# Generative Modelling: Addressing Open Problems in Model Misspecification and Differential Privacy


Sahra Ghalebikesabi
University College
University of Oxford

A thesis submitted for the degree of
*Doctor of Philosophy*
Michaelmas 2023

# Statement of Originality

I hereby declare that except where specific reference is made to the work of others, the intellectual contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification. My personal contributions are detailed in the authorship forms at the end of each chapter. This dissertation is my own work except as specified in the text and authorship forms.

Sahra Ghalebikesabi
Michaelmas 2023

*This thesis is dedicated to*
*my mom*
*for her unwavering source of support*

# Acknowledgements

First and foremost, I am deeply grateful to my supervisor, Chris Holmes, for his constant support and guidance throughout my DPhil journey. His passion for statistics and innovative research inspired me, while his expert mentorship taught me invaluable skills that have shaped my academic and professional pursuits.

Next, I extend my sincere appreciation to my mentors, Javier Gonzalez, Borja Balle and Jack Jewson, for their exceptional guidance during my internships at Microsoft Research Cambridge, DeepMind and Universitat de Pompeu Fabra, respectively. Their mentorship and encouragement played a pivotal role in honing my research abilities and broadening my horizons. I am also deeply thankful to my numerous collaborators whose valuable insights and collaborative efforts have significantly contributed to the success of my research. Their collective expertise and enthusiasm have enriched my work and opened up new avenues for exploration.

I would have not been able to pursue my studies without the generous support of the ESPRC, Novartis, University College Oxford and Microsoft Research for funding my studies.

I would like to acknowledge the outstanding administrative staff at Oxford, whose consistent assistance was instrumental in making my academic experience smooth and enjoyable. Special thanks to the IT team, Beverley Lane and Joanna Stoneham for their exceptional support and responsiveness.

Special thanks to Jonathan, whose unwavering support encouraged me at all times. To Rolf and Burcu, my found family whom I will cherish forever. To Anna, who has been there like an academic sister for me. To Dan, who I have bonded with over long walks and vegan snacks. To Faaiz and Imi, whose daily support was crucial to my success. To Lien and Riri, who have been and always will be my oldest friends. To my newfound colleagues Valentin and Abi, who were there for me in the last few months of the PhD. To Veit and Edwin, both mathematical geniuses, whom I've admired over the years. To Linying, Tiggy, and Kamélia, who have shown me incredible kindness that I will never be able to fully repay. To Aldaïr, Mélodie, and Taylor, all chance encounters who will stay with me as lifelong friends.

Lastly, I am forever indebted to my family for their unwavering love and support, with a special thanks to my mom, who selflessly made this entire journey possible from the very beginning. Your belief in me has been a constant source of motivation, and I am eternally grateful for your sacrifices and encouragement.

# Abstract

Generative modelling has become a popular application of artificial intelligence. Model performance can, however, be impacted negatively when the generative model is misspecified, or when the generative model estimator is modified to adhere to a privacy notion such as differential privacy. In this thesis, we approach generative modelling under model misspecification and differential privacy by presenting four different works.

We first present related work on generative modelling. Subsequently, we delve into the reasons that necessitate an examination of generative modelling under the challenges of model misspecification and differential privacy.

As an initial contribution, we consider generative modelling for density estimation. One way to approach model misspecification is to relax model assumptions. We show that this can also help in nonparametric models. In particular, we study a recently proposed nonparametric quasi-Bayesian density estimator and identify its strong model assumptions as a reason for poor performance in finite data sets. We propose an autoregressive extension relaxing model assumptions to allow for a-priori feature dependencies.

Next, we consider generative modelling for missingness imputation. After categorising current deep generative imputation approaches into the classes of nonignorable missingness models as introduced by Rubin [1976], we extend the formulation of variational autoencoders to factorise according to a nonignorable missingness model class that has not been studied in the deep generative modelling literature before. These explicitly model the missingness mechanisms to prevent model misspecification when missingness is not at random.

Then, we focus the attention of this thesis on improving synthetic data generation under differential privacy. For this purpose, we propose differentially private importance sampling of differentially private synthetic data samples. We observe that importance sampling helps more, the better the generative model is. We next focus on increasing data generation quality by considering differentially private diffusion models. We identify training strategies that significantly improve the performance of DP image generators.

We conclude the dissertation with a discussion, including contributions and limitations of the presented work, and propose potential directions for future work.

# Contents

# 1 Introduction

## Contents



*Generative modelling* estimates the probability distribution of data features (and targets). This is in contrast to discriminative modelling where classification models are used to estimate decision boundaries of targets based on observed features. The term "generative modelling" is hereby derived from the fact that modelling the feature distribution allows us to generate synthetic data points in the data space [Bishop and Nasrabadi, 2006]. When data synthesis is the goal of the modelling approach (as opposed to regression or classification), we specifically refer to the approach as synthetic data generation.

Here, we consider the case of synthetic data generation where we aim to generate an artificial dataset that follows the distribution of a real dataset. The synthetic dataset is desired to carry over all statistical properties and patterns of the real

dataset [Stadler et al., 2021]. This problem has been widely studied [Bond-Taylor et al., 2021, Harshvardhan et al., 2020, Jordon et al., 2022, Nikolenko, 2021, Salakhutdinov, 2015]. A common approach to synthetic data generation is defining a synthetic data generative process (SDGP) from which independent and identically distributed data samples can be sampled. The SDGP is hereby typically modelled after a true data generative process (DGP) observed through a real-world dataset.

Deep generative models are a subclass of generative models that parameterise the generative distribution by deep neural networks. With their advent, there has been ongoing work on variational autoencoders (VAEs) [Kingma and Welling, 2013, Rezende et al., 2014], generative adversarial nets (GANs) [Goodfellow et al., 2014], and diffusion models [Sohl-Dickstein et al., 2015, Song and Ermon, 2019]. These models have been commercialised, among other things, as chat assistants [e.g. OpenAI, 2023, Sundar Pichai, 2023], image and art generators [e.g. Muse AI, 2023, OpenAI, 2022], or music [e.g. Briot et al., 2017] and video [e.g. Kondratyuk et al., 2023] generators.

The model performance of generative models may be impacted when the distributional model is misspecified or the model estimation procedure is altered through differential privacy (DP). Model misspecification refers to the setting where the generative model assumptions of the SDGP do not match the true DGP for any choice of parameters. DP [Dwork, 2006] is a mathematical framework to provide individuals in a training dataset with privacy, i.e. a guarantee that their data won't be leaked from any released statistics. Similarly to model misspecification, it can impact the model performance as the optimal model estimator typically has to be altered with noise to provide privacy guarantees.

In this thesis, we will show how both model misspecification and DP impact the fit of the SDGP. We will then consider approaches that improve the statistical properties of the DGP that the SDGP preserves. These approaches include 1) relaxing the model assumptions of the SDGP (Chapters 3 and 4), 2) importance weighting samples from the SDGP such that the distribution imposed by the

weighted samples is closer to the DGP (Chapter 5), and lastly 3) improving the training algorithm of the SDGP (Chapter 6).

## 1.1 Applications of Generative Modelling

The contributions of this thesis are driven by the paramount importance of accurate generative modelling in a wide range of application domains. As it might be impossible to comprehensively cover the broad range of generative model applications, we focus on three key areas: density estimation, missingness imputation, and synthetic data generation.

### 1.1.1 Density Estimation

In density estimation, the goal is to learn a function $p_\theta : \mathcal{X} \to \mathbb{R}$ where $p_\theta$ is a probability density function if $x \in \mathcal{X}$ is continuous, or a probability mass function if $x \in \mathcal{X}$ is discrete. Given a density estimator, it is also possible to sample synthetic data from the estimated density, although this operation may be computationally intractable.

Note that not all synthetic data generators are suitable for density estimation. For instance, GANs do not model the SDGP explicitly. Similarly, VAEs and diffusion models approximate the model likelihood by a lower bound. These models are thus more appropriate for applications within synthetic data generation.

### 1.1.2 Synthetic Data Generation

As mentioned, within synthetic data generation, generative models are used to generate samples from the SDGP $x \sim p_\theta$. The idea of synthetic data can be traced back to at least Rubin [1978] who advanced synthetic data methods in the spirit of multiple imputation. Since then, synthetic data has been employed in a wide spectrum of applications.

One application is privacy-preserving dataset release. When real-world data contains sensitive information that prevents data holders from sharing it, they can instead release synthetic datasets that emulate key information present in the real

data [Bellovin et al., 2019, Raghunathan, 2021, Stadler et al., 2021]. Releasing synthetic data as a means of valid statistical inference has already been suggested by Rubin [1993], as noted by Raghunathan [2021]. Rubin [1993] proposes to employ synthetic data as a technique for statistical disclosure control, which encompasses the strategies employed by data collectors to safeguard the privacy of individuals in statistical datasets. However, subsequent research has revealed that synthetic data release alone is inadequate for ensuring privacy, and additional measures such as DP are necessary [Carlini et al., 2023, Hayes et al., 2019, Stadler et al., 2021].

Even when privacy concerns are not present, generative models are used for synthetic data generation in the context of chatbots [OpenAI, 2023, Sundar Pichai, 2023], for art generation or photo editing [OpenAI, 2022], or music generation [Briot et al., 2017].

### 1.1.3 Missingness Imputation

One special application of synthetic data generation is probabilistic missingness imputation. Compared to synthetic data generation, where we sample $x \sim p_\theta(x)$ or $x \sim p_\theta(x|y)$ with $x \in \mathbb{R}^J, y \in \mathcal{Y}$, in probabilistic missingness imputation we sample a subset of missing features $\mathcal{S}_\text{m} \subset \{1, ..., J\}$ given a subset of observed features $\mathcal{S}_\text{o} \subset \{1, ..., J\}$, that is $x^{\mathcal{S}_\text{m}} \sim p_\theta(x|x^{\mathcal{S}_\text{o}})$ where $|\mathcal{S}|$ is the size of $\mathcal{S}$, $x^\mathcal{S} \in \mathbb{R}^{|\mathcal{S}|}$ denotes the vector of features $x^j$ where $j \in \mathcal{S}$, and $x^j \in \mathbb{R}$ is the $j^\text{th}$ feature dimension of $x$.

Several generative models, even those trained primarily for unconditional generation, can be effectively repurposed for missingness imputation. The inpainting procedure proposed by Lugmayr et al. [2022a] exemplifies this concept by demonstrating how diffusion models can be adapted for missingness imputation by modifying the sampling procedure. In contrast, Nazabal et al. [2020] propose a modification to the training procedure of VAEs specifically tailored for handling missing data.

## 1.2 Contributions and Thesis Outline

The remainder of this dissertation delves into the following chapters.

**Chapter 2** We provide a comprehensive literature review of various SDGPs. We then shortly discuss model misspecification and DP within generative modelling, and explain how these two artefacts can impact model performance.

**Chapter 3 [Ghalebikesabi et al., 2023b]** The first application domain we consider for generative modelling is density estimation. To address the risk of model misspecification, one approach is to leverage highly expressive models such as nonparametric estimators. To this end, we build upon a recent quasi-Bayesian approach to estimating the predictive distribution of Dirichlet Process Mixture Models (DPMM). We show that, even in DPMMs, the model specification, i.e. choice of likelihood and prior, impacts model performance in finite datasets. To address this limitation, we propose a more complex DPMM formulation with an autoregressive likelihood factorisation and Gaussian Process prior, effectively generalising the assumptions of previous work. We show how to estimate the predictive distribution of this Bayesian model and evaluate the performance of the resulting estimator for density estimation and predictive modelling tasks.

**Chapter 4 [Ghalebikesabi et al., 2021a]** We then continue the thesis by considering a second application of generative modelling, namely missingness imputation. First, we provide a categorisation of state-of-the-art deep learning imputation methods according to the nonignorable missingness model classes as introduced by Rubin [1976]. Current deep learning models typically focus on the setting when missingness is at random, and the missingness mechanism does not need to be modelled explicitly. In real-world applications, however, missingness is typically not at random and it is necessary to model the missingness mechanism to prevent model misspecification. Motivated by this, we thus consider how to extend the model assumptions of VAEs to explicitly include the missingness process, and show how this improves imputation performance.

**Chapter 5 [Ghalebikesabi et al., 2022]** Finally, we focus the attention of this thesis on synthetic data generation. For this purpose, we consider a range of DP generative models, particularly GANs. We show that the performance of downstream estimators—models trained on the DP synthetic data, and evaluated on the real-world data—can be increased by releasing a set of DP importance weights alongside the synthetic data. Nevertheless, importance weighting only provides minimal benefits when the synthetic data distribution is not close to the real data distribution. This calls for improving upon the training procedure of private generators as addressed in Chapter 6.

**Chapter 6 [Ghalebikesabi et al., 2023a]** In this chapter, we show how to significantly outperform state-of-the-art DP image generators by employing a set of useful training techniques for DP diffusion models, including but not limited to pre-training, large batch sizes and a modified timestep sampling distribution. We show that fine-tuning these models on CIFAR-10 and Camelyon17, a histopathological lymph node tissue image dataset, can lead to the generation of images that preserve a set of useful statistical properties.

**Chapter 7** We conclude the thesis with a summary of our results. Notably, we delve into the insights gained from our approaches to improve generative model performance under model misspecification or DP. Eventually, we venture into potential future research directions that stem from our presented results.

## 1.3 An Overview of Research Conducted during the DPhil

To provide a comprehensive overview of the research undertaken during my doctoral studies, this section presents a detailed list of papers included and excluded in this thesis.

