 \\ 0 & 0 & b_i & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \cdots & 0 \\ 0 & 0 & 0 & 0 & \cdots & b_i \end{bmatrix} \in \mathbb{R}^{J \times dJ^2},$$

where $b_i = (b_{i1}, b_{i2}, \ldots, b_{iJ})$. The idea is that for any possible parametrization, the squared part can be represented by a product of the new matrix B with the vector $\theta = (\vartheta^T, \lambda^T)^T$. An ε-subspace embedding for ℓ_2 is therefore sufficient to approximate the squared part, i.e., it yields that $\forall \theta \colon \|B'\theta\|_2^2 = (1 \pm \varepsilon)\|B\theta\|_2^2$, where B' consists of few weighted rows subsampled from B according to their ℓ_2 leverage scores Woodruff, 2014.

Lemma 3.3.1. *There is a coreset S for f_1 of size $O(J^2 d/\varepsilon^2)$ which can be computed in time $O(\mathrm{nnz}(B)\log(nJ) + \mathrm{poly}(dJ))$ and with high probability by sampling and reweighting according to the ℓ_2 leverage scores of B, where $\mathrm{nnz}(B)$ denotes the number of non-zero entries of B. We then have $|f_1(A, \vartheta, \lambda) - f_1(A(S), \vartheta, \lambda, w)| \leq \varepsilon f_1(A, \vartheta, \lambda)$, where $A(S)$ denotes the restriction to S.*

Proof. We apply ℓ_2 leverage score sampling to B which gives us a set $S \subseteq [n] \times [J]$ and weights $w_{ij} \in \mathbb{R}_{\geq 1}$ for all $(i,j) \in S$ such that $|S| = O(J^2 d/\varepsilon^2)$ and with high probability for all $x \in \mathbb{R}^{dJ^2}$ it holds that

$$\left| \|Bx\|_2^2 - \sum_{(i,j)\in S} w_{i,j} \|b_{i,j} x\|_2^2 \right| \leq \varepsilon \|Bx\|_2^2.$$

Assume that the statement above holds and consider any parameterization $\vartheta_1, \ldots \vartheta_J \in \mathbb{R}^d$ and $\lambda \in \mathbb{R}^{J \times J}$. We define $x \in \mathbb{R}^{dJ^2}$ by $x_{jk} = \lambda_{j,k}\vartheta_k$. Then we have that

$$\begin{aligned} &\left| \sum_{i=1}^{n} \sum_{j=1}^{J} \left(\sum_{k=1}^{j} \lambda_{j,k} \langle \vartheta_k, a_{ij} \rangle \right)^2 \vartheta_k - \sum_{(i,j)\in S} w_{i,j} \left(\sum_{k=1}^{j} \lambda_{j,k} \langle \vartheta_k, a_{ij} \rangle \right)^2 \right| \\ &= \left| \|Bx\|_2^2 - \sum_{(i,j)\in S} w_{i,j} \|b_{i,j} x\|_2^2 \right| \leq \varepsilon \|Bx\|_2^2 = \varepsilon \left| \sum_{i=1}^{n} \sum_{j=1}^{J} \left(\sum_{k=1}^{j} \lambda_{j,j} \langle \vartheta_k, a_{ij} \rangle \right)^2 \right|. \end{aligned}$$

□

A.2.3 Logarithmic parts

We now turn our attention to the positive logarithmic part f_2. This is going to be handled using the sensitivity framework, which generalizes the previous leverage score sampling for the ℓ_2 norm to more general families of functions, we refer to Appendix A.2.6 for details. First of all, we bound the VC dimension of the logarithmic

function. This is done in a standard way. Using strict monotonicity, the logarithmic function of the inner product can be inverted (respecting their domain and range), relating it to linear classifiers in d dimensions. The latter have a known VC dimension of $d+1$ using classic learning theory results Kearns and Vazirani, 1994. The second part is bounding the total sensitivity, which is the sum of all sensitivity scores $s_i = \sup_{\vartheta,\lambda} \frac{f_2(a_{ij},\vartheta,\lambda)}{\sum f_2(a_{ij},\vartheta,\lambda)}$ of single data point contributions. To bound this value, we leverage that for all parameters λ and ϑ it holds that $\ln(\vartheta_j a'_{ij}) - \frac{1}{2}(w_{i,j}\sum_{k=1}^{j}\lambda_{j,k}\vartheta_k a_{ij})^2 \leq \ln(c)$, which allows us to relate the contribution of the logarithmic part to the squared part up to a constant $\gamma > 1$ such that $s_i \leq \gamma(u_i + 1/n)$. This allows to reuse the ℓ_2 leverage scores for this part as well, albeit with an additional uniform component, and with an increase of the sample size, which comes from incorporating the VC dimension and the Lipschitz bound c. We also note that the ε-error is relative to f_1 instead of f_2 directly.

Lemma 3.3.2. *Assume that S is a sample consisting of $O(J^2d^2\ln(cdJ)c^6/\varepsilon^2)$ points drawn with probability $p_i \geq \alpha(u_i + 1/n)$, where $\alpha \in O(J^2d\ln(cdJ)c^6/\varepsilon^2)$. Then with high probability it holds that $|f_2(A,\vartheta,\lambda) - f_2(A(S),\vartheta,\lambda,w)| \leq \varepsilon f_1(A,\vartheta,\lambda)$, where $A(S)$ denotes the restriction to S.*

Proof. Our goal is to apply the results of sensitivity sampling. For $(i,j) \in [n]\times[J]$ we define $h_{i,j}(\vartheta,\lambda) = \max\{-\ln(\vartheta_j a'_{ij}), 0\}$ and $h_0(\vartheta,\lambda) = f_1(A,\vartheta,\lambda)$. We set $h(A,\vartheta,\lambda,w) = h_0(\vartheta,\lambda) + \sum_{(i,j)\in[n]\times[J]} h_{i,j}(\vartheta,\lambda)$. It holds that the VC-dimension of $\{h_{i,j} \mid (i,j) \in [n]\times[J]\}\cup\{h_0\}$ is bounded by $d+2$ since $\log(\cdot)$ is a monotonic function and the VC dimension of $\{h_x : \vartheta \mapsto x\vartheta \mid x \in \mathbb{R}^d\}$ is bounded by $d+1$ Kearns and Vazirani, 1994 and adding the function h_0 can only increase the VC-dimension by 1.

For the sensitivity we observe the following: since for all parameters λ and ϑ it holds that

$$\ln(\vartheta_j a'_{ij}) - \frac{1}{2}\left(\sum_{k=1}^{j}\lambda_{j,k}\vartheta_k a_{ij}\right)^2 \leq \ln(c)$$

it in particular holds that

$$\ln(\vartheta_j a'_{ij}) - \frac{1}{2}\left(\vartheta_j a_{ij}\right)^2 \leq \ln(c)$$

Note that there is some $b \in \mathbb{R}$ such that $\vartheta_j a'_{ij} = b\vartheta_j a_{ij}$. Since we can scale ϑ_j arbitrarily we get that

$$\ln(bt) - \frac{1}{2}(t)^2 \leq \ln(c)$$

holds for any $t \in \mathbb{R}$. Note, that the term on the left hand side is maximized if $t = 1$ and thus

$$\ln(b) - \frac{1}{2}(1)^2 \leq \ln(c)$$

or equivalently $b \leq ec$. We further have that the derivative of

$$\frac{\ln(bt)}{\frac{1}{2}(t)^2}$$

is given by $(t/2 - \ln(bt)t)/(\frac{1}{2}(t)^2)^2$ which is 0 if $t = 0$ or $\ln(bt) = 1/2$ or equivalently $t = \exp(\frac{1}{2})/b$ which implies that

$$\frac{\ln(bt)}{\frac{1}{2}(t)^2} \leq \frac{\ln(\exp(\frac{1}{2}))}{(e/2)(1/b)^2} = b^2/e.$$

We thus have that $s_{(i,j)} = (ec^2)u_{i,j}$ is an upper bound for the sensitivity of $h_{i,j}$. Consequently the total sensitivity is bounded by $(ec^2)dJ^2$.

Lastly we have that $h(A, \vartheta, \lambda, w) \leq (ec^2 + 1)f_1(A, \vartheta, \lambda, w)$. Thus applying sensitivity sampling with error parameter ε/c^2 gives us the desired result. □

Next, we handle the remaining negative logarithmic part given by f_3. We note that the asymptote at 0 precludes finite bounds on the sensitivity. We handle this similarly to Lie and Munteanu, 2024 by restricting the optimization space to $D(\eta) = \{(\vartheta, \lambda) \mid \forall (i,j) \in [n] \times [J] : \langle \vartheta_j, a'_{ij} \rangle > \eta\}$ comprising only solutions for which the inner product is bounded away by $\eta \geq 0$ from zero. Setting $\eta = 0$ corresponds to the original domain.[1] By avoiding high sensitivity points in this way, f_3 can be bounded in terms of uniform sensitivities together with the VC dimension bound $d + 1$ and approximated by invoking the sensitivity framework again.

Lemma 3.3.3. *Let (ϑ^*, λ^*) be an optimal solution. Then there exist $(\vartheta, \lambda) \in D(\eta)$ with $|f(A, \vartheta, \lambda) - f(A, \vartheta^*, \lambda^*)| \leq 2J\eta f_1(A, \vartheta^*, \lambda^*) + \eta n + J^2\eta^2$. Further, if S is a sample consisting of points drawn with probability $p_i \geq \alpha(1/n)$ where $\alpha \in O(d \ln(cdJ)/\eta^2)$ combined with all points that are in the convex hull of $\{a'_{ij} \mid i \in [n], j \in [J]\}$. Then with high probability, it holds for all $(\vartheta, \lambda) \in D(\eta)$ that $|f_3(A, \vartheta, \lambda) - f_3(A(S), \vartheta, \lambda, w)| \leq \eta f_1(A, \vartheta, \lambda) + \eta n$.*

Proof. Let $e_1 \in \mathbb{R}^d$ be the first unit vector and let ϑ', λ be a feasible solution, i.e. $\vartheta'_j a'_{ij} > 0$. Then we claim that (ϑ, λ) with $\vartheta = \vartheta' + \eta e_1$ fulfills $f(A, \vartheta, \lambda) \leq f(A, \vartheta', \lambda) + \eta f_1(A, \vartheta', \lambda) + \eta$. Applying this to (ϑ^*, λ^*) proves the first part of the lemma. First note that as $\vartheta'_j a'_{ij} > 0$ we have that $(\vartheta, \lambda) \in D(\eta)$. Further we have that

$$f_3(A, \vartheta, \lambda) - f_2(A, \vartheta, \lambda) \leq f_3(A, \vartheta', \lambda) - f_2(A, \vartheta', \lambda)$$

as $\vartheta_j a'_{ij} \geq \vartheta'_j a'_{ij}$. Lastly note that

$$2f_1(A, \vartheta, \lambda) = \sum_{(i,j) \in [n] \times [J]} \left(\sum_{k=1}^{j} \lambda_{j,k} \vartheta_k a_{ij} \right)^2$$