### 1.3.1 Work Included in this Thesis

Each chapter of this thesis is based on a paper. The list of these is included in chronological order here for completeness.

1. **Ghalebikesabi, S.**, Holmes, C. C., Fong, E., & Lehmann, B. (2023). Quasi-Bayesian nonparametric density estimation via autoregressive predictive updates. In *Uncertainty in Artificial Intelligence* (pp. 658-668). PMLR. *(Awarded spotlight presentation)* [Ghalebikesabi et al., 2023b]

2. **Ghalebikesabi, S.**, Cornish, R., Holmes, C., & Kelly, L. (2021). Deep generative missingness pattern-set mixture models. In *International Conference on Artificial Intelligence and Statistics* (pp. 3727-3735). PMLR. [Ghalebikesabi et al., 2021a]

3. **Ghalebikesabi, S.**, Wilde, H., Jewson, J., Doucet, A., Vollmer, S., & Holmes, C. (2022). Mitigating statistical bias within differentially private synthetic data. In *Uncertainty in Artificial Intelligence* (pp. 696-705). PMLR. *(Awarded oral presentation)* [Ghalebikesabi et al., 2022]

4. **Ghalebikesabi, S.**, Berrada, L., Gowal, S., Ktena, I., Stanforth, R., Hayes, J., ... & Balle, B. (2023). Differentially private diffusion models generate useful synthetic images. In *International Workshop on Trustworthy Federated Learning in Conjunction with IJCAI 2023*. *(Awarded oral presentation and Best Student Paper Award)* [Ghalebikesabi et al., 2023a]

### 1.3.2 Work Omitted from this Thesis

To keep the thesis unified and concise, I have omitted several works that I have contributed to during my DPhil. A list of these, with a short description, is included in chronological order here for completeness. We then draw connections to the papers included in this thesis.

1. **Ghalebikesabi, S.***, Ter-Minassian, L.*, DiazOrdaz, K., & Holmes, C. C. (2021). On locality of local explanation models. *Advances in neural information processing systems*, 34, 18395-18407. [Ghalebikesabi et al., 2021b]

2. Ter-Minassian, L.*, **Ghalebikesabi, S.***, Diaz-Ordaz, K., & Holmes, C. (2023). Challenges and Opportunities of Shapley Values in a Clinical Context. *ICML 2022 Workshop on Interpretable Machine Learning in Healthcare*. [Ter-Minassian et al., 2022]

3. Pradier, M. F., Prasad, N., Chapfuwa, P., **Ghalebikesabi, S.**, Ilse, M., Woodhouse, S., ... & Meeds, E. (2023). AIRIVA: A Deep Generative Model of Adaptive Immune Repertoires. *Machine Learning for Healthcare* 2023. [Pradier et al., 2023]

4. Jewson, J.*, **Ghalebikesabi, S.***, & Holmes, C. (2023). Differentially Private Statistical Inference through $\beta$-Divergence One Posterior Sampling. In *Advances in neural information processing systems*, 36. [Jewson et al., 2023]:

5. Wild, V. D., **Ghalebikesabi, S.**, Sejdinovic, D., & Knoblauch, J. (2023). A Rigorous Link between Deep Ensembles and (Variational) Bayesian Methods. In *Advances in neural information processing systems*, 36. *(Awarded oral presentation)* [Wild et al., 2023]

In [Ghalebikesabi et al., 2021b, Ter-Minassian et al., 2022], we build upon insights from our previous work on model misspecification within missingness imputation [Ghalebikesabi et al., 2021a] and apply it within model interpretability. In particular, we consider Kernel SHAP, a popular tool for estimating post-hoc feature attributions within black-box models [Lundberg and Lee, 2017]. Kernel SHAP approximates Shapley values, which are feature removal-based explainability values where the removal (or missingness) of features is typically modelled via imputation methods [Covert et al., 2021]. Similarly to [Ghalebikesabi et al., 2021a], where we analyse how

*denotes equal contribution

to incorporate assumptions on the missingness mechanism in the generative model formulation, we here study how different modelling assumptions of the missingness mechanism affect the imputation approach, and thus the interpretation of the Shapley values [Ghalebikesabi et al., 2021b, Ter-Minassian et al., 2022].

Motivated by the vast opportunities of synthetic data generators, also outside of missingness imputation, we [Pradier et al., 2023] propose a domain-specific VAE architecture for immunomics. In particular, we design it to learn a compositional representation of T-cell receptoires to disentangle systematic effects induced by population genetics and sequencing depth. Similarly to the work presented in this thesis [Ghalebikesabi et al., 2021a], we manipulate the generative process within the standard formulation of VAEs to accurately reflect our prior assumptions on biological connections.

In another work [Jewson et al., 2023], we highlight the restrictive nature of the modelling assumptions commonly employed in DP Bayesian posterior sampling. This limitation hinders the widespread applicability of the DP posterior sampling methods proposed by Foulds et al. [2016], Minami et al. [2016], Wang et al. [2015]. To address this challenge, we first connect insights from handling model misspecification within Bayesian inference to the context of DP. Building upon these findings, we then transfer an approach proposed for Bayesian inference under model misspecification to DP posterior sampling. Our method, based on generalised Bayesian updating, relaxes previous assumptions on the model likelihood, expanding the applicability of DP posterior sampling in various settings.

Lastly, in [Wild et al., 2023], we connect deep ensembles to Bayesian methods by reformulating the typical deep learning optimisation problem in the space of probability measures. In particular, we connect several heuristic and sampling approaches (and develop a new sampling approach) by studying the generalised variational inference optimisation problem with the help of the Wasserstein gradient flow.

# 2

# Background and Literature Review

## Contents



This chapter delves into the foundational literature that underpins the work presented in this dissertation and provides the necessary background to understand it. We begin the review with an overview of generative modelling methods before we focus on model misspecification and DP.

## 2.1 Generative Modelling

To denote a labelled dataset, we write $\mathcal{D} = \{x_n, y_n\}_{n=1}^N$, $x \in \mathcal{X} \subseteq \mathbb{R}^J, y \in \mathcal{Y}$ with targets $y$ and features $x = (x^1, \ldots, x^J), x^j \in \mathbb{R}$ where the superscript denotes

the dimension of the vector. We let $(x_n, y_n) \overset{\text{iid}}{\sim} p_G$ be sampled from the DGP $p_G$ for $n \in \{1, \ldots, N\}$. We denote by $\theta \in \Theta$ the parameters of the synthetic data generative distribution $p_{D,\theta}$. For simplicity, we drop the subscripts of the distributions from now on unless required for clarity.

While predictive modelling targets the prediction of a target variable $y$ conditional on feature variables $x$, i.e. $p(y|x)$, generative modelling for synthetic data generation is typically defined as modelling only the features $p(x)$, the joint $p(x, y)$ or the conditional of the features given the targets $p(x|y)$. We will from now on focus on modelling the marginal $p(x)$ given dataset $\mathcal{D} = \{x_n\}_{n=1}^N$, and will mention differences to the other cases if required.

In the remainder of this section, we will review four different generative modelling approaches. While other generative models exist, such as normalising flows [Kobyzev et al., 2020, Rezende and Mohamed, 2015, Tabak and Turner, 2013, Tabak and Vanden-Eijnden, 2010] or deep Boltzmann machines [Salakhutdinov and Hinton, 2009], we do not consider these in this thesis and refer the interested reader to available reviews [Bond-Taylor et al., 2021, Raghunathan, 2021, Ruthotto and Haber, 2021, Salakhutdinov, 2015].

### 2.1.1 Dirichlet Process Mixture Models

Probabilistic models can be either parametric or nonparametric. The choice of parametric approaches with a fixed and finite number of parameters can introduce over- or underfitting issues when the model complexity does not fit the available dataset. In contrast, Bayesian nonparametric approaches provide a remedy by offering unbounded model complexity. We will now focus on a particular class of nonparameteric models. According to Bayes' law, for a likelihood function $p_\theta$ and prior $\pi(\theta)$ on the model parameters $\theta$, the *Bayesian posterior* $\pi_N(\theta)$ after observing data samples $\{x_n\}_{n=1}^N$ is given as

$$\pi_N(\theta) = \frac{p_\theta(x_1, ..., x_n)\pi(\theta)}{\int_\Theta p_\theta(x_1, ..., x_n)\pi(\theta)d\theta}. \tag{2.1}$$

One of the most popular nonparametric Bayesian models is the Dirichlet Process Mixture Model (DPMM). The DPMM [Escobar, 1988, Escobar and West, 1995] can be written as

$$p(x|G) = \int_{\Theta} \mathcal{K}(x|\theta)\, dG(\theta), \text{ with } G \sim \mathrm{DirP}(c, G_0), \tag{2.2}$$

where $\theta \in \Theta = \mathbb{R}^K$ are parameter vectors, the prior $\mathrm{DirP}(c, G_0)$ assigned to $G$ is a Dirichlet process prior with base measure $G_0$ and concentration parameter $c > 0$ [Ferguson, 1973], and $\mathcal{K}(x|\theta)$ is a user-specified kernel. While sampling from the DPMM posterior is possible, approximating the posterior, however, is computationally intensive because of the intractable denominator in equation (2.1) [Teh et al., 2010]. Instead, we consider the estimation of the *Bayesian predictive density* $p_n$ after having observed $n$ observation points,

$$p_n(x) = \int p_\theta(x)\pi_n(\theta)d\theta. \tag{2.3}$$

Given a density estimator, it is then possible to sample from the SDGP with Monte Carlo approaches in low-dimensional settings [Robert et al., 1999]. The contribution of this thesis will focus on a computationally efficient iterative estimation procedure for the predictive density. For this purpose, we will now briefly recap predictive density estimation via bivariate copula updates.

#### 2.1.1.1 Univariate Predictive Density Updates via Bivariate Copulas

Hahn et al. [2018] have introduced iterative Bayesian predictive updating via bivariate copulas for one-dimensional feature spaces. A *copula* is a multivariate cumulative distribution function (CDF) $C : [0, 1]^J \to [0, 1]$ with uniform marginal distributions. Copulas are used to characterise the dependence between multiple random variables independently of their marginals. They have thus been a popular tool within density estimation [e.g. Bayestehtashk and Shafran, 2013, Charpentier et al., 2007, Nagler and Czado, 2016].

Hahn et al. [2018] propose to decompose the predictive density $p_n(x)$ for $x \in \mathbb{R}$ and some bivariate function $h_n$ in (2.3) iteratively as

$$p_n(x) = p_{n-1}(x) h_n(x, x_n), \tag{2.4}$$

$$h_n(x, x_n) \overset{(a)}{=} \frac{p_n(x)}{p_{n-1}(x)} \overset{(b)}{=} \frac{p_{n-1}(x|x_n)}{p_{n-1}(x)} \overset{(c)}{=} \frac{p_{n-1}(x, x_n)}{p_{n-1}(x) p_{n-1}(x_n)}, \tag{2.5}$$

where (a) follows from the factorisation assumption in (2.4), (b) holds by definition, and (c) holds by Bayes' law. According to (2.5), we have that

$$h_n(x, x_n) = \frac{\int p(x|\theta) p(x_n|\theta) \pi_{n-1}(\theta) d\theta}{\int p(x|\theta) \pi_{n-1}(\theta) d\theta \int p(x_n|\theta) \pi_{n-1}(\theta) d\theta}. \tag{2.6}$$

The existence and form of $h_n(x, x_n)$ can be derived from Sklar's theorem [Sklar, 1959]:

**Theorem 1** (Sklar's theorem [Sklar, 1959])**.** *For any bivariate density $p(x_1, x_2)$ with continuous marginal CDFs, $P_1(x_1)$ and $P_2(x_2)$, and marginal densities, $p_1(x_1)$ and $p_2(x_2)$, there exists a unique bivariate copula $C$ with density $c$ such that*

$$p(x_1, x_2) = c\left\{P_1(x_1), P_2(x_2)\right\} p_1(x_1) p_2(x_2).$$

According to (2.4), it follows that

$$h_n(x, x_n) = c_n\{P_{n-1}(x), P_{n-1}(x_n)\},$$

where $P_{n-1}(x)$ denotes the CDF of $p_{n-1}(x)$.

The computation of $h_n(x, x_n)$ according to (2.6) is intractable. It is thus not possible to compute (2.4) for fully Bayesian models. Thus, Hahn et al. [2018] consider sequences of $h_n$ that match the Bayesian model in the first update step for $n = 1$, but not for any other update steps $n > 1$. This update no longer corresponds to a Bayesian model. For *conditionally identically distributed** [Berti et al., 2004] copula updates, the updated densities still exhibit desirable Bayesian characteristics such as coherence and regularisation, and are hence referred to as *quasi-Bayes* [Berti et al., 2004, Fortini and Petrone, 2020, Smith and Makov, 1978].

*The sequence $x_1, x_2, \ldots$ is conditionally identically distributed if we have $P(X_{n+k} \leq x | x_{1:n}) = P_n(x), \forall k \in \mathbb{N}$ almost surely for each $x \in \mathbb{R}^J$.

#### 2.1.1.2 Multivariate Predictive Density Updates

For multivariate $x \in \mathbb{R}^J$ with $J > 1$, the above derivations do not hold. Fong et al. [2021] instead propose to solve (2.5) explicitly for $n = 1$ for a pre-defined likelihood model and prior, and apply the derived update for $n > 1$. For convenient specifications of the Bayesian model, the predictive updates still exhibit bivariate copulas as building blocks.

In particular, Fong et al. [2021] consider DPMMs, as defined in (2.2) with base measure $G_0 = \text{Normal}(0, \tau^{-1} I_J)$ for some precision parameter $\tau \in \mathbb{R}_{>0}$, and factorised kernel $\mathcal{K}(x|\theta) = \text{Normal}(x|\theta, I_J)$ where $I_J$ is the $J$-dimensional identity matrix. Fong et al. [2021] then derive the following recursive predictive density update $p_n(x) = h_n(x, x_n) p_{n-1}(x)$ for which the first $j \in \{1, \ldots, J\}$ marginals take on the form

$$\begin{aligned} \frac{p_n(x^{1:j})}{p_{n-1}(x^{1:j})} &= 1 - \alpha_n + \alpha_n \prod_{j'=1}^{j} c\left(u_{n-1}^{j'}(x^{j'}), v_{n-1}^{j'}; \rho_0\right), \\ u_{n-1}^{j}(x^j) &:= P_{n-1}\left(x^j \mid x^{1:j-1}\right), v_{n-1}^{j} := P_{n-1}\left(x_n^j \mid x_n^{1:j-1}\right), \end{aligned} \tag{2.7}$$

where $c(u, v; \rho_0)$ is the bivariate Gaussian copula density with correlation $\rho_0 = 1/(1+\tau)$, $p_0$ can be any chosen prior density, and $\alpha_n = \left(2 - \frac{1}{n}\right) \frac{1}{n+1}$.

We note that the assumption of a factorised kernel form impacts the performance of the density estimator for structured data and also influences the form and modelling capacity of the corresponding copula update for finite datasets. As a remedy, we propose a factorisation with weaker assumptions as part of this thesis.

While the derived sampling scheme provides quick recursive predictive updates, synthetic data generation via Monte Carlo sampling based on density estimates does not scale well to high dimensional datasets. Approximation techniques like variational inference provide better scalability properties by simplifying model assumptions. We will thus now consider a special class of models trained with variational inference.

### 2.1.2 Variational Autoencoder

Variational Bayesian methods are a class of techniques used to approximate intractable Bayesian posterior distributions. In this subsection, we consider the estimation of the conditional $p_\theta(x|z)$ given latent variable $z$ instead of $p(x|\theta)$ as in the previous subsection. Variational methods approximate the conditional distribution $p_\theta(z|x)$ by fitting a pre-defined family of distributions over the latent variables, $q_\eta(z)$, with variational parameters $\eta$. Typically, this is done by minimising a dissimilarity function between $q_\eta(z)$ and $p_\theta(z|x)$ such as the Kullback-Leibler (KL) divergence

$$D_{KL}(q_\eta|p_\theta) = \int q_\eta(z) \log \frac{q_\eta(z)}{p_\theta(z|x)} dz.$$

We can then approximately sample synthetic observations from $p_\theta(x)$ by sampling $z \sim q_\eta(z)$ and $x \sim p_\theta(x|z)$.

VAEs [Kingma and Welling, 2013, Rezende et al., 2014] have emerged as a powerful and versatile tool for generative modelling via variational inference as they leverage the complex modelling capabilities of neural networks. Unlike traditional autoencoders [Bank et al., 2023] that focus solely on data reconstruction, VAEs are designed to learn the underlying generative process probabilistically by mapping the data to and from a meaningful latent representation. As such they cannot only be used for interpretable data compression [e.g. Joy et al., 2020], but also synthetic data generation [e.g. Kingma and Welling, 2013].

VAEs consist of two separate neural networks, namely an encoder and a decoder. During inference, a latent representation $z \sim q_\eta(z)$ is sampled, and decoded by sampling from $p_\theta(x|z)$. During training, the encoder takes an input data sample $x \in \mathbb{R}^J$ and maps it to the parameters $\eta(x)$ of the variational distribution $q_\eta(z|x)$—typically the mean and diagonal covariance matrix of a multivariate normal distribution. The decoder then takes a sample from the latent space $z \sim q_\eta(z|x)$ and maps it back to the data space, $p_\theta(x|z)$. The output distribution $p_\theta(x|z)$ is hereby parameterised by a differentiable network.

To train VAEs, Kingma and Welling [2013] propose to not maximise the intractable log-likelihood $\log p_\theta(x)$ directly, but to instead target the expected

lower bound which dissects into one term targeting the decoder and encoder, and one term targeting only the encoder:

$$\log p_\theta(x) \geq \mathop{\mathbb{E}}_{q_\eta(z|x)} \left[ \log \frac{p_\theta(x,z)}{q_\eta(z|x)} \right] = \mathop{\mathbb{E}}_{q_\eta(z|x)} \left[ \log p_\theta(x|z) \right] - D_{KL} \left( q_\eta(z|x) | p(z) \right) \cdot$$

The data reconstruction performance of VAEs can suffer under the prior regularisation term, $D_{KL}\left(q_\eta(z|x)|p_\theta(z)\right)$, if the prior $p_\theta(z)$ is too simple. As a remedy, some papers [Connor et al., 2021] have successfully introduced additional complexity in the latent space. One such instance is hierarchical VAEs [Kingma et al., 2016, Sønderby et al., 2016] that extend traditional VAEs by choosing a hierarchical factorisation of the latent space. In Markovian hierarchical VAEs with $T \in \mathbb{N}$ hierarchical latents [Luo, 2022, Williams et al., 2020], the generative process is defined as

$$p(x, z_{1:T}) = p(z_T) p_\theta(x|z_1) \prod_{t=2}^{T} p_\theta(z_{t-1}|z_t).$$

This factorisation has been shown to improve the performance of VAEs [Sønderby et al., 2016, Williams et al., 2020], and has as such encouraged further research into scaling up the hierarchical dependencies.

### 2.1.3 Denoising Diffusion Models

If we restrict the size of the latent variables within hierarchical VAEs to be the same as the input dimension, the resulting architecture becomes part of a model family called denoising diffusion models. Denoising diffusion models [Ho et al., 2020, Sohl-Dickstein et al., 2015, Song et al., 2020] are a class of likelihood-based generative models that recently established state-of-the-art results across diverse domains, including computer vision [Dhariwal and Nichol, 2021, Lugmayr et al., 2022b, Rombach et al., 2022], audio [Kong et al., 2020], and molecular dynamics [Watson et al., 2023].

Within diffusion models, the latent encoder is defined by a forward Markov chain that sequentially perturbs a real data sample $x_0$ to a pure noise sample $x_T$. Compared to hierarchical VAEs, the encoder is in general not learned, but fixed as a linear Gaussian kernel $q(x_t|x_{t-1})$ before training. Diffusion models instead learn

the decoder by parameterising the transition kernel of a backward Markov chain with deep neural networks to denoise $x_T$ sequentially back to $x_0$. The hierarchical level $t$ is hereby referred to as the timestep of the Markov chain.

At inference time, the decoder can be used to generate new data samples as in VAEs. For this purpose, we first sample an observation $x_T$ from the pure noise distribution, and then gradually denoise it $x_{T-1}, x_{T-2}, ...$ using the learned transition kernel until $x_0$ is obtained. To do so, Ho et al. [2020] propose to parameterise the denoising function $\epsilon_\theta(x_t, t)$ which predicts the noise component $\epsilon$ of a noisy sample $x_t$ given time step $t$ by a neural network.

For computational reasons, Ho et al. [2020] propose to learn the parameters of the denoiser via minimisation of a simplified loss function:

$$\mathcal{L}(\theta|x_0) = \mathbb{E}_{t,x_t,\epsilon}[\|\epsilon - \epsilon_\theta(x_t, t)\|^2]$$

with $t \sim \text{Uniform}\{0, T\}$, where $T$ is the pre-specified maximum timestep (i.e. the level of chained latent variables), and $\text{Uniform}\{a, b\}$ is the discrete uniform distribution bounded by $a$ and $b$. The noisy sample $x_t$ is defined by $x_t = \sqrt{\overline{\alpha}_t}x_0 + \sqrt{1 - \overline{\alpha}_t}\epsilon$ where $\epsilon \sim \text{Normal}(0, I_J)$ is a noise sample of the same dimensionality as the data sample $x_0$, and $\overline{\alpha}_t$ is defined such that $x_t$ follows the pre-specified forward process. We refer the interested reader to Ho et al. [2020] for further details.

While diffusion models have demonstrated state-of-the-art data generation quality, they are inherently limited by slow inference times [Xiao et al., 2021]. When the ability to sample quickly is important, opting for a different generative model class, such as GANs, could offer a solution.

### 2.1.4 Generative Adversarial Nets

GANs [Goodfellow et al., 2014] emerged as versatile tools for synthesising diverse and realistic data. Before diffusion models, they achieved state-of-the-art performance on many image generation tasks [Brock et al., 2018, Zhang et al., 2018]. Unlike the generative models we presented so far, GANs do not rely on explicit probability

distributional assumptions, but they instead employ an adversarial framework to learn complex data distributions implicitly.

In particular, GANs consist of two different deep neural networks: a generator $g_{\theta_{\text{gen}}}(x)$ with parameters $\theta_{\text{gen}}$ and a discriminator $g_{\theta_{\text{disc}}}(x)$ with parameters $\theta_{\text{disc}}$. On the one hand, the generator maps samples $z \sim p(z)$ to $\mathcal{X}$, learning the SDGP. On the other hand, the discriminator $d(x)$ is trained to differentiate between samples from the SDGP, as generated by the generator, and the training data observations following $p_G(x)$. It does so by emitting the probability, $p_{d,\theta_{\text{disc}}}(x)$, that $x$ is a real training data sample rather than a fake sample drawn from the generator $g(x)$ with associated probability distribution $p_{g,\theta_{\text{gen}}}(x)$. The training of GANs then typically consists of the optimisation of a two-player minimax game

$$\min_{\theta_{\text{gen}}} \max_{\theta_{\text{disc}}} \mathcal{L}(\theta_{\text{gen}}, \theta_{\text{disc}}|x) = \mathop{\mathbb{E}}_{p_G(x)} \left[\log p_{d,\theta_{\text{disc}}}(x)\right] + \mathop{\mathbb{E}}_{p_{g,\theta_{\text{gen}}}(x)} \left[\log(1 - p_{d,\theta_{\text{disc}}}(x))\right].$$

As this optimisation function is not convex and high-dimensional, GANs are hard to train and might either fail to converge or suffer from mode collapse where the generator always emits the same synthetic data sample [Salimans et al., 2016].

GANs, VAEs and diffusion models have often been compared and used together to circumvent their respective limitations. Rombach et al. [2022], for instance, leverage VAEs and GANs to map high-resolution images into a compressed latent space. Within the latent space, they then train diffusion models that train faster than if trained on the original feature space. Hu et al. [2017] formally show that sample generation from GANs can be formulated as performing posterior inference and that both GANs and VAEs then involve minimising the KL divergence of posterior and variational distributions. This insight allows them to leverage techniques shown to improve the model performance of one generative class to the other generative model class.

### 2.1.5 Measuring Model Performance

The goal of this thesis is, in informal terms, to improve the fit of the SDGP to the DGP. Defining and measuring the extent to which this goal is reached is harder

than for regression models as the ground truth is typically unknown. Indeed, Theis et al. [2015] have shown that some evaluation metrics can even be contradictory to each other when the data is high-dimensional.

For the purpose of this thesis, we limit ourselves to metrics that assess the similarity between DGP and SDGP. Hence, we here only mention those evaluation metrics that are used throughout the different chapters. This overview is neither meant to be exhaustive nor to provide mathematical definitions. We refer the interested reader to Chapters 3–6 and to Alqahtani et al. [2019], Borji [2019], Theis et al. [2015] for more details.

When $p_G(x)$ is modelled explicitly, we can assess the log-likelihood of a test dataset given the generative model [e.g. Ghalebikesabi et al., 2023b, Kingma et al., 2016, Larochelle and Murray, 2011]. While such an evaluation metric is especially useful to assess density estimators, it can however not be used to assess the model performance across different model classes if these include, for instance, GANs since GANs typically do not parameterise the output distribution.

The performance can then instead be measured empirically by assessing the distributional overlap between DGP and SDGP. Within image generation, metrics include the distributional measures within a latent space of the images, i.e. Fréchet Inception Distance [Heusel et al., 2017], or the Inception score [Salimans et al., 2016]. Different measures mostly differ by the choice of distance metric and the construction of the latent space [Baraheem et al., 2023].

Another way of assessing generative models is visual inspection. Within low-dimensional density estimation, for instance, density estimates are plotted on top of scatter plots [e.g. Chen, 2017, Fong et al., 2021, Ghalebikesabi et al., 2023b]. When generative models serve realistic image generation, practitioners often evaluate them by visually comparing samples from the SDGP and DGP.

In the special case of missingness imputation, test datasets can be constructed by artificially masking out some features in real-world samples [e.g. Ghalebikesabi et al., 2021a, Ipsen et al., 2020, Stekhoven and Bühlmann, 2012]. The masking can hereby follow different missingness mechanisms. Ultimately, the ground truth,

i.e. the true values of the missing features, is known, and the distance between imputed and real values serves as an evaluation metric. When it is computationally feasible, we can also evaluate the log-likelihood that the model assigns to the ground truth of the masked features.