[1] We also note that the final choice will be $\eta = \Theta(\varepsilon)$ and negative value correction into the positive domain is common practice and was implemented in (Klein et al., 2022) before our theoretical investigations.

$$
\begin{aligned}
&= \sum_{(i,j)\in[n]\times[J]} \left(\sum_{k=1}^{j} \lambda_{j,k}(\vartheta'_k + e_1)a_{ij} \right)^2 \\
&\leq \sum_{(i,j)\in[n]\times[J]} \left(\sum_{k=1}^{j} \lambda_{j,k}\vartheta'_k a_{ij} \right)^2 + 2J\eta \max\{(\sum_{k=1}^{j} \lambda_{j,k}\vartheta'_k a_{ij}), 1\} + J\eta^2 \\
&\leq \sum_{(i,j)\in[n]\times[J]} \left(\sum_{k=1}^{j} \lambda_{j,k}\vartheta'_k a_{ij} \right)^2 + 2J\eta \max\{(\sum_{k=1}^{j} \lambda_{j,k}\vartheta'_k a_{ij})^2, 1\} + J\eta^2 \\
&= (2 + 4J\eta) f_1(f(A, \vartheta', \lambda) + 2J\eta n + J^2\eta^2 n.
\end{aligned}
$$

Next we show by sampling techniques that with high probability it holds that $|f_3(A, \vartheta, \lambda) - f_3(A(S), \vartheta, \lambda, w)| \leq \eta f_1(A, \vartheta, \lambda) + \eta n$. Again we use sensitivity sampling. The bound of the VC dimension is again $d + 2$. For the sensitivity we have that $-\ln(\vartheta_j a'_{ij}) \leq \eta^{-1}$. Since we only need an absolute error of ηn we can apply the results of sensitivity sampling. □

A.2.4 Main result

We get the following theorem by a union bound over Lemmas 3.3.1 to 3.3.3 and putting their error bounds together using the triangle inequality. The additive errors of Lemma 3.3.3 are further charged against the optimal cost using the normalization shift given by $\log \mathcal{N}$, see above.

Theorem 3.3.4. *Assume that S is a sample consisting of $O(J^2d^2 \ln(cdJ)c^6/\varepsilon^2)$ points drawn with probability $p_i = \alpha(u_i + 1/n)$, where $\alpha \in O(J^2d \ln(cdJ)c^6/\varepsilon^2)$ combined with all extreme points $(i, j) \in [n] \times [J]$ that are on the convex hull of $\{a'_{ij} \mid i \in [n], j \in [J]\}$. Then with high probability it holds for any $(\vartheta, \lambda) \in D(\eta)$ that $|f(A, \vartheta, \lambda) - f(A(S), \vartheta, \lambda, w)| \leq \varepsilon f(A)$.*

Proof. By Lemma 3.3.1 we have that $|f_1(A, \vartheta, \lambda) - f_1(A(S), \vartheta, \lambda, w)| \leq \varepsilon f_1(A, \vartheta, \lambda)$ with high probability.

By Lemma 3.3.2 we have that $|f_2(A, \vartheta, \lambda) - f_2(A(S), \vartheta, \lambda, w)| \leq \varepsilon f_1(A, \vartheta, \lambda)$ with high probability.

By Lemma 3.3.3 we have that $|f_3(A, \vartheta, \lambda) - f_3(A(S), \vartheta, \lambda, w)| \leq \eta f_1(A, \vartheta, \lambda) + \eta n$ with high probability.

Overall, by the triangle inequality, we have that $|f(A, \vartheta, \lambda) - f(A(S), \vartheta, \lambda, w)| \leq 2\varepsilon f_1(A, \vartheta, \lambda) + \eta f_1(A, \vartheta, \lambda) + \eta n$. Substituting ε/J for η and using that $f(A, \vartheta, \lambda) \geq n/J$, and $f \geq f_1$ yields the desired bound. □

We remark that the convex hull can comprise $\Omega(nJ)$ points, however, different η-kernel coresets of size $\Theta(1/\eta^{(d-1)/2})$ exist for the problem (Agarwal et al., 2004; Chan, 2004), surveyed in (Agarwal et al., 2005), in the field of computational geometry. These η-kernels also match the requirements of the shifted domain $D(\eta)$. We discuss our particular choice (Blum et al., 2019) in the experimental section.

A.2.5 Lower bounds

It suffices to prove a lower bound for the squared part. In particular the bound will hold against preserving the subspace (equivalently the rank) spanned by the data. In the following, we give two lower bounds under different natural assumptions on the λ coefficients that define the dependence structure of the Gaussian copula. The first one is a weaker lower bound that holds without assumptions on λ other than its lower triangular structure. The second lower bound is stronger and holds under additional but common assumptions on λ.

Lemma 3.3.5. *There exists a dataset $\{a_{ij}\}_{i\in[n]j\in[J]}$ such that any coreset for MCTMs with $\varepsilon < 1$ has size at least $\Omega(dJ^2)$ even if $\lambda_{ij} = 0$ for all $i, j \in [n]$ with $j > i$ and $|\lambda_{ij}| \leq 1$ for all $i, j \in [J]$.*

Proof. Consider the instance with $n = (J-1)^2 d$ consisting of Jd blocks $\{a_{ij}\}_{tj\in[J]}$ where $t \in J \times d$. More precisely for $t = (j_0, k) \in J \times J \times d$ we have that

$$a_{tj} = \begin{cases} e_k, & \text{if } j \geq j_0\} \\ 0 & \text{else} \end{cases}$$

We set $A \in \mathbb{R}^{j^2 d \times J^2 d}$ to be the matrix with parameters that represents these rows. For $t = (3, k)$ the t-th block $A_t \in \mathbb{R}^{J \times Jd}$ of the matrix A looks as follows:

$$A_t = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & e_k & 0 & 0 & \cdots & 0 \\ 0 & 0 & e_k & e_k & 0 & \cdots & 0 \\ 0 & 0 & e_k & e_k & e_k & \vdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & 0 \\ 0 & 0 & e_k & e_k & e_k & \cdots & e_k \end{bmatrix}$$

and the matrix itself has the form

$$A = \begin{bmatrix} A_{1,1} \\ A_{1,2} \\ \vdots \\ A_{1,J} \\ A_{2,1} \\ \vdots \\ A_{d,J} \end{bmatrix}$$

Note that the matrix $A \in \mathbb{R}^{J^2 \times d}$ with rows (a_{ij}) is of rank at least $dJ(J-1)$ even if we restrict the parameter space by requiring $\lambda_{ij} = 0$ for all $i, j \in [n]$ with $j > i$ and $|\lambda_{ij}| \leq 1$ for all $i, j \in [J]$.. To see this observe that for each $k \in [d]$ and j_1 and j_2 with

$j_1 \leq j_2$ there is a parametrization such that only the contribution from row j_2 from block (k, j_1) is non zero:

$$\begin{aligned}
&\lambda_{j_2 j_1} = 1 && \\
&\lambda_{j_2(j_1-1)} = -1 && \text{if } j_1 < j_2 \text{ and } j_1 \geq 1 \\
&\lambda_{jl} = 0 && \text{else} \\
&\vartheta_k = e_k && \text{for } k = j_0 \\
&\vartheta_k = 0 && \text{else.}
\end{aligned}$$

Thus since any coreset coreset preserves the rank of the matrix any coreset must also be of rank at least $dJ(J-1)$ and thus of size at least $\Omega(dJ^2)$. □

Lemma 3.3.6. *There exists a dataset $\{a_{ij}\}_{i\in[n]j\in[J]}$ such that any coreset for MCTMs with $\varepsilon < 1$ has size at least $\Omega(dJ)$ even if $\lambda_{ii} = 1$ for all $i \in [J]$ and $\lambda_{ij} = 0$ for $i < j$.*

Proof. Consider the instance with $n = d$ and $a_{ij} = e_i$ for all $i \in [n]$ and $j \in [J]$. Now consider any instance $\{a'_{ij}\}_{i\in[n],j\in[J]}$ with at least one zero entry, i.e. $a'_{i_0 j_0} = 0$ for some $i_0 \in [n], j_0 \in [J]$. Then A' cannot be a coreset of A for the following reason: let $A' \in \mathbb{R}^{n\times d}$ be the matrix consisting of rows $a'_{1j_0}, a'_{2j_0}, \ldots a'_{nj_0}$. Since $n = d$ and $a'_{i_0 j_0} = 0$ there exists $\beta \in ker(A') \setminus \{0\}$. Consider the following parameterizations:

$$\begin{aligned}
&\lambda_{jl} = 0 && \text{for } j > l \text{ and } l = j_0 \\
&\lambda_{jj} = 1 && \text{for all } j \in [J] \\
&\lambda_{jl} = 0 && \text{else} \\
&\vartheta_k = \beta && \text{for } k = j_0 \\
&\vartheta_k = 0 && \text{else}
\end{aligned}$$

Now we have that

$$\sum_{i=1}^{n}\sum_{j=1}^{J}\left(\sum_{k=1}^{j}\lambda_{j,j}\vartheta_k a_{ij}\right)^2 = \|\beta\|_2^2 > 0$$

and

$$\sum_{i=1}^{n}\sum_{j=1}^{J}\left(\sum_{k=1}^{j}\lambda_{j,j}\vartheta_k a'_{ij}\right)^2 = 0$$

thus $\{a'_{ij}\}_{i\in[n],j\in[J]}$ cannot be a coreset. □

A.2.6 Sensitivity sampling framework

The sensitivity sampling framework, as outlined and most recently updated in (Feldman et al., 2020), provides a methodology for generating coresets for optimization problems where the objective is to reduce the cost associated with and aggregate over the input data points. In this method, data points are selected at random, yet the selection probability for each point is proportional to its sensitivity with respect to

the optimization problem, as an importance measure. This technique is designed to ensure the inclusion of pivotal points that might otherwise be missed under an unbiased uniform random selection due to the equally low probability of selection.

Definition A.2.1 (Sensitivity (Langberg and Schulman, 2010)). *Let $F = \{g_1, \ldots, g_n\}$ be a family of functions that map from $\mathbb{R}^d$ to the non-negative real numbers, weighted with $w \in \mathbb{R}^n_{>0}$. The sensitivity of g_i for $f_w(\theta) = \sum_{j=1}^n w_j \cdot g(x_j\theta)$ is*

$$\zeta_i = \sup_{\theta \in \mathbb{R}^d, f_w(\theta) > 0} \frac{w_i g_i(\theta)}{f_w(\theta)}.$$

The sum of the sensitivities $Z = \sum_{i=1}^n \zeta_i$ is called the total sensitivity.