Alternatively, we can assess the performance of downstream models trained on samples from the SDGP [e.g. Ghalebikesabi et al., 2022, 2023a, Stadler et al., 2021, Torkzadehmahani et al., 2019]. In the cases of missingness imputation or synthetic data generation, a possible goal of practitioners might be to estimate downstream classifiers or regressors on the imputed or generated data. In this case, a desirable property of the SDGP is for the predictors to not decrease in performance on real test data no matter whether the predictors were fitted on synthetic or real data [Azizi et al., 2023]. Given the tools to measure the performance of generative models, we can now assess how generative models are impacted by modelling artefacts such as model misspecification or DP.

## 2.2 Model Misspecification and Differential Privacy

As outlined before, we consider how to use generative models to capture the true DGP. The SDGP can, however, deviate from the DGP for several reasons. Here, we consider two such possibilities, namely 1) model misspecification, and 2) DP.

### 2.2.1 Model Misspecification

Under model misspecification, we understand the setting when the SDGP model family does not contain the DGP for any choice of parameters. Our definition of model misspecification draws inspiration from White's [1982] analysis of model misspecification under standard maximum likelihood estimation. White [1982, p. 1] define model misspecification as the violation of the condition that "the stochastic law that determines the behaviour of the phenomena investigated (the "true" structure) is known to lie within the specified parametric family of probability distributions (the model)". Maximum likelihood estimation under model misspecification was earlier

considered by Berk [1966, 1970], Huber [1963], Huber et al. [1967], Tukey [1960] under different terminology. While Pearson [1931] might have been the first to analyse statistical estimation under deviations of the probability distribution from standard model assumptions [Davies and Gather, 2012], Tukey [1960] has been considered the driving force to identify the sensitivity of conventional statistical procedures to minor deviations from distributional assumptions as an important problem [Davies and Gather, 2012, Huber, 2002]. Building upon results by Tukey [1960], Huber [1963] offers the first systematic investigation of model robustness [Davies and Gather, 2012, Huber, 2004] leading to the proposal of M-estimation. To deal with model misspecification, Tukey [1960] and Huber [1963] propose robust mean estimators that offer consistency properties even when the model is misspecified. Later works [Hampel et al., 2011, Huber et al., 1967] further shaped the theory of robust estimation. Model misspecification has been studied in regression analysis [e.g. Begg and Lagakos, 1990, Florax and Nijkamp, 2003, Rao, 1971], Bayesian inference [Bissiri et al., 2016, Medina et al., 2022, Nott et al., 2023, Schmitt et al., 2021], causal inference [e.g. Lenis et al., 2018, Vansteelandt et al., 2012, Waernbaum, 2012] and synthetic data generation [e.g. Roskams-Hieter et al., 2023, Wilde et al., 2021].

Even when the SDGP is not misspecified, the learned parameterisation of the SDGP may be sub-optimal. This can, for instance, stem from non-converging training algorithms. Reasons for non-convergence include limited compute, unsuitable hyperparameters, or also DP.

### 2.2.2 Differential Privacy

DP is a formal privacy notion that limits the extent to which releasing the output of a randomised algorithm $A$ compromises the privacy of a single data observation. Notably, the randomness does not stem from the fixed training dataset. In statistics and machine learning, the randomised algorithm is typically a statistical estimator where the randomisation might have been artificially introduced, such as a noised logistic regression estimator. The output of the randomised algorithm is then the statistical estimate. In particular, DP states the following:

**Definition 1** (Differential Privacy [Dwork et al., 2006]). *Let $A$ be a randomised algorithm, and let $\varepsilon > 0$, $\delta \in [0, 1]$. We say that $A$ is $(\varepsilon, \delta)$-DP if for any two neighbouring datasets $\mathcal{D}, \mathcal{D}'$ differing by a single element and $\mathcal{S}$ denoting the support of $A$, we have that*

$$\forall\, S \subset \mathcal{S},\ \mathbb{P}[A(\mathcal{D}) \in S] \leq \exp(\varepsilon)\mathbb{P}[A(\mathcal{D}') \in S] + \delta.$$

As we see, the privacy guarantee is controlled by the so-called privacy budget $(\varepsilon, \delta)$. While $\varepsilon$ bounds the log-likelihood ratio of any set of possible outputs that can be obtained when running $A$ on two neighbouring datasets, $\delta$ bounds the occurrence of infrequent outputs that violate this bound. It is typically chosen to be smaller than $1/N$ for training datasets of size $N$ to prevent the leakage of any single training example. The smaller $\varepsilon$ and $\delta$ get, the stronger the privacy guarantee which explains the term "privacy budget".

A popular general-use DP algorithm is the exponential mechanism [McSherry and Talwar, 2007] according to which sampling $\theta$ with probability proportional to $\exp(-\epsilon\ell(\mathcal{D}, \theta)/(2S(h))$ for loss $\ell(\mathcal{D}, \theta) \in \mathbb{R}$ is $(\epsilon, 0)$-DP. The exponential mechanism subsumes many DP approaches, such as the widely used sensitivity method [Dwork et al., 2006] which adds Laplace noise with scale inversely proportional to the sensitivity of the estimator. The sensitivity is hereby defined as the maximum extent to which the statistical estimate can change between neighbouring datasets. For example, consider empirical risk minimisation for $\theta \in \Theta \subseteq \mathbb{R}^K$:

$$\hat{\theta} := \underset{\theta \in \Theta}{\operatorname{argmin}} \frac{1}{N} \sum_{n=1}^{N} \ell(x_n, y_n, f_\theta) + \lambda R(\theta), \tag{2.8}$$

where $\ell(x_n, y_n, f_\theta)$ is the loss function given parameterisation $f_\theta$, $R(\theta)$ is a regulariser, and $\lambda > 0$ is the regularisation weight. Chaudhuri et al. [2011] then show that

$$\tilde{\theta} = \hat{\theta}(D) + z, \text{ with } \mathbb{R}^K \ni z = (z_1, \ldots, z_K) \overset{\text{iid}}{\sim} \mathcal{L}\left(0, \frac{2}{n\lambda\epsilon}\right),$$

where $\mathcal{L}(\mu, s)$ is a Laplace distribution with density $f(z) = \frac{1}{2s}\exp\{-|z - \mu|/s\}$, is $(\epsilon, 0)$-DP under some convexity and differentiability conditions.

Relaxing the convexity assumptions, differentially private stochastic gradient descent (DPSGD) [Abadi et al., 2016] has been established as a widely applied DP

gradient descent procedure [see De et al., 2022, McMahan et al., 2018]. To achieve bounded sensitivity, DPSGD clips every single gradient within the stochastic gradient descent update step before averaging the clipped gradients within the batch. In line with the sensitivity method, it then adds noise to the averaged batch gradients.

Due to its flexibility, DPSGD has been widely applied to privatise deep classification networks [e.g. De et al., 2022, Ponomareva et al., 2023] and also deep generative models [e.g. Dockhorn et al., 2022, Ghalebikesabi et al., 2022, 2023a, Torkzadehmahani et al., 2019, Xie et al., 2018].

The latter, in particular, represents a significant area of study as research has demonstrated that deep generative models tend to memorise their training data [Carlini et al., 2023, Hayes et al., 2019]. DP training mitigates such memorisation [Hayes et al., 2023] and thus emerges as a potential solution. This makes DP generative modelling an interesting topic to study, especially in the context of synthetic data release. Bellovin et al. [2019], for instance, propose that synthetic data generators in sensitive domains, such as healthcare, should be trained with DP for enhanced privacy properties. Additionally, DP or similar notions [Vyas et al., 2023] hold the potential for training on copyright-protected data [Amiri et al., 2023].

Achieving strong privacy guarantees typically comes at the expense of reduced model performance. This trade-off between privacy and model performance has been studied exhaustively [Amin et al., 2019, Barber and Duchi, 2014, Bun et al., 2014, Duchi et al., 2013, Dwork et al., 2015, Kamath et al., 2020, 2023, Karwa and Vadhan, 2017, Stadler et al., 2021, Zhu et al., 2021, 2023]. In particular, Kamath et al. [2023] show for the estimation of the DP empirical average that no algorithm can have low bias, low variance and low privacy loss for any arbitrary distribution at the same time. What is more, they prove that unbiased mean estimation is impossible for exponential families. As a result, in practice, biased estimates are tolerated.

### 2.2.3 Considering Model Misspecification and Differential Privacy Together

As discussed, both model misspecification and DP can negatively impact model performance. In some cases, techniques developed under one setting can then also be used to improve model performance under the other setting. We will illustrate two such examples in the following.

First, we consider sampling from the Gibbs posterior under generalised Bayes for $w > 0$ [Bissiri et al., 2016, Jiang and Tanner, 2008, Zhang, 2006]:

$$\pi(\theta|x_1, \ldots, x_N) \propto \pi(\theta) \exp\left\{ w \sum_{n=1}^{N} \log p(x_n; \theta) \right\}.$$

On the one hand, choosing $0 < w < 1$ has been shown to improve model performance under model misspecification. The resulting posterior is then less concentrated and the model exhibits a higher posterior variance [Jiang and Tanner, 2008]. On the other hand, Wang et al. [2015] have shown that choosing $w$ can provide DP estimation by sampling from the Bayesian posterior as $w$ allows to adjust the sensitivity of the posterior samples to training examples. For more details, please see [Jewson et al., 2023] where we study the connection between Bayesian inference under model misspecification and DP in more detail.

Second, we consider the use of importance weighting [Hammersley and Morton, 1954]. Importance weighting estimates expectations under a target distribution with samples from a proposal distribution and the corresponding log-likelihood ratios of the data samples. Due to its general assumptions, importance weighting can be applied in diverse model settings. We thus use it to tackle model misspecification in Chapter 4 and to improve model performance under DP in Chapter 5.

While these two examples do not constitute a rigorous link between model misspecification and DP, they illustrate that there exists some overlap between approaches for generative modelling under these two settings.

# 3

# Quasi-Bayesian Nonparametric Density Estimation via Autoregressive Predictive Updates

# Quasi-Bayesian Nonparametric Density Estimation via Autoregressive Predictive Updates

**Sahra Ghalebikesabi**[1] **Chris Holmes**[2] **Edwin Fong**[*2] **Brieuc Lehmann**[*3]

[1]University of Oxford
[2]Novo Nordisk
[3]University College London

## Abstract

Bayesian methods are a popular choice for statistical inference in small-data regimes due to the regularization effect induced by the prior. In the context of density estimation, the standard nonparametric Bayesian approach is to target the posterior predictive of the Dirichlet process mixture model. In general, direct estimation of the posterior predictive is intractable and so methods typically resort to approximating the posterior distribution as an intermediate step. The recent development of quasi-Bayesian predictive copula updates, however, has made it possible to perform tractable predictive density estimation without the need for posterior approximation. Although these estimators are computationally appealing, they struggle on non-smooth data distributions. This is due to the comparatively restrictive form of the likelihood models from which the proposed copula updates were derived. To address this shortcoming, we consider a Bayesian nonparametric model with an autoregressive likelihood decomposition and a Gaussian process prior. While the predictive update of such a model is typically intractable, we derive a quasi-Bayesian update that achieves state-of-the-art results in small-data regimes.

## 1 INTRODUCTION

Modelling the joint distribution of multivariate random variables with density estimators is a central topic in modern unsupervised machine learning research [Durkan et al., 2019, Papamakarios et al., 2017]. As well as providing insight into the statistical properties of the data, density estimates are used in a number of downstream applications, including image restoration [Zoran and Weiss, 2011], density-based clustering [Scaldelai et al., 2022], and simulation-based inference [Lueckmann et al., 2021]. In small-data regimes, Bayesian methods are a popular choice for a wide range of machine learning tasks, including density estimation, thanks to their attractive generalization capacities. For density estimation, the typical Bayesian approach is to target the *Bayesian predictive density*, $p_n(x) = \int f(x|\theta)\pi_n(\theta)d\theta$, where $\pi_n$ denotes the posterior density of the model parameters $\theta$ after observing $x_1, \ldots, x_n$, and $f$ denotes the likelihood function.

De Finetti's representation theorem [De Finetti, 1937, Hewitt and Savage, 1955] states that an exchangeable joint density fully characterises a Bayesian model, which then implies a sequence of predictive densities. Further, Fong et al. [2021] recently showed that a sequence of predictive densities can be sufficient for full Bayesian posterior inference. This provides theoretical motivation for an iterative approach to Bayesian predictive density estimation by updating the predictive $p_{i-1}(x)$ to $p_i(x)$ given observation $x_i$ for $i = 1, \ldots, n$. The idea of recursive Bayesian updates goes back to at least Hill [1968], but was only recently made more widely applicable through the relaxation of the assumption of exchangeability in favour of conditionally identically distributed [Berti et al., 2004] sequences.

Here, we focus on a particular class of one-step-ahead predictive updates $p_{i-1}(x) \to p_i(x)$ based on bivariate copulas, which were first introduced by Hahn et al. [2018] for univariate data, and extended by Fong et al. [2021] to the multivariate setting and to regression analyses. This class of updates is inspired by Bayesian models and thus retains many desirable Bayesian properties, such as coherence and regularization. However, we emphasize that the copula updates do not correspond exactly, nor approximately, to a traditional Bayesian likelihood-prior model, and we thus refer to them

[*]equal contribution

*Proceedings of the 39$^{th}$ Conference on Uncertainty in Artificial Intelligence* (UAI 2023), PMLR 216:658–668.

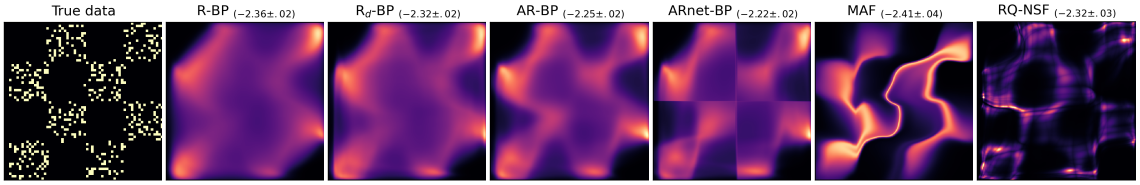



Figure 1: Density estimates of 600 observations from a chessboard distribution, reported with mean and standard deviation of test log likelihoods. For larger training sizes, see Supplement C.2. Our methods, AR-BP and ARnet-BP, outperform R-BP and AR neural networks.

as *quasi-Bayesian* [Fortini and Petrone, 2020]. The most related Bayesian density estimator proposed to date, henceforth referred to as the **R**ecursive **B**ayesian **P**redictive (R$_d$-BP), lacks flexibility to model highly complex data distributions (see Figure 1). This is because the existing copula updates rely on a Gaussian copula with a single scalar bandwidth parameter, corresponding to a Bayesian model with a likelihood that factorizes over dimensions. In contrast, popular neural network based approaches, such as masked autoregressive flows (MAFs) [Papamakarios et al., 2017], and rational-quadratic neural spline flows (RQ-NSFs) [Durkan et al., 2019] can struggle in small-data regimes (see Figure 1).

**Contributions** This motivates our main contribution, namely the formulation of a more flexible autoregressive (AR) copula update based on which we propose a new Dirichlet Process Mixture Model (DPMM) inspired density estimator. In particular:

- By considering a DPMM with an AR likelihood and a Gaussian process (GP) prior, we formulate a tractable copula update with a novel *data-dependent bandwidth* based on the Euclidean metric in data space. Our method, **A**utoregressive **R**ecursive **B**ayesian **P**redictives (AR-BP), outperforms traditional density estimators on tabular data with up to 63 features, and 10,000 samples.
- We observe in practice that the Euclidean metric used in AR-BP can be inadequate for highly non-smooth data distributions. For such cases, we propose using an AR neural network [Bengio and Bengio, 1999, Frey et al., 1998, Germain et al., 2015, Larochelle and Murray, 2011] that maps the observations into a latent space before bandwidth estimation. This introduces additional non-linearity through the dependence of the bandwidth on the data, leading to a density estimator, ARnet-BP, that is more accurate on non-smooth densities.

## 2 BACKGROUND

We briefly recap predictive density estimation via bivariate copula updates, before describing a particular such update inspired by DPMMs.

### 2.1 UNIVARIATE PREDICTIVE DENSITY UPDATES

To compute predictive densities quickly, Hahn et al. [2018] propose an iterative approach. For $x \in \mathbb{R}$, any sequence of Bayesian posterior predictive densities $p_i(x)$ with likelihood $f$ and posterior $\pi_i$, conditional on $x_{1:i}$, can be expressed as

$$p_i(x) = \int f(x|\theta)\pi_i(\theta)d\theta = p_{i-1}(x)h_i(x, x_i), \qquad (1)$$

for some bivariate function $h_i(x, x_i)$ [Hahn et al., 2018]. Rearranging for $h_i$, we have

$$h_i(x, x_i) = \frac{p_i(x)}{p_{i-1}(x)} \overset{(a)}{=} \frac{p_{i-1}(x|x_i)}{p_{i-1}(x)} \overset{(b)}{=} \frac{p_{i-1}(x, x_i)}{p_{i-1}(x)p_{i-1}(x_i)} \qquad (2)$$

where (a) holds by definition, and (b) $p_{i-1}(x, x_i) = p_{i-1}(x|x_i)p_{i-1}(x_i) = p_i(x)p_{i-1}(x_i)$ holds by Bayes' law. Hahn et al. [2018] show that $h_i(x, x_i)$ is the transformation of a bivariate copula density. A *bivariate copula* is a bivariate cumulative distribution function (CDF) $C : [0, 1]^2 \to [0, 1]$ with uniform marginal distributions that is used to characterise the dependence between two random variables independent of their marginals:

**Theorem 1** (Sklar's theorem [Sklar, 1959])**.** *For any bivariate density $f(y_1, y_2)$ with continuous marginal CDFs, $F_1(y_1)$ and $F_2(y_2)$, and marginal densities $f_1(y_1)$ and $f_2(y_2)$, there exists a unique bivariate copula $C$ with density $c$ such that*

$$f(y_1, y_2) = c\{F_1(y_1), F_2(y_2)\} f_1(y_1)f_2(y_2).$$

Applying the copula factorization from Sklar's theorem to (2) yields that there exists some bivariate copula density $c_i$ such that $p_{i-1}(x, x_i) = c_i\{P_{i-1}(x), P_{i-1}(x_i)\}p_{i-1}(x)p_{i-1}(x_i)$, and thus $h_i(x, x_i) = c_i\{P_{i-1}(x), P_{i-1}(x_i)\}$, where $P_{i-1}$ is the CDF corresponding to the predictive density $p_{i-1}$. Given prior $\pi$ and likelihood $f$, Equation 2 suggests that the update function can be written as

$$h_i(x, x_i) = \frac{\int f(x|\theta)f(x_i|\theta)\pi_{i-1}(\theta)d\theta}{\int f(x|\theta)\pi_{i-1}(\theta)d\theta \int f(x_i|\theta)\pi_{i-1}(\theta)d\theta}.$$

For each Bayesian model, there is thus a unique sequence of symmetric copula densities $c_i(u,v) = c_i(v,u)$. This sequence has the property that $c_n(\cdot,\cdot) \to 1$ converges to a constant function as $n \to \infty$, ensuring that the predictive density converges asymptotically with sample size $n$.

In general, the above equation is intractable due to the posterior so it is not possible to compute the iterative update in (1) for fully Bayesian models. Alternatively, we will consider sequences of $h_i$ that match the Bayesian model for $i = 1$, but not for $i > 1$. As mentioned above, this copula update no longer corresponds to a Bayesian model, nor are the resulting predictive density estimates approximations to a Bayesian model. Nevertheless, if the copula updates are *conditionally identically distributed*, they still exhibit desirable Bayesian characteristics such as coherence and regularization, and are hence referred to as *quasi-Bayes*. Please refer to Berti et al. [2004] for details.

### 2.2 MULTIVARIATE PREDICTIVE DENSITY UPDATES

The above arguments cannot directly be extended to multivariate $x \in \mathbb{R}^d$ since $h_i$ cannot necessarily be written as $c_i\{P_{i-1}(x), P_{i-1}(x_i)\}$ for $d > 1$. However, (2) still holds, and recursive predictive updates with bivariate copulas as building blocks can be derived explicitly given a pre-defined likelihood model and a prior, which we now exhibit.

Hahn et al. [2018] and Fong et al. [2021] propose to use DPMMs as a general-use nonparametric model. The DPMM [Escobar, 1988, Escobar and West, 1995] can be written as

$$f(x|G) = \int_\Theta K(x|\theta)\, dG(\theta), \text{ with } G \sim \mathrm{DP}(c, G_0) \quad (3)$$

where $\theta \in \Theta = \mathbb{R}^d$ are parameter vectors, the prior assigned to $G$ is a Dirichlet process (DP) prior with base measure $G_0$ and concentration parameter $c > 0$ [Ferguson, 1973], and $K(x|\theta)$ is a user-specified kernel (not to be confused with the covariance function of a GP). In particular, Fong et al. [2021] consider the base measure $G_0 = \mathcal{N}(0, \tau^{-1} I_d)$ for some precision parameter $\tau \in \mathbb{R}_{>0}$, and the factorized kernel $K(x|\theta) = \mathcal{N}(x|\theta, I_d)$ where $I_d$ is the $d$-dimensional identity matrix. The likelihood is then

$$f(x|G) = \int \prod_{j=1}^{d} \mathcal{N}\left(x^j \mid \theta^j, 1\right) dG(\theta), \quad (4)$$

where the dimensions of $x$ are conditionally independent given $\theta$. Following Hahn et al. [2018], we denote the dimension $j$ of a vector $y$ with $y^j$. We note that the strong assumption of a factorised kernel form drastically impacts the performance of the regular DPMM and also influences the form and modelling capacity of the corresponding copula update.

This model inspires the following recursive predictive density update $p_i(x) = h_i(x, x_i) p_{i-1}(x)$ for which the first $d' \in \{1, \ldots, d\}$ marginals take on the form

$$\begin{aligned} \frac{p_i(x^{1:d'})}{p_{i-1}\left(x^{1:d'}\right)} &= 1 - \alpha_i + \alpha_i \prod_{j=1}^{d'} c\left(u_{i-1}^j(x^j), v_{i-1}^j; \rho_0\right), \quad (5) \\ u_{i-1}^j(x^j) &:= P_{i-1}\left(x^j \mid x^{1:j-1}\right), \\ v_{i-1}^j &:= P_{i-1}\left(x_i^j \mid x_i^{1:j-1}\right), \end{aligned}$$

where $c(u, v; \rho_0)$ is the bivariate Gaussian copula density with correlation $\rho_0 = 1/(1+\tau)$, $p_0$ can be any chosen prior density, and $\alpha_i = \left(2 - \frac{1}{i}\right) \frac{1}{i+1}$ (see Supplement A and Fong et al. [2021]). Note that the above update requires a specific ordering of the feature dimensions, and the Gaussian copula follows from the Gaussian distribution in the kernel and $G_0$ for the DPMM. Unlike the DPMM, there are now no underlying parameters (beyond $\rho_0$) in the copula update as we have integrated out $\theta$, so we do not carry out clustering directly. While $\rho_0$ is a scalar here, Fong et al. [2021] also consider the setting with a distinct bandwidth parameter for each dimension. We refer to these recursive Bayesian predictives as $\mathrm{R}_d$-BP, or simply R-BP if the dimensions share a single bandwidth.

## 3 AR-BP: AUTOREGRESSIVE BAYESIAN PREDICTIVES

For smooth data distributions, the recursive update defined in (5) generates density estimates that are highly competitive against other popular density estimation procedures such as kernel density estimation (KDE) and DPMM [Fong et al., 2021]. Moreover, the iterative updates provide a fast estimation alternative to fitting the full DPMM through Markov chain Monte Carlo (MCMC). When considering more structured data, however, performance suffers due to the choices of the factorized kernel $K(\cdot|\theta) = \mathcal{N}(\cdot|\theta, I_d)$ and simple base measure $G_0 = \mathcal{N}(0, \tau^{-1} I_d)$ in the DPMM. These choices induce a priori independence between the data dimensions, and are thus insufficiently flexible to capture more complex dependencies.

### 3.1 BAYESIAN MODEL FORMULATION

We therefore propose employing more general kernels and base measures in the DPMM and show that these inspire a more general tractable recursive predictive

update. In particular, we allow the kernel to take on an autoregressive structure

$$K(x|\theta) = \prod_{j=1}^{d} \mathcal{N}\left(x^j \mid \theta^j\left(x^{1:j-1}\right), 1\right), \tag{6}$$

where $\theta^j : \mathbb{R}^{j-1} \to \mathbb{R}$ is now an unknown mean *function*, and not scalar, for dimension $x^j$, which we allow to depend on the previous $j-1$ dimensions of $x$. Thus, specifying our DPMM requires a base measure supported on the function space in which $(\theta^1, \ldots, \theta^d)$ is valued. We specify this base measure as a product of independent GP priors on the functional parameters

$$\theta^j \sim \mathrm{GP}(0, \tau^{-1}k^j) \text{ for } j = 1, ..., d \tag{7}$$

where $k^j : \mathbb{R}^{j-1} \times \mathbb{R}^{j-1} \to \mathbb{R}$ and $k^j$ can be any given covariance function that takes as input a pair of $x^{1:j-1}$ values. In practice, we use the same functional form of $k$ for each $j$, so we will drop the superscript $j$. For later convenience, we have also written the scaling term $\tau^{-1}$ explicitly. We highlight that for $j = 1$, $\theta^1 \sim \mathcal{N}(0, \tau^{-1})$. Under this choice, the mean of the normal kernels in the DPMM for each dimension $j$ is thus a flexible function of the first $j-1$ dimensions $x^{1:j-1}$, on which we elicit independent GP priors. The conjugacy of the GP with the Gaussian DPMM kernel in (6) is crucial for deriving a tractable density update.

*Remark.* The proposed DPMM kernel in (6) is in fact more flexible than a general multivariate kernel, $K(x \mid \theta) = \mathcal{N}(x \mid \theta, \Sigma)$. This is because the multivariate kernel also implies an AR form like (6) but where the parameters $\theta^j$ are restricted to be linear in $x^{1:j-1}$; see Wade et al. [2014] for details.