The sensitivity sampling framework has demonstrated considerable advantages in the construction of coresets across various computational problems and statistical models, including linear regression (Clarkson, 2005; Drineas et al., 2006; Rudelson and Vershynin, 2007; Dasgupta et al., 2009), logistic regression (Munteanu et al., 2018), probit regression (Ding et al., 2024), and Poisson regression (Molina et al., 2018; Lie and Munteanu, 2024).

Sampling with probabilities proportional to sensitivity scores provably leads to a good approximation, although it requires the determination of the exact sensitivities to solve the original problem. However, it has been shown that the sample can also be drawn proportionally to any upper bounds such that $S = \sum_{i=1}^n s_i \geq \sum_{i=1}^n \zeta_i = Z$. Therefore, in order to be able to build a coreset using sensitivity sampling, it suffices to find the upper bounds to approximate a sensitivity. However, since the total sensitivity determines the sample size, this overestimation must be controlled carefully.

The following theorem builds the core for sensitivity sampling based coreset construction and is presented in its most recently updated and optimized version due to (Feldman et al., 2020).

Theorem A.2.2 ((Langberg and Schulman, 2010; Feldman and Langberg, 2011; Feldman et al., 2020)). *Let $F = \{g_1, \ldots, g_n\}$ be a finite set of functions that map from $\mathbb{R}^d$ to $\mathbb{R}_{\geq 0}$ and let $w \in \mathbb{R}^n_{>0}$ be a vector of positive weights. Let ε, δ be in $(0, 1/2)$. Moreover, let $s_i \geq \zeta_i$ be upper bounds for the sensitivities and $S = \sum_{i=1}^n s_i \geq Z$. Then for given s_i, a set $R \subseteq F$ can be found in time $O(|F|)$ with*

$$|R| = O\left(\frac{S}{\varepsilon^2}\left(\Delta \log S + \log\left(\frac{1}{\delta}\right)\right)\right)$$

With the calculations of the weighted functions, such that with probability $1 - \delta$ for all $\theta \in \mathbb{R}^d$ it holds:

$$(1-\varepsilon)\sum_{g \in F} w_i g_i(\theta) \leq \sum_{g \in R} u_i g_i(\theta) \leq (1+\varepsilon)\sum_{g \in F} w_i g_i(\theta)$$

Each element of R is drawn i.i.d. with probability $p_j = \frac{s_j}{S}$ from F, $u_i = \frac{S w_j}{s_j |R|}$ denotes the weight of a function $g_i \in R$, which corresponds to $g_i \in F$, and Δ is an upper bound for the

VC dimension of the Range Spaces $\mathcal{R}_{F^*}$, induced by F^* that we obtain by scaling all functions $g_i \in F$ with $\frac{Sw_j}{s_j|R|}$. It is thus

$$F^* = \left\{ \frac{Sw_j}{s_j|R|} g_j(\theta) \;\middle|\; j \in [n] \right\}.$$

The set of functions $R \subseteq F$ with new weights u is a representation of our sought coreset. The size of R depends on both, the estimate of the total sensitivity, as well as the VC-dimension of the range spaces, which is induced by the reweighted function set F^*. Since the construction of the coreset is a probabilistic process, it cannot be ruled out that this may fail (unless we choose all data points for our coreset). The theorem thus introduces an controllable error probability δ, which also influences the size of the coreset logarithmically.

A.2.7 Relationship between MCTMs and Normalizing Flows

Multivariate Conditional Transformation Models (MCTMs) and Normalizing Flows (NFs) are both powerful frameworks for modeling complex conditional probability distributions $p_Y(y \mid x)$. They are built upon the same mathematical foundation, namely the probability transformation formula, which enables density estimation by transforming a complex target distribution into a simpler base distribution through a bijective and differentiable function.

Let $Y \in \mathbb{R}^J$ be the response variable whose conditional density $p_Y(y \mid x)$ is of interest, and let $Z \in \mathbb{R}^J$ be a latent variable with a known base distribution $p_Z(z)$, typically a standard multivariate Gaussian. Both frameworks assume the existence of a transformation function $h : \mathbb{R}^J \times \mathcal{X} \to \mathbb{R}^J$ that is bijective in y for each fixed x, such that

$$Z = h(Y \mid x).$$

The inverse function $g = h^{-1}$ satisfies

$$Y = g(Z \mid x).$$

By the probability transformation formula, the conditional density of Y given x can be expressed as

$$p_Y(y \mid x) = p_Z(h(y \mid x)) \left| \det \left(\frac{\partial h(y \mid x)}{\partial y} \right) \right|.$$

Taking the logarithm yields

$$\log p_Y(y \mid x) = \log p_Z(h(y \mid x)) + \log \left| \det \left(\frac{\partial h(y \mid x)}{\partial y} \right) \right|,$$

which is the objective typically maximized in both MCTM and NF during maximum likelihood estimation respectively training.

In MCTM, the transformation $h(y \mid x)$ is defined in a structured, semi-parametric way. It follows a lower triangular structure such that the j-th component $h_j(y \mid x)$

depends only on $y_1, \ldots, y_j$ and x. This triangular form ensures that the Jacobian matrix $\frac{\partial h(y|x)}{\partial y}$ is lower triangular. Each h_j is further decomposed as a linear combination of marginal transformations $\tilde{h}_\ell(y_\ell \mid x)$ for $\ell < j$, and its own marginal $\tilde{h}_j(y_j \mid x)$. Formally,

$$h_j(y_1, \ldots, y_j \mid x) = \sum_{\ell=1}^{j-1} \lambda_{j\ell}(x)\tilde{h}_\ell(y_\ell \mid x) + \tilde{h}_j(y_j \mid x).$$

In matrix-vector notation, the full transformation is written as $h(y \mid x) = \Lambda(x)\tilde{h}(y \mid x)$, where $\tilde{h}(y \mid x) = (\tilde{h}_1(y_1 \mid x), \ldots, \tilde{h}_J(y_J \mid x))^T$, and $\Lambda(x)$ is a lower triangular matrix with ones on the diagonal. Each marginal transformation $\tilde{h}_j$ is modeled using a basis function expansion of the form $\tilde{h}_j(y_j \mid x) = a_j(y_j)^T\vartheta_j(x)$, where $a_j(y_j)$ is a fixed basis vector and $\vartheta_j(x)$ is a parameter vector that may depend on x. Monotonicity is enforced by constraining the derivative $\frac{\partial \tilde{h}_j}{\partial y_j} > 0$. The Jacobian determinant simplifies significantly due to the triangular structure: it is the product of the marginal derivatives,

$$\det\left(\frac{\partial h(y \mid x)}{\partial y}\right) = \prod_{j=1}^{J} \frac{\partial \tilde{h}_j(y_j \mid x)}{\partial y_j} = \prod_{j=1}^{J} (a_j'(y_j))^T\vartheta_j(x).$$

Therefore, the conditional log-likelihood becomes

$$\log p_Y(y \mid x) = \log \phi_J(\Lambda(x)\tilde{h}(y \mid x)) + \sum_{j=1}^{J} \log((a_j'(y_j))^T\vartheta_j(x)),$$

where ϕ_J denotes the density of the standard J-variate normal distribution.

Normalizing Flows employ the same principle of density transformation, but define $h(y \mid x)$ or its inverse $g(z \mid x)$ through a sequence of invertible mappings composed of simple building blocks. These mappings are typically parameterized using deep neural networks. A transformation in NF is written as $h = h_L \circ \cdots \circ h_1$, where each h_ℓ is a bijective transformation with tractable Jacobian determinant. The parameters of each transformation may be functions of x, enabling conditional modeling.

Popular designs include autoregressive flows, where each output h_j depends only on $y_1, \ldots, y_{j-1}$ and x, resulting in a lower triangular Jacobian similar to MCTMs. Another class of NF architectures is based on coupling layers, where the input vector is split into two parts, one of which remains fixed while the other is transformed using an affine function whose parameters are learned from the fixed part and the context x. These transformations are composed repeatedly, with role reversal between parts in successive layers to ensure expressiveness.

Despite their differences in parameterization and implementation, MCTM and NF share the same mathematical foundation and rely on the same probability transformation principle. Both frameworks define a transformation from the observed data space to a latent space with a known density, compute the associated Jacobian determinant, and maximize the resulting log-likelihood over the data.

Appendix B

Supplementary Experimental Results

B.1 Supplementary Experimental Results of Ding et al. (2024)

B.1.1 Figures

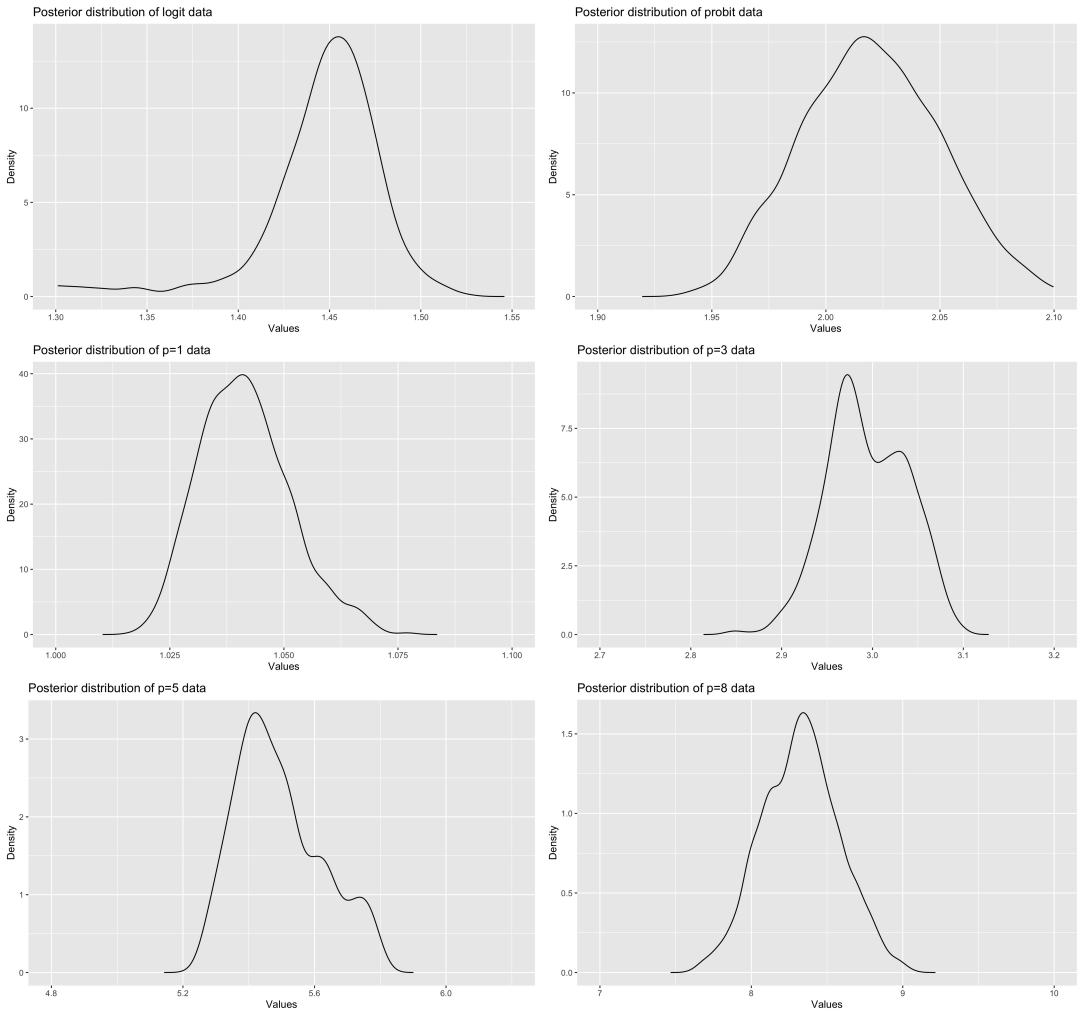


FIGURE B.1: The estimated posterior distributions of p for simulated data with $N = 50\,000$ omitting a burn-in phase of 500 iterations. First row: logit data, probit data. Second row: $p = 1$ and $p = 3$. Third row: $p = 5$ and $p = 8$.

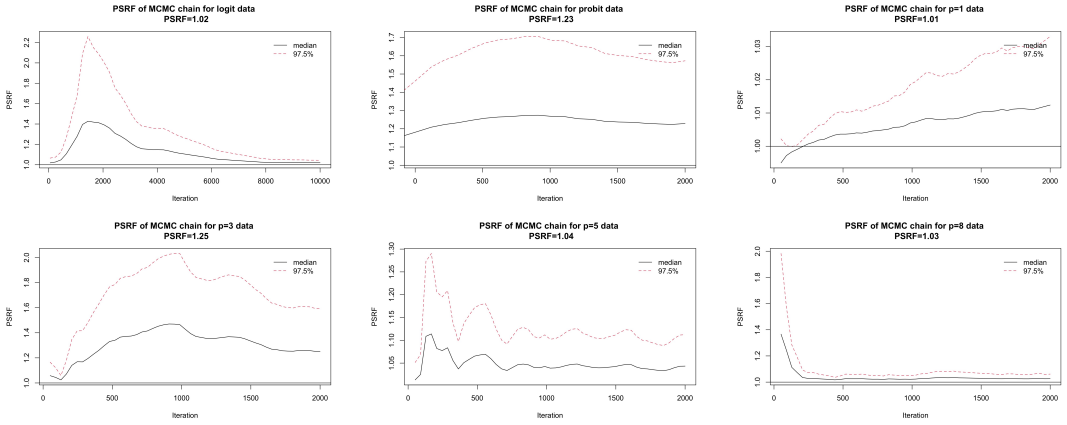


FIGURE B.2: The Potential Scale Reduction Factors (PSRF) of p for simulated data with $N = 5\,000$. First row: logit data, probit data, and $p = 1$. Second row: $p = 3$, $p = 5$, and $p = 8$.

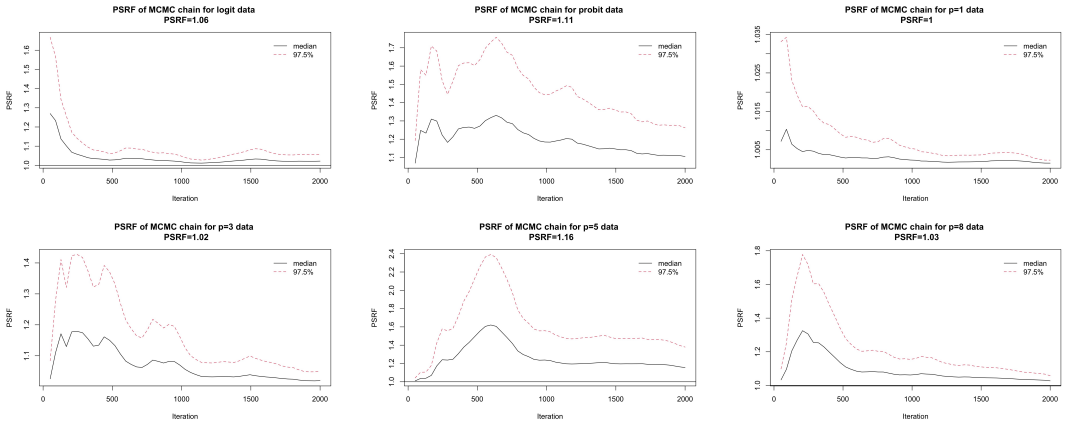


FIGURE B.3: The Potential Scale Reduction Factors (PSRF) of p for simulated data with $N = 10\,000$. First row: logit data, probit data, and $p = 1$. Second row: $p = 3$, $p = 5$, and $p = 8$.

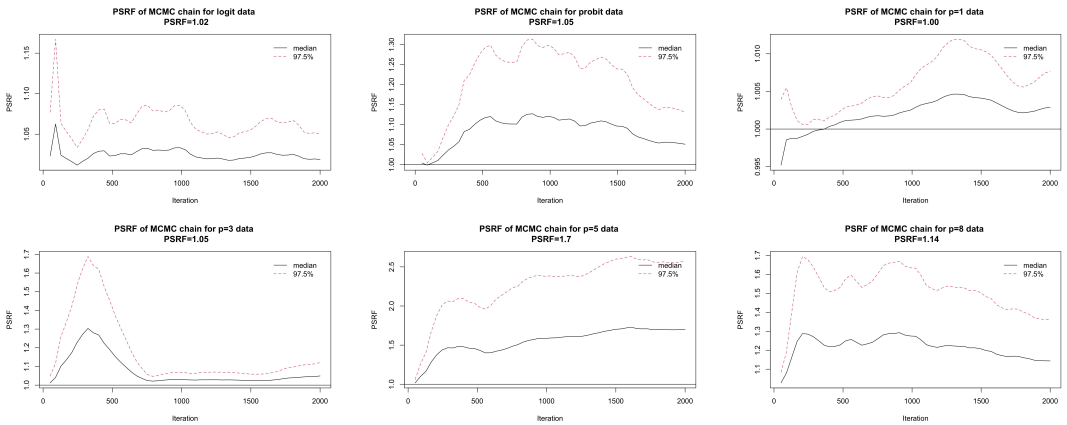


FIGURE B.4: The Potential Scale Reduction Factors (PSRF) of p for simulated data with $N = 20\,000$. First row: logit data, probit data, and $p = 1$. Second row: $p = 3$, $p = 5$, and $p = 8$.

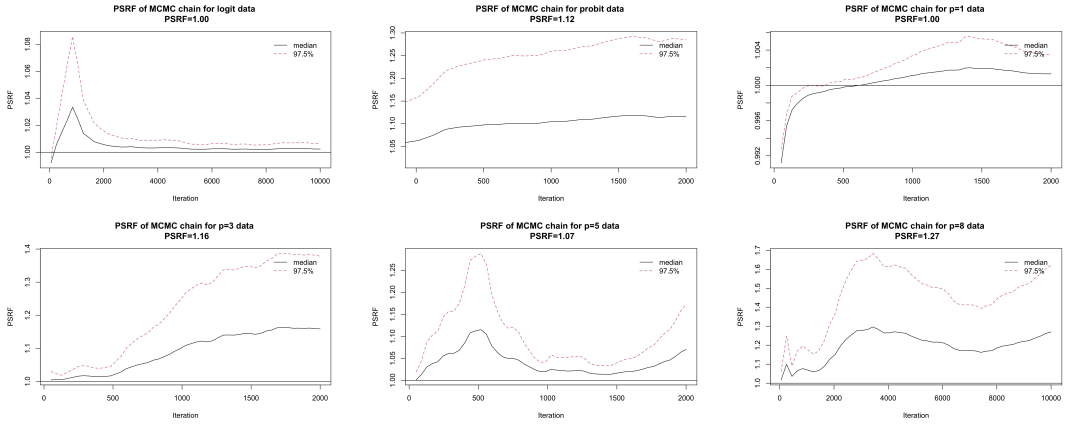


FIGURE B.5: The Potential Scale Reduction Factors (PSRF) of p for simulated data with $N = 50\,000$. First row: logit data, probit data, and $p = 1$. Second row: $p = 3$, $p = 5$, and $p = 8$.

B.1.2 Tables

B.1.3 Simulation results for fixed p

TABLE B.1: Model comparison for different fixed p scenarios using p-probit, probit, logit and cloglog link functions for $N = 5\,000$. * indicates the smallest RSS and AE values, ** indicates the second smallest RSS and AE values.

Scenarios	**p-probit link**		**probit link**		**logit link**		**cloglog link**	
	RSS	AE	RSS	AE	RSS	AE	RSS	AE
Logit data	2.902	59.621	0.530*	36.286**	0.843**	30.530*	6.156	99.389
probit data	0.807**	25.017	0.486*	17.725*	0.820	24.685**	2.535	50.230
$p = 0.5$	7.281*	83.000*	23.376	219.272	11.992**	150.764**	45.562	324.353
$p = 1$	2.011*	36.153*	3.731	63.849	2.087**	42.089**	12.712	136.547
$p = 1.5$	2.208	42.655	1.515**	35.754**	1.659*	35.042*	2.988	55.022
$p = 3$	1.489**	29.930**	1.059*	26.039*	1.436	33.199	3.317	52.676
$p = 4$	1.674**	29.839**	1.342*	28.552*	1.780	35.739	3.879	53.436
$p = 5$	1.705*	29.799*	2.680**	38.163**	3.803	47.477	4.415	53.962
$p = 8$	1.854*	28.953*	3.776**	42.628**	4.842	50.341	3.681	49.187

TABLE B.2: Model comparison for different fixed p scenarios using p-probit, probit, logit and cloglog link functions for $N = 10\,000$. * indicates the smallest RSS and AE values, ** indicates the second smallest RSS and AE values.

Scenarios	***p*-probit link**		**probit link**		**logit link**		**cloglog link**	
	RSS	AE	RSS	AE	RSS	AE	RSS	AE
Logit data	4.817	174.792	1.983**	113.217**	1.895*	104.772*	11.126	290.799
probit data	0.648 **	56.917**	0.555*	54.363*	0.834	74.138	14.135	305.693
$p = 0.5$	68.201	634.357	32.276**	494.286**	22.663 *	407.023*	60.849	702.322
$p = 1$	1.720 *	87.923*	9.175	247.763	4.726**	166.047**	32.899	503.431
$p = 1.5$	1.217*	80.314*	1.932	105.110	1.393**	84.426**	15.601	326.186
$p = 3$	1.492*	80.535*	3.034**	122.068**	5.372	177.548	10.432	251.769
$p = 4$	1.631 *	84.170*	4.377 **	152.791**	7.577	214.100	9.021	228.528
$p = 5$	1.710**	83.105*	6.848*	186.192**	11.002	248.507	12.827	264.621
$p = 8$	0.995*	61.392*	7.966**	207.240**	11.939	265.612	13.051	264.066

TABLE B.3: Model comparison for different fixed p scenarios using p-probit, probit, logit and cloglog link functions for $N = 20\,000$. * indicates the smallest RSS and AE values, ** indicates the second smallest RSS and AE values.