### 3.2 ITERATIVE PREDICTIVE DENSITY UPDATES

Computing the Bayesian posterior predictive density induced by the DPMM with kernel given by (6) and base measure given by (7) through posterior estimation is *intractable* and requires MCMC. However, as before, we can utilize the model to derive tractable iterative copula updates. In Supplement A.1, we derive the corresponding recursive predictive density update $p_i(x) = h_i(x, x_i)p_{i-1}(x)$ for the first $d'$ marginals and show that it takes on the form

$$\frac{p_i(x^{1:d'})}{p_{i-1}(x^{1:d'})} = 1 - \alpha_i + \alpha_i \cdot \prod_{j=1}^{d'} c\left(u_{i-1}^j(x^j), v_{i-1}^j; \rho^j(x^{1:j-1}, x_i^{1:j-1})\right), \tag{8}$$

with $u_{i-1}^j(x^j), v_{i-1}^j$ defined as in (5), $\alpha_i = \left(2 - \frac{1}{i}\right)\frac{1}{i+1}$, and the bandwidth given by

$$\rho^j(x^{1:j-1}, x_i^{1:j-1}) = \rho_0 k\left(x^{1:j-1}, x_i^{1:j-1}\right), \tag{9}$$

(a) **Train:** Estimate $v_i^j = P_{i-1}(x_i^{1:j})$ for each $i$

Initialise $u_0^j(x_i) \leftarrow \Phi(x_i^j)$

For each preceding observation $x_k$ with $k < i$:

- For each feature $j$:
  - Compute *data-dependent* bandwidth $\rho^j(x_i^{1:j}, x_k^{1:j})$ (9)
  - Update conditional CDF $u_i^j(x_k) := P_i^j(x_k)$ based on the similarity between $u_{i-1}^j(x_k)$ and $v_{i-1}^j$ (18)

Set $v_i^j \leftarrow u_i^j(x_i)$ for all $j$

(b) **Test:** Estimate predictive at test point $p_n(z)$

Initialise $u_0^j(z) \leftarrow \Phi(z^j)$

For each train observation $x_i$:

- For each feature $j$:
  - Compute *data-dependent* bandwidth $\rho^j(x_i^{1:j}, z^{1:j})$ (9)
  - Update conditional CDF $u_i^j(z) := P_i^j(z)$ based on the similarity between $u_{i-1}^j(z)$ and $v_{i-1}^j$ (18)
- Update predictive density $p_{i-1}(z) \to p_i(z)$ (8)

Figure 2: Simplified summary of AR-BP. We repeat the training update for each train datum $x_i$ to estimate $v_i^j = P_{i-1}(x_i^{1:j})$. These are needed at test time to update from $p_{i-1}(z) \to p_i(z)$. All steps are averaged over different feature and sample permutations. The main step that induces autoregression in the observations is highlighted pink. Please see Supplement B.3 for detailed algorithms.

for $\rho_0 = 1/(1+\tau)$, and $\rho_i^1 = \rho_0$. Where appropriate, we henceforth drop the argument $x$ for brevity. The conditional CDFs $u_{i-1}^j$ can also be computed through an iterative closed form expression similarly to (8) (Supplement B.3). Please see Figure 2 for a simplified overview of the density estimation pipeline.

Note that the estimation is identical to the update given in (5) induced by the factorized DPMM kernel, except for the main difference that the bandwidth $\rho$ is *no longer a constant*, but is now *data-dependent*. More precisely, the bandwidth for dimension $j$ is a transformation of the GP covariance function $k$ on the first $j-1$ dimensions. The additional flexibility afforded by the inclusion of $k$ enables us to capture more complex dependency structures, as we do not enforce a-priori independence between the dimensions of the parameter $\theta$. Similarly to the extension of R-BP to R$_d$-BP, we can also define AR$_d$-BP by introducing dimension dependence in $\rho_0$. Finally, we highlight that extending R-BP to mixed data is possible as given in Appendix E.1.3 of Fong et al. [2021], which also extends naturally to AR-BP.

*Remark.* The data-dependent bandwidth also appears when starting from other Bayesian nonparametric models, such as dependent DPs and GPs (see Supplement

A.2.2 for the derivation).

Our approach can be viewed as a Bayesian version of an online KDE procedure. To see this, note that a KDE trained on $i-1$ observations – yielding the density estimate $q_{i-1}(x)$ – can be updated after observing the $i^{th}$ observation $x_i$ via $q_i(x) = (1-\alpha_i)q_{i-1}(x) + \alpha_i d(x, x_i)$, where $\alpha_i = 1/i$ and $d(\cdot,\cdot)$ denotes the kernel of the KDE. Rather than adding a weighted kernel term directly, AR-BP instead adds an adaptive kernel that depends on a notion of distance between $x$ and $x_i$ based on the predictive CDFs conditional on $x_{1:i-1}$.

To better understand the importance of the data-dependent bandwidth, we compare the conditional predictive mean of R-BP and AR-BP in the bivariate setting $X \times Y$. Under the simplifying assumption of Gaussian predictive densities, we show in Supplement A.3 that the conditional mean of $Y \mid X$ is given by

$$\mu_i(x) = \mu_{i-1}(x) + \alpha_i(x, x_i)\rho(x, x_i)(y_i - \mu_{i-1}(x_i)),$$
$$\alpha_i(x, x_i) = \frac{\alpha_i c(P_{i-1}(x), P_{i-1}(x_i); \rho)}{1 - \alpha_i + \alpha_i c(P_{i-1}(x), P_{i-1}(x_i); \rho)}.$$

Note that $\rho(x, x_i) = \rho_0$ for R-BP. Intuitively, the updated mean is the previous mean plus a residual term at $y_i$ scaled by some notion of distance between $x$ and $x_i$. For R-BP, this distance between $x$ and $x_i$ depends only on their predictive CDF values through $\alpha_i(x, x_i)$. This can result in undesirable behaviour as shown in the upper plot in Figure 3(a), where the peak of $\alpha_i(x, x_i)$, as a function of $x$, is not centred at $x_i$. Counterintuitively, there is thus an $x > x_i$ where $\mu_i(x)$ is updated more than at the actual observed $x = x_i$. This follows from the lack of focus on *conditional* density estimates for R-BP, which is alleviated by AR-BP. In the AR case, $\rho(x, x_i)$ takes into account the Euclidean distance between $x$ and $x_i$ in the data space. We see in the lower plot in Figure 3(a) that the peak is closer to $x_i$. Figure 3(b) further demonstrates this difference on another toy example - we see that R-BP struggles to fit a linear conditional mean function for $n = 4$, focussing density in data sparse regions, while AR-BP succeeds to assign significant density only to points on the data manifold.

**Training the update parameters** In order to compute the predictive density $p_n(x^*)$, we require the vector of conditional CDFs $[v_1^j, \ldots, v_{n-1}^j]$ where $v_i^j = P_i(x_{i+1}^j \mid x_{i+1}^{1:j-1})$. Given a bandwidth parameterization, obtaining this vector thus amounts to model-fitting, and each $v_i^j$ requires $i-1$ iterations (Supplement B.3), for $i \in \{1, \ldots, n\}$. We note that the order of samples and dimensions influences the prediction performance in AR density estimators [Vinyals et al., 2015]. In practice, averaging over different permutations of these improves performance (Supplement B.3). Full implementation details can be found in Supplement B.

**Computational complexity** The above procedure results in a computational complexity of $\mathcal{O}(Mdn^2)$ at the training stage where $M$ is the number of permutations. At test time, we have already obtained the necessary conditional prequential CDFs $v_n^j$ in computing the prequential log-likelihood above. As a result, we have a computational complexity $\mathcal{O}(Mdn)$ for each test observation. Note that the introduction of a data-dependent bandwidth does not increase the computational complexity at train or test time relative to R-BP and only adds a negligible factor to the computational time for the calculation of the bandwidth.

### 3.3 BANDWIDTH PARAMETERISATION

The choice of covariance function in (7) provides substantial modelling flexibility in our AR-BP framework. Moreover, the additional parameters associated with the covariance function allow us to tune the implied covariance structure according to the observed data. This formulation enables us to draw upon the rich literature on the choice of covariance functions for Gaussian processes [Williams and Rasmussen, 2006]. For simplicity we only consider the most popular such choice here, but study the more flexible rational-quadratic covariance in Supplement C.2. The radial basis function (RBF) covariance function is defined as $k_\ell(x^{1:j-1}, x'^{1:j-1}) = \exp[-\sum_{\kappa=1}^{j-1}\{(x^\kappa - x'^\kappa)/\ell^\kappa\}^2]$, where $\ell \in \mathbb{R}_{>0}^{d-1}$ is the length scale.

**Neural parameterisation** As we saw in the motivating example of the density estimation of a chessboard distribution in Figure 1, the RBF kernel can restrict the capacity of the predictive density update to capture intricate nonlinearities if the training data size is not sufficient. While the parameterization of the bandwidth in (9) was initially derived via the first predictive update for a DPMM, all we require is that the bandwidth function $\rho^j : \mathbb{R}^{j-1} \times \mathbb{R}^{j-1} \to \mathbb{R}$ lies in (0,1). We would also like $\rho^j(x^{1:j-1}, x'^{1:j-1})$ to take larger values when $x^{1:j-1}$ and $x'^{1:j-1}$ are 'close' in some sense. Motivated by this observation, we now consider more expressive bandwidth functions that can lead to increased predictive performance. In particular, we formulate an AR neural network $f_w : \mathbb{R}^d \to \mathbb{R}^{d \times d'}$ for $d' \in \mathbb{N}$ with the property that the $j^{th}$ row of the output depends only on the first $j-1$ dimensions of the input. Let $Z = f_w(x)$ and denoting $z^j$ to be the $j^{th}$ row of the matrix $Z$, the covariance function is then computed as $\rho^j(x^{1:j-1}, x'^{1:j-1}) = \rho_0 \exp(-\sum_{\kappa=1}^{j-1} ||z^\kappa - z'^\kappa||_2^2)$.

Numerous AR neural network models have been extensively used for density estimation [Dinh et al., 2014, Huang et al., 2018, Kingma et al., 2016]. In our experiments, we use a relatively simple model with parameter

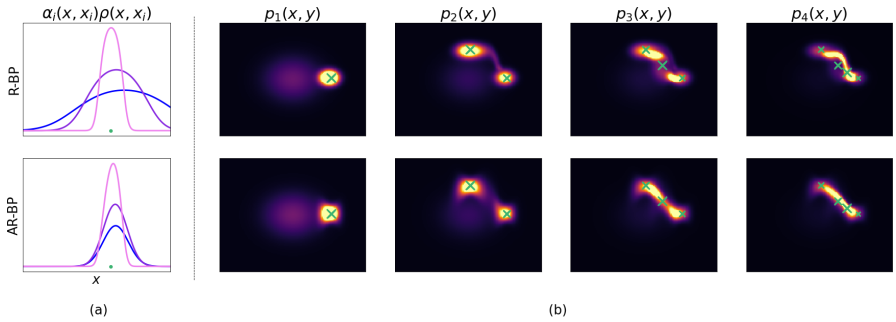



Figure 3: (a) Plots of $\alpha_i(x, x_i)\rho(x, x_i)$ for R-BP and AR-BP for $\rho_0 \in \{0.5, 0.7, 0.95\}$ (—,—,—) with new observation $x_i$ (•). Note that $\rho(x, x_i) = \rho_0$ for R-BP, and $\ell = 1$ for AR-BP. (b) Density plots for R-BP and AR-BP trained on 4 sequential data points (×). Both figures show that the update of R-BP, unlike AR-BP, is not centred around the new datum.

sharing inspired by NADE, an AR neural network designed for density estimation [Larochelle and Murray, 2011]. More advanced properties like the permutation invariance of MADE [Papamakarios et al., 2017] create an additional overhead that cannot be used in the copula formulation as the predictive update is not permutation-invariant. We refer to Bayesian predictive densities estimated using AR neural networks as *ARnet Bayesian predictives* (ARnet-BP).

**Tuning the bandwidth function** Recall that the bandwidths $\rho_i(\cdot, \cdot)$ are parameterised by $\rho_0$ and the parameters of the chosen covariance functions or neural embedders. For AR-BP, these are the length scales $\ell$ of the RBF covariance function, while for ARnet-BP, these are the parameters $w$ of the AR neural network. We fit these tunable parameters in a data-driven approach by maximising the prequential [Dawid, 1997] log-likelihood $\sum_{i=1}^{n} \log p_{i-1}(x_i)$ which is analogous to the Bayesian marginal likelihood – the tractable predictive density allows us to compute this exactly, and this approach is analogous to empirical Bayes. Specifically, we use gradient descent optimisation with Adam, sampling a different random permutation of the training data at each optimisation step (Supplement B.3).

## 4 RELATED WORK

Our work falls into the broad area of multivariate density estimation [Scott, 2015]. While AR networks have been previously used directly for the task of density estimation [Bengio and Bengio, 1999, Germain et al., 2015, Larochelle and Murray, 2011], we use them to elicit a data-dependent bandwidth in the predictive update to mitigate the smoothing effect observed in AR-BP. Neural network based approaches, however, often underperform in small-data regimes. Deep learning approaches that do target few-shot density estimation require complex meta-learning and pre-training pipelines [Gu et al., 2020, Reed et al., 2017].

Our work directly extends the contributions of Hahn et al. [2018] and Fong et al. [2021] through an alternative specification of the nonparametric Bayesian model in the recursive predictive update scheme. R-BP has recently been used for nonparametric solvency risk prediction [Hong and Martin, 2019], and survival analysis [Fong and Lehmann, 2022]. Berti et al. [2021a,b, 2004] also focus on univariate predictive updates in the Bayesian nonparametric paradigm, specifically exploring the use of the conditionally identically distributed condition as a relaxation of the standard exchangeability assumption. Other studies have investigated quasi-Bayesian updates in the special case of the mixing distribution in nonparametric mixture models [Dixit and Martin, 2022, Fortini and Petrone, 2020, Martin, 2018, Tokdar et al., 2009], though these typically focus on univariate or low-dimensional spaces. See also Martin [2021] for a survey.

Finally, copulas are a well-studied tool for modelling the correlations in multivariate data (see e.g. Kauermann et al. [2013], Ling et al. [2020], Nelsen [2007]). Copula density estimation aims to construct density estimates whose univariate marginals are uniform [Gijbels and Mielniczuk, 1990], and often focus on modelling strong tail dependencies [Wiese et al., 2019]. In contrast, we employ bivariate copulas for generic multivariate density estimation as a tool to model the correlations between subsequent subjective predictive densities, rather than across the data dimensions directly.

## 5 EXPERIMENTS

We demonstrate the benefits of AR-BP, AR$_d$-BP and ARnet-BP for density estimation and prediction tasks in an experimental study with five baseline approaches

Table 1: Average NLL with standard error over five runs on data sets analysed by Fong et al. [2021].

| n/d | WINE 89/12 | BREAST 97/14 | PARKIN 97/16 | IONO 175/30 | BOSTON 506/13 |
|---|---|---|---|---|---|
| KDE | $13.69_{\pm 0.00}$ | $10.45_{\pm 0.24}$ | $12.83_{\pm 0.27}$ | $32.06_{\pm 0.00}$ | $8.34_{\pm 0.00}$ |
| DPMM (Diag) | $17.46_{\pm 0.6}$ | $16.26_{\pm 0.71}$ | $22.28_{\pm 0.66}$ | $35.30_{\pm 1.28}$ | $7.64_{\pm 0.09}$ |
| DPMM (Full) | $32.88_{\pm 0.82}$ | $26.67_{\pm 1.32}$ | $39.95_{\pm 1.56}$ | $86.18_{\pm 10.22}$ | $9.45_{\pm 0.43}$ |
| MAF | $39.60_{\pm 1.41}$ | $10.13_{\pm 0.40}$ | $11.76_{\pm 0.45}$ | $140.09_{\pm 4.03}$ | $56.01_{\pm 27.74}$ |
| RQ-NSF | $38.34_{\pm 0.63}$ | $26.41_{\pm 0.57}$ | $31.26_{\pm 0.31}$ | $54.49_{\pm 0.65}$ | $-2.20_{\pm 0.11}$ |
| R-BP | $13.57_{\pm 0.04}$ | $7.45_{\pm 0.02}$ | $9.15_{\pm 0.04}$ | $21.15_{\pm 0.04}$ | $4.56_{\pm 0.04}$ |
| $R_d$-BP | $13.32_{\pm 0.01}$ | $6.12_{\pm 0.05}$ | $7.52_{\pm 0.05}$ | $19.82_{\pm 0.08}$ | $-13.50_{\pm 0.59}$ |
| AR-BP | $13.45_{\pm 0.05}$ | $6.18_{\pm 0.05}$ | $8.29_{\pm 0.11}$ | $17.16_{\pm 0.25}$ | $-0.45_{\pm 0.77}$ |
| $AR_d$-BP | $\mathbf{13.22_{\pm 0.04}}$ | $\mathbf{6.11_{\pm 0.04}}$ | $\mathbf{7.21_{\pm 0.12}}$ | $16.48_{\pm 0.26}$ | $\mathbf{-14.75_{\pm 0.89}}$ |
| ARnet-BP | $14.41_{\pm 0.11}$ | $6.87_{\pm 0.23}$ | $8.29_{\pm 0.17}$ | $\mathbf{15.32_{\pm 0.35}}$ | $-5.71_{\pm 0.62}$ |

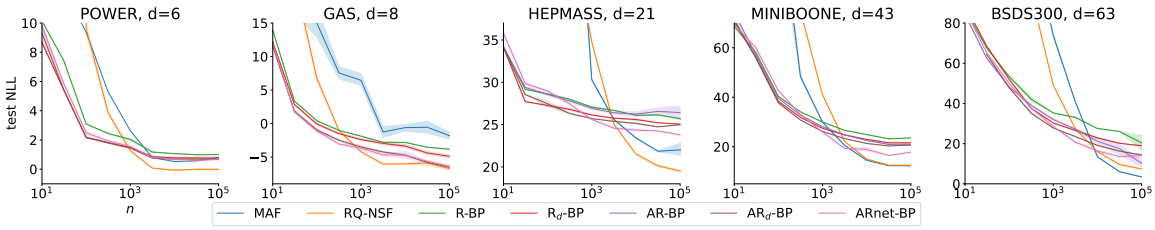



Figure 4: Average NLL and standard errors over 10 runs for training sets of different size. Our models outperform neural methods for data sets up to 10,000 samples.

and 13 different data sets. The code and data is available at `https://github.com/sghalebikesabi/autoregressive-bayesian-predictives`. See Supplement C for additional experimental details and results, including a sensitivity study, an ablation study, further illustrative examples, a preliminary investigation into image examples, and an empirical study of the computational complexity of the proposed methods.

### 5.1 DENSITY ESTIMATION

We compared our models against KDEs [Parzen, 1962], DPMMs [Rasmussen, 1999], MAFs [Papamakarios et al., 2017] and RQ-NSFs [Durkan et al., 2019]. The hyperparameters of the baselines were tuned with cross-validation. Unless otherwise specified, we use respectively 10 permutations over samples and features to average the quasi-Bayesian estimates. We did not see substantial improvements with more permutations. We use the same few hyperparameters (initialisation of $\rho_0, l_1, \ldots, l_d$, number of permutations, neural network architecture, and learning rate) on all data sets as our method is robust to their choice. See Supplement C.1 for further information.

**Data sets analysed by Fong et al. [2021]** See Table 1 for the negative log-likelihood (NLL) estimated on five UCI data sets [Asuncion and Newman, 2007] of small size with up to 506 samples, as investigated by Fong et al. [2021]. Our proposed methods display highly competitive performance: $AR_d$-BP achieved the best test NLL on four of the data sets, while ARnet-BP prevailed on ionosphere.

**Data sets analysed by Papamakarios et al. [2017]** A number of UCI data sets have become the standard evaluation benchmark for deep AR models [Durkan et al., 2019, Huang et al., 2018, Papamakarios et al., 2017]. These include low-dimensional data sets with up to 63 features, but at least 29,000 with up to $10^6$ samples. In many circumstances, data sets of such a data size are not available. To investigate performance as a function of sample size, we trained the models on subsets of the full data set. We do not report results for the KDEs and the DPMM estimators here as these estimators performed significantly worse than the other approaches. Similarly, we do not report deep learning results for sample sizes smaller than $10^2$. See Supplement C.2 for complete results.

In the small-data regime, we observe that the R-BP methods significantly outperform the neural density estimators (Figure 4). As the sample size increases, the gap in performance decreases until eventually the neural density estimators outcompete the R-BP methods. The performance between the R-BP methods and our proposed AR extensions is largely similar, though we note that the AR-BP methods were generally more effective on the GAS dataset.

### 5.2 SUPERVISED LEARNING

R-BP methods, including AR-BP, can be used for prediction tasks such as regression and classification [Fong

Table 2: Average NLL over five runs reported with standard error for supervised tasks

| | Regression | | | Classification | | |
|---|---|---|---|---|---|---|
| | BOSTON | CONCR | DIAB | IONO | PARKIN | MNIST01 |
| n/d | 506/13 | 1,030/8 | 442/10 | 351/33 | 195/22 | 12,031/784 |
| Linear | $0.87_{\pm 0.03}$ | $0.99_{\pm 0.01}$ | $1.07_{\pm 0.01}$ | $0.33_{\pm 0.01}$ | $0.38_{\pm 0.01}$ | $0.003_{\pm 0.000}$ |
| GP | $0.42_{\pm 0.08}$ | $0.36_{\pm 0.02}$ | $1.06_{\pm 0.02}$ | $0.30_{\pm 0.02}$ | $0.42_{\pm 0.02}$ | $0.035_{\pm 0.000}$ |
| MLP | $1.42_{\pm 1.01}$ | $2.01_{\pm 0.98}$ | $3.32_{\pm 4.05}$ | $0.26_{\pm 0.05}$ | $0.31_{\pm 0.02}$ | $\mathbf{0.003_{\pm 0.000}}$ |
| R-BP | $0.76_{\pm 0.09}$ | $0.87_{\pm 0.03}$ | $1.05_{\pm 0.03}$ | $0.26_{\pm 0.01}$ | $0.37_{\pm 0.01}$ | $0.015_{\pm 0.001}$ |
| $R_d$-BP | $0.40_{\pm 0.03}$ | $0.42_{\pm 0.00}$ | $1.00_{\pm 0.02}$ | $0.34_{\pm 0.02}$ | $0.27_{\pm 0.03}$ | $0.018_{\pm 0.001}$ |
| AR-BP | $0.52_{\pm 0.13}$ | $0.42_{\pm 0.01}$ | $1.06_{\pm 0.02}$ | $0.21_{\pm 0.02}$ | $0.29_{\pm 0.02}$ | $0.015_{\pm 0.001}$ |
| $AR_d$-BP | $\mathbf{0.37_{\pm 0.10}}$ | $0.39_{\pm 0.01}$ | $\mathbf{0.99_{\pm 0.02}}$ | $\mathbf{0.20_{\pm 0.02}}$ | $0.28_{\pm 0.03}$ | $0.017_{\pm 0.001}$ |
| ARnet-BP | $0.45_{\pm 0.11}$ | $\mathbf{-0.03_{\pm 0.00}}$ | $1.41_{\pm 0.07}$ | $0.24_{\pm 0.04}$ | $\mathbf{0.26_{\pm 0.04}}$ | $0.014_{\pm 0.001}$ |

et al., 2021]. In short, this is achieved by estimating the conditional predictive density $p_n(y|x)$ of the labels $y$ directly by assuming a dependent Dirichlet process likelihood. See Supplement B.2 for details. Again, we follow the experimental set-up of Fong et al. [2021], and additionally report results on the MNIST data set, restricted to digits of class 0 and 1. We report the conditional test NLL $-\frac{1}{n'}\sum_i \log p_n(y_i^*|x_i^*)$ for a test set $\{(x_1^*, y_1^*), \ldots, (x_{n'}^*, y_{n'}^*)\}$. We compared our models against a GP, a linear Bayesian model (Linear), and a one-hidden-layer multilayer perceptron (MLP) on several classification and regression tasks. To get a distribution over the predicted outcome in the regression case, we trained an ensemble over 10 MLPs. Our proposed methods were again highly competitive (Table 2). $AR_d$-BP performed best on two regression tasks and one classification task. ARnet-BP was substantially better than the remaining methods on CONCR and also performed best on the PARKIN. On the other hand, the MLP model was best on MNIST.

## 6 DISCUSSION

Although Bayesian methods generally perform well in the small sample setting, the conventional Bayesian approach to density estimation, i.e. DPMM estimation via the posterior predictive, is computationally intensive. Here, we set out to propose a computationally efficient density estimator as an alternative to DPMM density estimation. We recommend its use for tabular data sets of up to 63 features, and 10,000 observations. Such data set sizes are ubiquitous in healthcare, finance, hyperparameter tuning, and survey data applications.

We expand upon the tractable recursive copula updates of Fong et al. [2021], Hahn et al. [2018] by incorporating regression methods, such as kernels and neural networks. This introduces a data-dependent bandwidth, thus increasing the flexibility of this class of models, with little computational overhead compared to R-BP. More generally, it would be of interest to integrate other machine learning methods with recursive copula updates. Furthermore, other Bayesian nonparametric models may inspire other recursive copula updates–see Appendix A.2 for an example based on GPs.

An appealing feature of AR-BP is that it requires no manual hyperparameter tuning. Further, on small data sets, AR-BP shows state-of-the-art generalization and is faster than competing deep learning models. It significantly increases the modelling capacity of the baseline R-BP via a data-dependent bandwidth. Additionally, ARnet-BP provides a useful illustration of how powerful neural network models can be incorporated into R-BP methods to improve density estimation. Future work can investigate alternative architectures for structured data. Our work adds to the rich body of density estimators and thus we do not anticipate any additional negative societal impact arising from our proposal.

This strong performance of AR-BP (and other copula methods) in the small data regime is likely due to its Bayesian-like regularization towards an initial density $p_0$, as shown in the weighted sum in (8). Its weaker performance in the large data regime may be due to the importance of the sequence $\alpha_i$ which governs how regularization decays, but further theoretical work is needed to understand AR-BP's asymptotic behaviour. A limitation of R-BP methods, including AR-BP, is the quadratic time dependence on the number of training observations. Subsampling techniques thus offer a particularly promising avenue to reduce the overall computational cost and warrant further investigation. Although the recursive updates depend on the sample and covariate ordering, it is possible to alleviate this dependence though by estimating the R-BP over multiple permutations in parallel, as we have done in the above experiments. Nevertheless, the algorithm is relatively fast: with a single GPU, we were able to train models with 100,000 observations in less than an hour.

The use of a GP prior greatly increases the flexibility of our framework. Moreover, it opens the door to future research to incorporate ideas from the vast GP literature to further boost performance in high-dimensional settings. Our use of the RBF kernel was illustrative; other kernels are discussed in Appendix C.2 where we find that the RBF kernel performs best. For example, we anticipate that the use of recent advances in convo-

lutional kernels [Van der Wilk et al., 2017] would be particularly suited for computer vision tasks.