Scenarios	***p*-probit link**		**probit link**		**logit link**		**cloglog link**	
	RSS	AE	RSS	AE	RSS	AE	RSS	AE
Logit data	8.189	310.193	3.505**	197.954**	1.936*	133.325*	18.951	490.983
probit data	2.539**	135.983**	2.417*	128.596*	5.030	202.133	23.982	501.766
$p = 0.5$	3.814*	150.980*	77.922	1022.505	50.465 **	1022.502**	63.519	1635.157
$p = 1$	2.046 *	120.853*	20.009	467.888	8.635**	286.183**	35.306	1442.950
$p = 1.5$	0.937*	80.504*	2.479	154.448	1.171**	101.367**	24.681	1286.137
$p = 3$	1.313*	89.729*	3.719**	168.275**	8.407	278.710	22.872	1576.779
$p = 4$	1.898*	105.718*	7.352**	236.262**	13.137	340.259	31.323	340.259
$p = 5$	1.322*	88.130*	10.890**	283.692**	17.863	384.601	29.337	530.940
$p = 8$	2.620*	116.885*	11.834**	296.972**	18.519	393.044	33.297	541.105

B.1.4 Simulation results for parameter estimation for p

TABLE B.4: The estimation results for p, based on different p-GGDs generated simulations, for $N = 5\,000$, $10\,000$, $20\,000$, and $50\,000$. The different rows show different link functions of generated data, and the columns represent the estimated values $\hat{p}$ of p, the variance of $\hat{p}$, the distance between the true and estimated parameters β, and the residual sum of squares, respectively.

Criterion	$N = 5\,000$				$N = 10\,000$			
	$\hat{p}$	$\hat{\sigma}$	$\|\|\boldsymbol{\beta} - \hat{\boldsymbol{\beta}}\|\|_2$	RSS	$\hat{p}$	$\hat{\sigma}^2$	$\|\|\boldsymbol{\beta} - \hat{\boldsymbol{\beta}}\|\|_2$	RSS
Logit data	1.279	0.030	4.033	254.010	1.646	0.041	5.554	1752.570
probit data	2.355	0.064	0.512	150.104	2.033	0.023	0.773	1246.423
$p = 0.5$	0.517	0.001	1.039	166.088	0.652	0.006	1.922	1275.791
$p = 1$	0.997	0.017	0.446	155.312	0.930	0.041	0.832	1255.551
$p = 1.5$	2.061	0.044	0.801	151.908	1.568	0.035	0.936	1247.989
$p = 3$	4.481	0.252	1.768	152.645	4.039	0.158	1.717	1252.065
$p = 4$	4.310	0.161	0.446	152.474	4.070	0.126	0.897	1254.704
$p = 5$	7.483	0.810	1.184	153.056	4.234	0.187	0.900	1259.295
$p = 8$	9.574	0.359	0.773	155.006	9.351	0.436	0.784	1267.948
	$N = 20\,000$				$N = 50\,000$			
	$\hat{p}$	$\hat{\sigma}$	$\|\|\boldsymbol{\beta} - \hat{\boldsymbol{\beta}}\|\|_2$	RSS	$\hat{p}$	$\hat{\sigma}^2$	$\|\|\boldsymbol{\beta} - \hat{\boldsymbol{\beta}}\|\|_2$	RSS
Logit data	1.453	0.013	6.280	2752.739	1.472	0.046	5.679	2563.471
probit data	2.045	0.012	0.330	1672.447	2.002	0.006	0.123	1378.033
$p = 0.5$	0.611	0.005	2.027	1766.327	0.505	0.001	0.285	1470.865
$p = 1$	0.944	0.019	0.462	1692.259	1.069	0.010	0.564	1399.571
$p = 1.5$	1.442	0.026	0.805	1677.101	1.390	0.020	0.517	1385.033
$p = 3$	3.060	0.083	0.329	1679.120	3.021	0.033	0.449	1382.025
$p = 4$	4.769	0.040	0.938	1683.829	4.128	0.064	0.295	1389.862
$p = 5$	5.881	0.210	0.297	1689.743	5.507	0.135	0.259	1395.519
$p = 8$	9.240	0.403	0.960	1697.123	8.525	0.576	0.381	1401.118

TABLE B.5: Checking for proper posterior and PSRF convergence. PSRF smaller than 1.2 indicates that the chains are converged.

Scenarios	**N=5 000**		**N=10 000**		**N=20 000**		**N=50 000**	
	PSRF	Integral	PSRF	Integral	PSRF	Integral	PSRF	Integral
Logit data	1.02	1.0009	1.06	1.0097	1.02	1.0009	1.03	1.0007
probit data	1.21	1.0009	1.11	1.0004	1.05	1.0009	1.11	1.0008
$p = 0.5$	1.02	1.0004	1.02	1.0049	1.04	1.0008	1.02	1.0212
$p = 1$	1.01	1.0009	1.00	1.0025	1.00	1.0006	1.03	1.0012
$p = 1.5$	1.16	1.0010	1.02	1.0010	1.01	1.0009	1.05	1.0006
$p = 3$	1.25	1.0009	1.02	1.0008	1.05	1.0009	1.13	1.0009
$p = 4$	1.02	1.0009	1.09	1.0009	1.05	1.0008	1.08	1.0009
$p = 5$	1.04	1.0009	1.16	1.0008	1.7	1.0009	1.08	1.0009
$p = 8$	1.03	1.0009	1.03	1.0009	1.14	1.0009	1.27	1.0009

FIGURE B.6: Credible intervals for p=1 and p=2 under Bayesian estimation and confidence intervals using logit and probit links under maximum likelihood estimation, respectively, for credit card data (upper row) and heart disease data (lower row)

Appendix C

Algorithms

C.1 Coreset Algorithm

Algorithm 6 Coreset algorithm for p-generalized probit regression. (Munteanu et al., 2022)

Initialize sketch $X'' = \mathbf{0} \in \mathbb{R}^{n' \times d}$, where $n' = O(d^2)$ for $p \leq 2$ or $n' = O(n^{1-2/p} \log n \cdot \text{poly}(d))$ for $p > 2$);
for $i = 1$ **To** n **do**
    Draw a random number $B_i \in [n']$; ▷ hash to bucket B_i
    Draw a random number $\sigma_i \in \{-1, 1\}$; ▷ random sign
    if $p \neq 2$ **then**
        Draw a random number $\lambda_i \sim \exp(1)$; ▷ ℓ_p embedding
        $\sigma_i = \sigma_i / \lambda_i^{1/p}$
    end if
    $X''_{B_i} = X''_{B_i} + \sigma_i \cdot x_i$. ▷ sketch
end for
Compute the QR-decomposition of $X'' = QR$. ▷ well conditioned basis
Initialize coreset $X' = \mathbf{0} \in \mathbb{R}^{k \times d}$ ▷ coreset points
Initialize weights $w = 0 \in \mathbb{R}^k$ ▷ coreset weights
Initialize k independent weighted reservoir samplers S_j, sampling row X'_j , for each $j \in [k]$;
Initialize $G = I \in \mathbb{R}^{d \times d}$; ▷ Identity matrix
if $p = 2$ and $\log n < d$ **then**
    Draw $G \in \mathbb{R}^{d \times \log n}$ with $G_{ij} \sim N(0, \frac{1}{\log n})$ ▷ JL-embedding
end if
for $i = 1$ to n **do**
    Compute $q_i = \|X_i(R^{-1}G)\|_p^p$ ▷ ℓ_p-leverage score approximation
    for $j = 1$ to k **do**
        Feed $s_i = q_i + 1/n$ to S_j ; ▷ unnormalized sampling probabilities
        if S_j samples x_i **then**
            $w_j = 1/(k \cdot s_i)$; ▷ unnormalized weights
            $X'_j = x_i$; ▷ save row identity in the coreset
        end if
    end for
end for
$w = w \cdot \sum_{i=1}^n s_i$; ▷ normalize weights **return** $C = (X', w)$

Algorithm 7 Sparse approximation of the convex hull (Blum et al., 2019)

Input: A set of points P, a query point q, tolerance ϵ
Output: Approximated convex hull point t_M closest to q
Initialize first two points: randomly select one point a_0, then obtain a_1 as the furthest point from a_0
Obtain a_2 as the furthest point from the line $\overline{a_0 a_1}$; the set $\{a_0, a_1, a_2\}$ forms the initial convex hull
for $j \leftarrow 1$ to $n - 3$ **do**
    $t_0 \leftarrow$ closest point of P to q
    $M \leftarrow O(1/\epsilon^2)$
    for $i \leftarrow 1$ to M **do**
        $v_i \leftarrow q - t_{i-1}$
        $p_i \leftarrow$ point in P extremal in the direction of v_i
        if p_i exists **then**
            Compute the projection of q onto the line through t_{i-1} and p_i to find t_i
            $t_i \leftarrow$ closest point to q on line segment $s_i = t_{i-1} p_i$
        else
            $t_i \leftarrow t_{i-1}$
            break
        end if
        if $\|q - t_i\| < \epsilon$ or $i = M$ **then**
            return t_i
        end if
    end for
end for

Bibliography

Oriol Abril-Pla, Virgile Andreani, Colin Carroll, Larry Dong, Christopher J. Fonnesbeck, Maxim Kochurov, Ravin Kumar, Junpeng Lao, Christian C. Luhmann, Osvaldo A. Martin, Michael Osthege, Ricardo Vieira, Thomas Wiecki, and Robert Zinkov (2023). "PyMC: A modern and comprehensive probabilistic programming framework in Python". In: *PeerJ Computer Science* 9.e1516. DOI: 10.7717/peerj-cs.1516.

Pankaj K. Agarwal, Sariel Har-Peled, and Kasturi R. Varadarajan (2004). "Approximating extent measures of points". In: *J. ACM* 51.4, pp. 606–635. DOI: 10.1145/1008731.1008736.

Pankaj K Agarwal, Sariel Har-Peled, and Kasturi R Varadarajan (2005). "Geometric approximation via coresets". In: *Combinatorial and computational geometry* 52.1, pp. 1–30.

Sungjin Ahn, Anoop Korattikara Balan, and Max Welling (2012). "Bayesian posterior sampling via stochastic gradient Fisher scoring". In: *Proceedings of the 29th International Conference on Machine Learning (ICML)*.

Nir Ailon and Edo Liberty (2009). "Fast dimension reduction using Rademacher series on dual BCH codes". In: *Discrete & Computational Geometry* 42.4, pp. 615–630.

James H. Albert and Siddhartha Chib (1993). "Bayesian analysis of binary and polychotomous response data". In: *Journal of the American Statistical Association* 88.422, pp. 669–679.

Fabian Altekrüger, Johannes Hertrich, and Gabriele Steidl (2023). "Neural Wasserstein gradient flows for discrepancies with Riesz kernels". In: *Proceedings of the 40st International Conference on Machine Learning (ICML)*. PMLR, pp. 664–690.

Martin Arjovsky, Soumith Chintala, and Léon Bottou (2017). "Wasserstein generative adversarial networks". In: *International Conference on Machine Learning (ICML)*, pp. 214–223.

Olivier Bachem, Mario Lucic, and Silvio Lattanzi (2018). "One-shot coresets: The case of k-clustering". In: *Proceedings of the 21st International Conference on Artificial Intelligence and Statistics, (AISTATS)*. Vol. 84, pp. 784–792.

Rémi Bardenet, Arnaud Doucet, and Chris Holmes (2017). "On Markov chain Monte Carlo methods for tall data". In: *Journal of Machine Learning Research* 18.1, 1515–1557.

Rémi Bardenet, Arnaud Doucet, and Chris Holmes (2014). "Towards scaling up Markov chain Monte Carlo: An adaptive subsampling approach". In: *Proceedings of the 31st International Conference on Machine Learning (ICML)*. Vol. 32, pp. 405–413.

Mathieu Besançon, Theodore Papamarkou, David Anthoff, Alex Arslan, Simon Byrne, Dahua Lin, and John Pearson (2021). "Distributions.jl: Definition and modeling of probability distributions in the JuliaStats ecosystem". In: *Journal of Statistical Software* 98.16, pp. 1–30. DOI: 10.18637/jss.v098.i16.

Jock Blackard (1998). *Covertype*. UCI Machine Learning Repository. DOI: 10.24432/C50K5N.

David M Blei, Alp Kucukelbir, and Jon D McAuliffe (2017). "Variational inference: A review for statisticians". In: *Journal of the American Statistical Association* 112.518, pp. 859–877.

Avrim Blum, Sariel Har-Peled, and Benjamin Raichel (2019). "Sparse approximation via generating point sets". In: *ACM Transactions on Algorithms (TALG)* 15.3, pp. 1–16.

Salomon Bochner (1959). *Lectures on Fourier integrals*.: Princeton University Press.

Dean A Bodenham and Yoshinobu Kawahara (2023). "euMMD: Efficiently computing the MMD two-sample test statistic for univariate data". In: *Statistics and Computing* 33.5, p. 110.

Nicolas Bonneel, Julien Rabin, Gabriel Peyré, and Hanspeter Pfister (2015). "Sliced and Radon Wasserstein barycenters of measures". In: *Journal of Mathematical Imaging and Vision* 51, pp. 22–45.

Nicolas Bonnotte (2013). "Unidimensional and evolution methods for optimal transportation". PhD thesis. Université Paris Sud-Paris XI; Scuola normale superiore (Pise, Italie).

George EP Box and George C Tiao (2011). *Bayesian inference in statistical analysis*. John Wiley & Sons.

Tamara Broderick, Nicholas Boyd, Andre Wibisono, Ashia C Wilson, and Michael I Jordan (2013). "Streaming variational Bayes". In: *Advances in Neural Information Processing Systems (NeurIPS)*. Vol. 26, pp. 1727–1735.

Sebastien Bubeck (2015). "Convex optimization: Algorithms and complexity". In: *Foundations and Trends in Machine Learning* 8.3-4, pp. 231–357. DOI: 10.1561/2200000050.

Trevor Campbell and Tamara Broderick (2018). "Bayesian coreset construction via greedy iterative geodesic ascent". In: *Proceedings of the 35th International Conference on Machine Learning (ICML)*, pp. 698–706.

Trevor Campbell and Tamara Broderick (2019). "Automated scalable Bayesian inference via Hilbert coresets". In: *Journal of Machine Learning Research* 20.1, 551–588.

Trevor Campbell, Julian Straub, John W Fisher III, and Jonathan P How (2015). "Streaming, distributed variational Inference for Bayesian nonparametrics". In: *Advances in Neural Information Processing Systems (NeurIPS)*. Vol. 28, pp. 280–288.

Manuel Carlan, Thomas Kneib, and Nadja Klein (2024). "Bayesian conditional transformation models". In: *Journal of the American Statistical Association* 119.546, pp. 1360–1373.

Bob Carpenter, Andrew Gelman, Matthew D. Hoffman, Daniel Lee, Ben Goodrich, Michael Betancourt, Marcus Brubaker, Jiqiang Guo, Peter Li, and Allen Riddell (2017). "Stan: A probabilistic programming language". In: *Journal of Statistical Software* 76.1. DOI: 10.18637/jss.v076.i01.

Centers for Disease Control and Prevention (2015). *Behavioral risk factor surveillance system survey data*.

Elizabeth A. Chambers and D. R. Cox (1967). "Discrimination between alternative binary response models". In: *Biometrika* 54.3/4, pp. 573–578.

Alan Chan, Hugo Silva, Sungsu Lim, Tadashi Kozuno, A Rupam Mahmood, and Martha White (2022). "Greedification operators for policy optimization: Investigating forward and reverse KL divergences". In: *Journal of Machine Learning Research* 23.253, pp. 1–79.

Timothy M Chan (2004). "Faster core-set constructions and data stream algorithms in fixed dimensions". In: *Proceedings of the twentieth annual symposium on Computational geometry*, pp. 152–159.

Moses Charikar, Michael Kapralov, Navid Nouri, and Paris Siminelakis (2020). "Kernel density estimation through density constrained near neighbor Search". In: *61st IEEE Annual Symposium on Foundations of Computer Science, (FOCS)*. IEEE, pp. 172–183. DOI: 10.1109/FOCS46700.2020.00025.

Kenneth L. Clarkson (2005). "Subgradient and sampling algorithms for ℓ_1 regression". In: *Proceedings of the 16th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. Vancouver, British Columbia, 257–266.

Kenneth L. Clarkson, Petros Drineas, Malik Magdon-Ismail, Michael W. Mahoney, Xiangrui Meng, and David P. Woodruff (2016). "The fast Cauchy transform and faster robust linear regression". In: *SIAM Journal on Computing* 45.3, pp. 763–810. DOI: 10.1137/140963698.

Andrew Cotter, Joseph Keshet, and Nathan Srebro (2011). "Explicit approximations of the Gaussian kernel". In: *arXiv preprint arXiv:1109.4603*.

Claudia Czado (1992). "On link selection in generalized linear models". In: *Advances in GLIM and Statistical Modelling*. Ed. by Ludwig Fahrmeir, Brian Francis, Robert Gilchrist, and Gerhard Tutz. New York, NY: Springer New York, pp. 60–65.

Claudia Czado and Thomas J. Santner (1992). "The effect of link misspecification on binary regression inference". In: *Journal of Statistical Planning and Inference* 33.2, pp. 213–231. DOI: 10.1016/0378-3758(92)90069-5.

Anirban DasGupta (2008). *Asymptotic theory of statistics and probability*. Vol. 180. Springer.

Anirban Dasgupta, Petros Drineas, Boulos Harb, Ravi Kumar, and Michael W. Mahoney (2009). "Sampling algorithms and coresets for ℓ_p Regression". In: *SIAM Journal on Computing* 38.5, pp. 2060–2078. DOI: 10.1137/070696507.

Akash Kumar Dhaka, Alejandro Catalina, Manushi Welandawe, Michael R Andersen, Jonathan Huggins, and Aki Vehtari (2021). "Challenges and opportunities in high

dimensional variational inference". In: *Advances in Neural Information Processing Systems* 34, pp. 7787–7798.

Zeyu Ding, Cornelius Grunwald, Katja Ickstadt, Kevin Kröninger, and Salvatore La Cagnina (2025). "MCBench: A benchmark suite for Monte Carlo sampling algorithms". In: *arXiv preprint arXiv:2501.03138*.

Zeyu Ding, Katja Ickstadt, Nadja Klein, Alexander Munteanu, and Simon Omlor (2026). "Scalable Learning of Multivariate Distributions via Coresets". In: *Proceedings of the 29th International Conference on Artificial Intelligence and Statistics (AISTATS)*. To appear.

Zeyu Ding, Katja Ickstadt, and Alexander Munteanu (2023). "Bayesian analysis for dimensionality and complexity reduction". In: *Machine Learning under Resource Constraints, Volume 3 - Applications*. Berlin, Boston: De Gruyter. Chap. 2.4, pp. 58–70. DOI: doi:10.1515/9783110785982-012.

Zeyu Ding, Simon Omlor, Katja Ickstadt, and Alexander Munteanu (2024). "Scalable Bayesian p-generalized probit and logistic regression". In: *Advances in Data Analysis and Classification*. DOI: 10.1007/s11634-024-00599-1.

Petros Drineas, Michael W. Mahoney, and S. Muthukrishnan (2006). "Sampling algorithms for l_2 regression and applications". In: *Proceedings of the Seventeenth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. ACM Press, pp. 1127–1136.

Dan Feldman and Michael Langberg (2011). "A unified framework for approximating and clustering data". In: *Proceedings of the Forty-Third Annual ACM Symposium on Theory of Computing (STOC)*, 569–578. DOI: 10.1145/1993636.1993712.

Dan Feldman, Melanie Schmidt, and Christian Sohler (2020). "Turning big data into tiny data: Constant-zize coresets for k-Means, PCA, and projective clustering". In: *SIAM Journal on Computing* 49.3, pp. 601–657. DOI: 10.1137/18M1209854.

Erich Gamma (1995). *Design patterns: elements of reusable object-oriented software*. Pearson Education India.

Hanjia Gao and Xiaofeng Shao (2023). "Two sample testing in high dimension via maximum mean discrepancy". In: *Journal of Machine Learning Research* 24.304, pp. 1–33.

Andrew Gelman, John B. Carlin, Hal S. Stern, David B. Dunson, Aki Vehtari, and Donald B. Rubin (2013). *Bayesian Data Analysis, Third Edition*. Chapman & Hall/CRC Texts in Statistical Science. Taylor & Francis.

Andrew Gelman, Walter R Gilks, and Gareth O Roberts (1997). "Weak convergence and optimal scaling of random walk Metropolis algorithms". In: *The Annals of Applied Probability* 7.1, pp. 110–120.

Andrew Gelman, Aleks Jakulin, Maria Grazia Pittau, and Yu-Sung Su (2008). "A weakly informative default prior distribution for logistic and other regression models". In: *The Annals of Applied Statistics* 2.4, pp. 1360 –1383. DOI: 10.1214/08-AOAS191.

Andrew Gelman and Donald B Rubin (1992). "Inference from iterative simulation using multiple sequences". In: *Statistical science* 7.4, pp. 457–472.

Leo N. Geppert, Katja Ickstadt, Alexander Munteanu, Jens Quedenfeld, and Christian Sohler (2017). "Random projections for Bayesian regression". In: *Statistics and Computing* 27.1, pp. 79–101. DOI: 10.1007/s11222-015-9608-z.

Leo N. Geppert, Katja Ickstadt, Alexander Munteanu, and Christian Sohler (2020). "Streaming statistical models via Merge & Reduce". In: *International Journal of Data Science and Analytics* 10.4, pp. 331–347. DOI: 10.1007/s41060-020-00226-0.

Massimiliano Giacalone, Demetrio Panarello, and Raffaele Mattera (2018). "Multicollinearity in regression: An efficiency comparison between ℓ_p-norm and least squares estimators". In: *Quality & Quantity* 52.4, pp. 1831–1859. DOI: 10.1007/s11135-017-0571-y.

Tilmann Gneiting and Adrian E Raftery (2007). "Strictly proper scoring rules, prediction, and estimation". In: *Journal of the American Statistical Association* 102.477, pp. 359–378.

Ethan Goan and Clinton Fookes (2020). "Bayesian neural networks: An introduction and survey". In: *Case Studies in Applied Bayesian Data Science: CIRM Jean-Morlet Chair, Fall 2018*. Cham: Springer International Publishing, pp. 45–87. DOI: 10.1007/978-3-030-42553-1_3.

Gene H. Golub and Charles F. van Loan (2013). *Matrix Computations*. 4th. Baltimore: Johns Hopkins University Press.

Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio (2014). "Generative adversarial nets". In: *Advances in neural information processing systems* 27.

Irwin R Goodman and Samuel Kotz (1973). "Multivariate θ-generalized normal distributions". In: *Journal of Multivariate Analysis* 3.2, pp. 204–219. DOI: 10.1016/0047-259X(73)90023-7.

Arthur Gretton, Karsten Borgwardt, Malte Rasch, Bernhard Schölkopf, and Alex Smola (2006). "A kernel method for the two-sample-problem". In: *Advances in neural information processing systems* 19.

Arthur Gretton, Karsten M Borgwardt, Malte J Rasch, Bernhard Schölkopf, and Alexander Smola (2012a). "A kernel two-sample test". In: *The Journal of Machine Learning Research* 13.1, pp. 723–773.

Arthur Gretton, Dino Sejdinovic, Heiko Strathmann, Sivaraman Balakrishnan, Massimiliano Pontil, Kenji Fukumizu, and Bharath K Sriperumbudur (2012b). "Optimal kernel choice for large-scale two-sample tests". In: *Advances in neural information processing systems* 25.

Samuele Grossi, Marco Letizia, and Riccardo Torre (2025). "Refereeing the referees: Evaluating two-sample tests for validating generators in precision sciences". In: *Machine Learning: Science and Technology* 6.1, p. 015052.

Paul Hagemann, Johannes Hertrich, Fabian Altekrüger, Robert Beinert, Jannis Chemseddine, and Gabriele Steidl (2023). "Posterior sampling based on gradient flows of the MMD with negative distance kernel". In: *arXiv preprint arXiv:2310.03054*.

Matthias Herp, Johannes Brachem, Michael Altenbuchinger, and Thomas Kneib (2025). "Graphical transformation models". In: *arXiv:2503.17845*.

Johannes Hertrich, Christian Wald, Fabian Altekrüger, and Paul Hagemann (2023). "Generative sliced MMD flows with Riesz kernels". In: *arXiv preprint arXiv:2305.11463*.

J. R. M. Hosking and James R. Wallis (1997). *Regional frequency analysis: An approach based on L-moments*. Cambridge University Press. DOI: 10.1017/CBO9780511529443.

Torsten Hothorn, Thomas Kneib, and Peter Bühlmann (2014). "Conditional transformation models". In: *Journal of the Royal Statistical Society Series B: Statistical Methodology* 76.1, pp. 3–27.

Torsten Hothorn, Lisa Möst, and Peter Bühlmann (2018). "Most likely transformations". In: *Scandinavian Journal of Statistics* 45.1, pp. 110–134. DOI: https://doi.org/10.1111/sjos.12291.

Jonathan Huggins, Trevor Campbell, and Tamara Broderick (2016). "Coresets for scalable Bayesian logistic regression". In: *Advances in Neural Information Processing Systems (NeurIPS)*. Vol. 29, pp. 4080–4088.

Leonid V Kantorovich (1942). "On the translocation of masses". In: *Dokl. Akad. Nauk. USSR (NS)*. Vol. 37, pp. 199–201.

Michael J. Kearns and Umesh V. Vazirani (1994). *An introduction to computational learning theory*. Cambridge, MA, USA: MIT Press.

GB King, AE Lovell, L Neufcourt, and FM Nunes (2019). "Direct comparison between Bayesian and frequentist uncertainty quantification for nuclear reactions". In: *Physical review letters* 122.23, p. 232502.

Nadja Klein, Torsten Hothorn, Luisa Barbanti, and Thomas Kneib (2022). "Multivariate conditional transformation models". In: *Scandinavian Journal of Statistics* 49.1, pp. 116–142.

Thomas Kneib, Susanne Konrath, and Ludwig Fahrmeir (2011). "High dimensional structured additive regression models: Bayesian regularization, smoothing and predictive performance". In: *Journal of the Royal Statistical Society Series C: Applied Statistics* 60.1, pp. 51–70.

Ivan Kobyzev, Simon J.D. Prince, and Marcus A. Brubaker (2021). "Normalizing flows: An introduction and review of current methods". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 43.11, pp. 3964–3979. DOI: 10.1109/TPAMI.2020.2992934.

Roger Koenker and Jungmo Yoon (2009). "Parametric links for binary choice models: A Fisherian-Bayesian colloquy". In: *Journal of Econometrics* 152.2, pp. 120–130. DOI: 10.1016/j.jeconom.2009.01.009.

Soheil Kolouri, Kimia Nadjahi, Umut Simsekli, Roland Badeau, and Gustavo Rohde (2019). "Generalized sliced Wasserstein distances". In: *Advances in neural information processing systems* 32.

Alp Kucukelbir, Dustin Tran, Rajesh Ranganath, Andrew Gelman, and David M Blei (2017). "Automatic differentiation variational inference". In: *Journal of machine learning research* 18.14, pp. 1–45.

Solomon Kullback and Richard A Leibler (1951). "On information and sufficiency". In: *The annals of mathematical statistics* 22.1, pp. 79–86.

Sang Gyu Kwak and Jong Hae Kim (2017). "Central limit theorem: the cornerstone of modern statistics". In: *Korean journal of anesthesiology* 70.2, p. 144.

Michael Langberg and Leonard J. Schulman (2010). "Universal ε-approximators for integrals". In: *Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms*. SODA '10. Austin, Texas: Society for Industrial and Applied Mathematics, 598–607.

Dung Le, Huy Nguyen, Khai Nguyen, Trang Nguyen, and Nhat Ho (2024). "Fast approximation of the generalized sliced-Wasserstein distance". In: *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6920–6924.

Dan Li, Xia Wang, Lizhen Lin, and Dipak K. Dey (2016). "Flexible link functions in nonparametric binary regression with Gaussian process priors". In: *Biometrics* 72.3, pp. 707–719.

Tao Li and Jinwen Ma (2018). "Bayesian probit model with L^{α} and elastic net regularization". In: *Intelligent Computing Theories and Application: 14th International Conference, ICIC 2018, Wuhan, China, August 15-18, 2018, Proceedings, Part I 14*. Springer, pp. 309–321.

Zhu Li, Jean-Francois Ton, Dino Oglic, and Dino Sejdinovic (2019). "Towards a unified analysis of random Fourier features". In: *International Conference on Machine Learning (ICML))*, pp. 3905–3914.

Han Cheng Lie and Alexander Munteanu (2024). "Data subsampling for Poisson regression with *pth*-root-link". In: *Annual Conference on Neural Information Processing Systems (NeurIPS)*. Vol. 37, pp. 136286–136325.

Dahua Lin, John Myles White, Simon Byrne, Douglas Bates, Andreas Noack, John Pearson, Alex Arslan, Kevin Squire, David Anthoff, Theodore Papamarkou, Mathieu Besançon, Jan Drugowitsch, Moritz Schauer, and other contributors (July 2019). *JuliaStats/Distributions.jl: A Julia package for probability distributions and associated functions*. DOI: 10.5281/zenodo.2647458.

Måns Magnusson, Jakob Torgander, Paul-Christian Bürkner, Lu Zhang, Bob Carpenter, and Aki Vehtari (2024). "posteriordb: Testing, benchmarking and developing Bayesian inference algorithms". In: *arXiv preprint arXiv:2407.04967*.

Tung Mai, Alexander Munteanu, Cameron Musco, Anup Rao, Chris Schwiegelshohn, and David P. Woodruff (2023). "Optimal Sketching Bounds for Sparse Linear Regression". In: *Proceedings of the 26th International Conference on Artificial Intelligence and Statistics (AISTATS)*. Vol. 206, pp. 11288–11316.

Tung Mai, Anup B. Rao, and Cameron Musco (2021). "Coresets for classification - simplified and strengthened". In: *Advances in Neural Information Processing Systems 34 (NeurIPS)*, pp. 11643–11654.

Stephan Mandt, Florian Wenzel, Shinichi Nakajima, John Cunningham, Christoph Lippert, and Marius Kloft (2017). "Sparse probit linear mixed model". In: *Machine Learning* 106, pp. 1621–1642.

Richard McElreath (2018). *Statistical rethinking: A Bayesian course with examples in R and Stan*. Chapman and Hall/CRC.

Carmen van Meegen, Navina Waschinky, Katja Ickstadt, Tim Ricken, Swetlana Herbrandt, and Carla Henning (2025). "Uncertainty quantification for fluid saturated porous media - Bayesian analysis, variational sensitivity analysis, surrogate modeling and reliability analysis." In: *Polymorphic Uncertainty Modelling for the Numerical Design of Structures*. Ed. by Michael Kaliske and Wolfgang Graf. Lecture Notes in Applied and Computational Mechanics. submitted. Cham: Springer.

Mhamed Mesfioui, Taoufik Bouezmarni, and M. Belalia (June 2023). "Copula-based link functions in binary regression models". In: *Statistical Papers* 64, pp. 1–29. DOI: 10.1007/s00362-022-01330-y.

Nicholas Metropolis, Arianna W. Rosenbluth, Marshall N. Rosenbluth, Augusta H. Teller, and Edward Teller (June 1953). "Equation of state calculations by fast computing machines". In: *The Journal of Chemical Physics* 21.6, pp. 1087–1092. DOI: 10.1063/1.1699114.

Alejandro Molina, Alexander Munteanu, and Kristian Kersting (2018). "Core dependency networks". In: *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence (AAAI-18)*. AAAI Press, pp. 3820–3827. DOI: 10.1609/AAAI.V32I1.11726.

Gaspard Monge (1781). "Mémoire sur la théorie des déblais et des remblais". In: *Mem. Math. Phys. Acad. Royale Sci.*, pp. 666–704.

James R. Munkres (2000). *Topology*. 2nd ed. Prentice Hall.

Alexander Munteanu (2018). "On Large-Scale Probabilistic and Statistical Data Analysis". PhD thesis. Dortmund, Germany: Technische Universität Dortmund.

Alexander Munteanu (2023). "Coresets and sketches for regression problems on data streams and distributed data". In: *Machine Learning under Resource Constraints, Volume 1 - Fundamentals*. Berlin, Boston: De Gruyter. Chap. 3.2, pp. 85–97. DOI: doi:10.1515/9783110785944-003.

Alexander Munteanu and Simon Omlor (2024a). "Optimal bounds for ℓ_p sensitivity sampling via ℓ_2 augmentation". In: *Proceedings of the 41st International Conference on Machine Learning (ICML)*. PMLR, pp. 36769–36796.

Alexander Munteanu and Simon Omlor (2024b). "Turnstile ℓ_p leverage score sampling with applications". In: *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*.

Alexander Munteanu, Simon Omlor, and Christian Peters (2022). "p-Generalized probit regression and scalable maximum likelihood estimation via Sketching and Coresets". In: *Proceedings of the 25th International Conference on Artificial Intelligence and Statistics (AISTATS)*, pp. 2073–2100.

Alexander Munteanu, Simon Omlor, and David P. Woodruff (2021). "Oblivious sketching for logistic regression". In: *Proceedings of the 38th International Conference on Machine Learning (ICML)*, pp. 7861–7871.

Alexander Munteanu, Simon Omlor, and David P. Woodruff (2023). "Almost linear constant-factor sketching for ℓ_1 and logistic Regression". In: *Proceedings of the 11th International Conference on Learning Representations (ICLR)*, pp. 1–35.

Alexander Munteanu and Chris Schwiegelshohn (2018). "Coresets-methods and history: A theoreticians design pattern for approximation and streaming algorithms". In: *Künstliche Intelligenz* 32.1, pp. 37–53. DOI: 10.1007/s13218-017-0519-3.

Alexander Munteanu, Chris Schwiegelshohn, Christian Sohler, and David P. Woodruff (2018). "On coresets for logistic regression". In: *Proceedings of the 32nd International Conference on Neural Information Processing Systems (NeurIPS)*, 6562–6571.

Khai Nguyen and Nhat Ho (2022). "Amortized projection optimization for sliced Wasserstein generative models". In: *Advances in Neural Information Processing Systems* 35, pp. 36985–36998.

Victor M Panaretos and Yoav Zemel (2019). "Statistical aspects of Wasserstein distances". In: *Annual Review of Statistics and Its Application* 6.1, pp. 405–431.

George Papamakarios, Eric Nalisnick, Danilo Jimenez Rezende, Shakir Mohamed, and Balaji Lakshminarayanan (2021). "Normalizing flows for probabilistic modeling and inference". In: *Journal of Machine Learning Research* 22.57, pp. 1–64.

Jeff M Phillips (2017). "Coresets and sketches". In: *Handbook of Discrete and Computational Geometry*. 3rd. Chapman and Hall/CRC, pp. 1269–1288.

Jeff M. Phillips and Wai Ming Tai (2018). "Improved coresets for kernel density estimates". In: *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. SIAM, pp. 2718–2727. DOI: 10.1137/1.9781611975031.173.

Jeff M. Phillips and Wai Ming Tai (2020). "Near-optimal coresets of kernel density estimates". In: *Discrete & Computational Geometry* 63.4, pp. 867–887. DOI: 10.1007/S00454-019-00134-6.

Juho Piironen and Aki Vehtari (2017a). "On the hyperprior choice for the global shrinkage parameter in the horseshoe prior". In: *Artificial Intelligence and Statistics*. PMLR, pp. 905–913.

Juho Piironen and Aki Vehtari (2017b). "Sparsity information and regularization in the horseshoe and other shrinkage priors". In: *Electronic Journal of Statistics* 11.2, pp. 5018 –5051. DOI: 10.1214/17-EJS1337SI.

Martyn Plummer, Nicky Best, Kate Cowles, and Karen Vines (2006). "CODA: convergence diagnosis and output analysis for MCMC". In: *R News* 6.1, pp. 7–11.

Andrea Dal Pozzolo, Olivier Caelen, Reid A. Johnson, and Gianluca Bontempi (2015). "Calibrating probability with undersampling for unbalanced classification". In: *2015 IEEE Symposium Series on Computational Intelligence*, pp. 159–166. DOI: 10.1109/SSCI.2015.33.

Rindang Bangun Prasetyo, Heri Kuswanto, Nur Iriawan, and Brodjol Sutijo Suprih Ulama (2020). "Binomial regression models with a flexible generalized logit link function". In: *Symmetry* 12.2.

Daryl Pregibon (1981). "Logistic regression diagnostics". In: *The Annals of Statistics* 9.4, pp. 705 –724. DOI: 10.1214/aos/1176345513.

Matias Quiroz, Robert Kohn, Mattias Villani, and Minh-Ngoc Tran (2019). "Speeding up MCMC by efficient data subsampling". In: *Journal of the American Statistical Association* 114.526, pp. 831–843. DOI: 10.1080/01621459.2018.1448827.

Julien Rabin, Gabriel Peyré, Julie Delon, and Marc Bernot (2012). "Wasserstein barycenter and its application to texture mixing". In: *Scale Space and Variational Methods in Computer Vision: Third International Conference (SSVM)*. Springer, pp. 435–446.

Maxim Rabinovich, Elaine Angelino, and Michael I Jordan (2015). "Variational consensus Monte Carlo". In: *Advances in Neural Information Processing Systems (NeurIPS)*. Vol. 28, pp. 1207–1215.

Edward A. Roualdes, Brian Ward, Bob Carpenter, Adrian Seyboldt, and Seth D. Axen (2023). "BridgeStan: Efficient in-memory access to the methods of a Stan model". In: *Journal of Open Source Software* 8.87, p. 5236. DOI: 10.21105/joss.05236.

Mark Rudelson and Roman Vershynin (2007). "Sampling from large matrices: An approach through geometric functional analysis". In: *J. ACM* 54.4, p. 21. DOI: 10.1145/1255443.1255449.

Dominic Schuhmacher, Björn Bähre, Nicolas Bonneel, Carsten Gottschlich, Valentin Hartmann, Florian Heinemann, Bernhard Schmitzer, and Jörn Schrieber (2024). *transport: Computation of optimal transport plans and Wasserstein distances*. R package version 0.15-4.

Oliver Schulz, Frederik Beaujean, Allen Caldwell, Cornelius Grunwald, Vasyl Hafych, Kevin Kröninger, Salvatore La Cagnina, Lars Röhrig, and Lolian Shtembari (2021). "BAT.jl: A Julia-based tool for Bayesian inference". In: *SN Computer Science* 2.3, p. 210. DOI: 10.1007/s42979-021-00626-4.

Steven L. Scott, Alexander W. Blocker, Fernando V. Bonassi, Hugh A. Chipman, Edward I. George, and Robert E. McCulloch (2016). "Bayes and big data: the consensus Monte Carlo algorithm". In: *International Journal of Management Science and Engineering Management* 11.2, pp. 78–88. DOI: 10.1080/17509653.2016.1142191.

M Sklar (1959). "Fonctions de répartition à n dimensions et leurs marges". In: *Annales de l'ISUP*. Vol. 8, pp. 229–231.

Christian Sohler and David P. Woodruff (2011). "Subspace embeddings for the ℓ_1-norm with applications". In: *Proceedings of the Forty-Third Annual ACM Symposium on Theory of Computing (STOC)*, 755–764. DOI: 10.1145/1993636.1993736.

Peter Song (2000). "Multivariate dispersion models generated from Gaussian copula". In: *Scandinavian Journal of Statistics* 27.2, pp. 305–320.

Bharath K Sriperumbudur, Arthur Gretton, Kenji Fukumizu, Bernhard Schölkopf, and Gert RG Lanckriet (2010). "Hilbert space embeddings and metrics on probability measures". In: *The Journal of Machine Learning Research* 11, pp. 1517–1561.

Therese A. Stukel (1988). "Generalized logistic models". In: *Journal of the American Statistical Association* 83.402, pp. 426–431.

M. T. Subbotin (1923). "On the law of frequency of error". In: *Matematicheskij Sbornik* 31.2, pp. 296–301.

Martin A. Tanner and Wing Hung Wong (1987). "The calculation of posterior distributions by data augmentation". In: *Journal of the American Statistical Association* 82.398, pp. 528–540.

Sven Teschke, Katja Ickstadt, and Alexander Munteanu (2024). "Detecting interactions in high-dimensional data using cross leverage scores". In: *Biometrical Journal* 66.8, e70014.

Alexandre Verine, Benjamin Negrevergne, Muni Sreenivas Pydi, and Yann Chevaleyre (2023). "Precision-recall divergence optimization for generative modeling with GANs and normalizing flows". In: *Advances in Neural Information Processing Systems* 36, pp. 32539–32573.

Cédric Villani (2009). *Optimal transport: old and new*. Vol. 338. Springer.

Hadley Wickham and Jennifer Bryan (2023). *R packages*. " O'Reilly Media, Inc."

Christopher Williams and Matthias Seeger (2000). "Using the Nyström method to speed up kernel machines". In: *Advances in neural information processing systems* 13.

Christopher K.I. Williams and Carl Edward Rasmussen (2006). *Gaussian processes for machine learning*. Cambridge, MA: MIT Press.

David Woodruff and Qin Zhang (2013). "Subspace embeddings and ℓ_p-regression using exponential random variables". In: *Proceedings of the 26th Annual Conference on Learning Theory (COLT)*, pp. 546–567.

David P. Woodruff (2014). "Sketching as a tool for numerical linear algebra". In: *Found. Trends Theor. Comput. Sci.* 10.1-2, pp. 1–157. DOI: 10.1561/0400000060.

David P. Woodruff and Taisuke Yasuda (2022). "High-Dimensional Geometric Streaming in Polynomial Space". In: *63rd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2022, Denver, CO, USA, October 31 - November 3, 2022*, pp. 732–743. DOI: 10.1109/FOCS54457.2022.00075.

David P. Woodruff and Taisuke Yasuda (2023). "Online Lewis weight sampling". In: *Proceedings of the 2023 ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pp. 4622–4666.

Yujia Xie, Xiangfeng Wang, Ruijia Wang, and Hongyuan Zha (2020). "A fast proximal point method for computing exact Wasserstein distance". In: *Uncertainty in artificial intelligence*, pp. 433–453.

Jacky Zhang, Rajiv Khanna, Anastasios Kyrillidis, and Sanmi Koyejo (2021). "Bayesian coresets: Revisiting the nonconvex optimization perspective". In: *International Conference on Artificial Intelligence and Statistics*. PMLR, pp. 2782–2790.