**Acknowledgements**

SG is a student of the EPSRC CDT in Modern Statistics and Statistical Machine Learning (EP/S023151/1) and receives funding from the Oxford Radcliffe Scholarship and Novartis. CH is supported by The Alan Turing Institute, Health Data Research UK, the Medical Research Council UK, the EPSRC through the Bayes4Health programme Grant EP/R018561/1, and AI for Science and Government UK Research and Innovation (UKRI). EF is supported by Novo Nordisk. BL is supported by the UK Engineering and Physical Sciences Research Council through the Bayes4Health programme (grant number EP/R018561/1) and gratefully acknowledges funding from Jesus College, Oxford.

## REFERENCES

Arthur Asuncion and David Newman. UCI machine learning repository, 2007.

Yoshua Bengio and Samy Bengio. Modeling high-dimensional discrete data with multi-layer neural networks. *Advances in Neural Information Processing Systems*, 12, 1999.

Patrizia Berti, Luca Pratelli, and Pietro Rigo. Limit theorems for a class of identically distributed random variables. *The Annals of Probability*, 32(3):2029–2052, 2004.

Patrizia Berti, Emanuela Dreassi, Fabrizio Leisen, Pietro Rigo, and Luca Pratelli. Bayesian predictive inference without a prior. *arXiv preprint arXiv:2104.11643*, 2021a.

Patrizia Berti, Emanuela Dreassi, Luca Pratelli, and Pietro Rigo. A class of models for Bayesian predictive inference. *Bernoulli*, 27(1):702–726, 2021b.

A Philip Dawid. Prequential analysis. *Encyclopedia of Statistical Sciences*, 1:464–470, 1997.

Bruno De Finetti. La prévision: ses lois logiques, ses sources subjectives. *Annales de l'institut Henri Poincaré*, 7:1–68, 1937.

Laurent Dinh, David Krueger, and Yoshua Bengio. Nice: Non-linear independent components estimation. *arXiv preprint arXiv:1410.8516*, 2014.

Vaidehi Dixit and Ryan Martin. A prticle filter algorithm for nonparametric estimation of multivariate mixing distributions. *arXiv preprint arXiv:2204.01646*, 2022.

Conor Durkan, Artur Bekasov, Iain Murray, and George Papamakarios. Neural spline flows. *Advances in neural information processing systems*, 32, 2019.

Michael D Escobar and Mike West. Bayesian density estimation and inference using mixtures. *Journal of the american statistical association*, 90(430):577–588, 1995.

Michael David Escobar. *Estimating the means of several normal populations by nonparametric estimation of the distribution of the means*. PhD thesis, Yale University, 1988.

Thomas S Ferguson. A Bayesian analysis of some nonparametric problems. *The annals of statistics*, pages 209–230, 1973.

Edwin Fong and Brieuc Lehmann. A predictive approach to bayesian nonparametric survival analysis. In Gustau Camps-Valls, Francisco J. R. Ruiz, and Isabel Valera, editors, *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics*, volume 151 of *Proceedings of Machine Learning Research*, pages 6990–7013. PMLR, 28–30 Mar 2022. URL `https://proceedings.mlr.press/v151/fong22a.html`.

Edwin Fong, Chris Holmes, and Stephen G Walker. Martingale posterior distributions. *To appear at the Journal of the Royal Statistical Society: Series B (with discussion)*, 2021.

Sandra Fortini and Sonia Petrone. Quasi-bayes properties of a procedure for sequential learning in mixture models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 82(4):1087–1114, 2020.

Brendan J Frey, J Frey Brendan, and Brendan J Frey. *Graphical models for machine learning and digital communication*. MIT press, 1998.

Mathieu Germain, Karol Gregor, Iain Murray, and Hugo Larochelle. Made: Masked autoencoder for distribution estimation. In *International Conference on Machine Learning*, pages 881–889. PMLR, 2015.

Iène Gijbels and Jan Mielniczuk. Estimating the density of a copula function. *Communications in Statistics - Theory and Methods*, 19(2):445–464, January 1990. ISSN 0361-0926. doi: 10.1080/03610929008830212. URL `https://doi.org/10.1080/03610929008830212`.

Ke Gu, Yonghui Zhang, and Junfei Qiao. Ensemble meta-learning for few-shot soot density recognition. *IEEE Transactions on Industrial Informatics*, 17(3):2261–2270, 2020.

P Richard Hahn, Ryan Martin, and Stephen G Walker. On recursive Bayesian predictive distributions. *Journal of the American Statistical Association*, 113(523): 1085–1093, 2018.

Edwin Hewitt and Leonard J Savage. Symmetric measures on cartesian products. *Transactions of the American Mathematical Society*, 80(2):470–501, 1955.

Bruce M Hill. Posterior distribution of percentiles: Bayes' theorem for sampling from a population. *Journal of the American Statistical Association*, 63(322): 677–691, 1968.

Liang Hong and Ryan Martin. Real-time Bayesian non-parametric prediction of solvency risk. *Annals of Actuarial Science*, 13(1):67–79, 2019.

Chin-Wei Huang, David Krueger, Alexandre Lacoste, and Aaron Courville. Neural autoregressive flows. In *International Conference on Machine Learning*, pages 2078–2087. PMLR, 2018.

Göran Kauermann, Christian Schellhase, and David Ruppert. Flexible copula density estimation with penalized hierarchical b-splines. *Scandinavian Journal of Statistics*, 40(4):685–705, 2013.

Durk P Kingma, Tim Salimans, Rafal Jozefowicz, Xi Chen, Ilya Sutskever, and Max Welling. Improved variational inference with inverse autoregressive flow. *Advances in neural information processing systems*, 29, 2016.

Hugo Larochelle and Iain Murray. The neural autoregressive distribution estimator. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 29–37. JMLR Workshop and Conference Proceedings, 2011.

Chun Kai Ling, Fei Fang, and J Zico Kolter. Deep archimedean copulas. *Advances in Neural Information Processing Systems*, 33:1535–1545, 2020.

Jan-Matthis Lueckmann, Jan Boelts, David Greenberg, Pedro Goncalves, and Jakob Macke. Benchmarking simulation-based inference. In *International Conference on Artificial Intelligence and Statistics*, pages 343–351. PMLR, 2021.

Ryan Martin. On nonparametric estimation of a mixing density via the predictive recursion algorithm. *arXiv preprint arXiv:1812.02149*, 2018.

Ryan Martin. A survey of nonparametric mixing density estimation via the predictive recursion algorithm. *Sankhya B*, 83(1):97–121, 2021.

R.B. Nelsen. *An Introduction to Copulas*. Springer Series in Statistics. Springer New York, 2007. ISBN 978-0-387-28678-5.

George Papamakarios, Theo Pavlakou, and Iain Murray. Masked autoregressive flow for density estimation. *Advances in neural information processing systems*, 30, 2017.

Emanuel Parzen. On estimation of a probability density function and mode. *The annals of mathematical statistics*, 33(3):1065–1076, 1962.

Carl Rasmussen. The infinite gaussian mixture model. *Advances in neural information processing systems*, 12, 1999.

Scott Reed, Yutian Chen, Thomas Paine, Aäron van den Oord, SM Eslami, Danilo Rezende, Oriol Vinyals, and Nando de Freitas. Few-shot autoregressive density estimation: Towards learning to learn distributions. *arXiv preprint arXiv:1710.10304*, 2017.

D Scaldelai, LC Matioli, SR Santos, and M Kleina. Multiclusterkde: a new algorithm for clustering based on multivariate kernel density estimation. *Journal of Applied Statistics*, 49(1):98–121, 2022.

David W Scott. *Multivariate density estimation: theory, practice, and visualization*. John Wiley & Sons, 2015.

M Sklar. Fonctions de repartition an dimensions et leurs marges. *Publ. inst. statist. univ. Paris*, 8:229–231, 1959.

Surya T Tokdar, Ryan Martin, and Jayanta K Ghosh. Consistency of a recursive estimate of mixing distributions. *The Annals of Statistics*, pages 2502–2522, 2009.

Mark Van der Wilk, Carl Edward Rasmussen, and James Hensman. Convolutional gaussian processes. *Advances in Neural Information Processing Systems*, 30, 2017.

Oriol Vinyals, Samy Bengio, and Manjunath Kudlur. Order matters: Sequence to sequence for sets. *arXiv preprint arXiv:1511.06391*, 2015.

Sara Wade, Stephen G Walker, and Sonia Petrone. A predictive study of dirichlet process mixture models for curve fitting. *Scandinavian Journal of Statistics*, 41(3):580–605, 2014.

Magnus Wiese, Robert Knobloch, and Ralf Korn. Copula & marginal flows: Disentangling the marginal from its joint. *arXiv preprint arXiv:1907.03361*, 2019.

Christopher K Williams and Carl Edward Rasmussen. *Gaussian processes for machine learning*. Number 3 in 2. MIT press Cambridge, MA, 2006.

Daniel Zoran and Yair Weiss. From learning models of natural image patches to whole image restoration. In *2011 International Conference on Computer Vision*, pages 479–486. IEEE, 2011.

# Quasi-Bayesian Nonparametric Density Estimation via Autoregressive Predictive Updates (Supplementary Material)

**Sahra Ghalebikesabi**[1] **Chris Holmes**[2] **Edwin Fong**[*2] **Brieuc Lehmann**[*3]

[1]University of Oxford
[2]Novo Nordisk
[3]University College London

## A DERIVATIONS

### A.1 DERIVATION OF AR-BP

For illustration purposes, we first start by summarising the derivation of the update without autoregression, closely following Appendix E.1.2 in Fong et al. [2021].

#### A.1.1 No Autoregression (R-BP)

The multivariate DPMM with factorized kernel has the form

$$f_G(x) = \int \prod_{j=1}^{d} \mathcal{N}(x^j \mid \theta^j, 1)\, dG(\theta), \quad G \sim \mathrm{DP}\left(a, G_0\right), \quad G_0(\theta) = \prod_{j=1}^{d} \mathcal{N}(\theta^j \mid 0, \tau^{-1}).$$

Given

$$p_i(x) = p_{i-1}(x) h_i(x, x_i),$$

Hahn et al. [2018] and Fong et al. [2021] derive the predictive density updates for R-BP by initally only considering the first step update $h_1$

$$p_1(x) = p_0(x) h_1(x, x_1)\cdot$$

From

$$h_i(x, x_i) = \frac{\int f(x|\theta) f(x_i|\theta) \pi_{i-1}(\theta) d\theta}{\int f(x|\theta)\pi_{i-1}(\theta)d\theta \int f(x_n|\theta)\pi_{i-1}(\theta)d\theta},$$

it follows that

$$h_1(x, x_1) = \frac{E\left[f_G(x)\, f_G(x_1)\right]}{p_0(x)\, p_0(x_1)} \tag{1}$$

where the expectation is over $G$ coming from the prior. Following the stick-breaking representation of the DP, Fong et al. [2021] write $G$ as

$$G = \sum_{k=1}^{\infty} w_k\, \delta_{\theta_k^*}$$

[*]equal contribution

*Accepted for the 39th Conference on Uncertainty in Artificial Intelligence* (UAI 2023).

where $w_k = v_k \prod_{j<k}\{1-v_j\}$, $v_k \overset{iid}{\sim} \text{Beta}(1,a)$ and $\theta_k^* \overset{iid}{\sim} G_0$. Fong et al. [2021] then derive the numerator as

$$E\left[\sum_{j=1}^{\infty}\sum_{k=1}^{\infty} w_j\, w_k\, K(x \mid \theta_j^*)\, K(x_1 \mid \theta_k^*)\right]$$
$$= \left(1 - E\left[\sum_{k=1}^{\infty} w_k^2\right]\right) E\left[K(x \mid \theta^*)\right] E\left[K(x_1 \mid \theta^*)\right] + E\left[\sum_{k=1}^{\infty} w_k^2\right] E\left[K(x \mid \theta^*)\, K(x_1 \mid \theta^*)\right]$$

where they have used the fact that $\sum_{k=1}^{\infty} w_k = 1$ almost surely. As $p_0(x) = E\left[K(x \mid \theta^*)\right]$, it follows that (1) can be expressed as

$$1 - \alpha_1 + \alpha_1 \frac{E\left[K(x \mid \theta^*)\, K(x_1 \mid \theta^*)\right]}{p_0(x)\, p_0(x_1)}.$$

for some fixed $\alpha_1$. For R-BP, the kernel $K$ factorises with independent priors on each dimension, and $p_0(x) = \prod_{j=1}^{d} p_0(x^j) = \prod_{j=1}^{d} \mathcal{N}(x^j \mid 0, 1+\tau^{-1})$, so

$$\frac{E\left[K(x \mid \theta^*)\, K(x_1 \mid \theta^*)\right]}{p_0(x)\, p_0(x_1)} = \prod_{j=1}^{d} \frac{E\left[K(x^j \mid \theta^{*j})\, K(x_1^j \mid \theta^{*j})\right]}{p_0(x^j)\, p_0(x_1^j)}. \tag{2}$$

Fong et al. [2021] then show that each univariate term corresponds to the bivariate Gaussian copula density,

$$c(u, v; \rho) = \frac{\mathcal{N}_2\left\{\Phi^{-1}(u), \Phi^{-1}(v) \mid 0, 1, \rho\right\}}{\mathcal{N}\left\{\Phi^{-1}(u) \mid 0, 1\right\} \mathcal{N}\left\{\Phi^{-1}(v) \mid 0, 1\right\}},$$

where $\Phi$ is the normal cumulative distribution function (CDF), and $\mathcal{N}_2$ is the standard bivariate density with correlation parameter $\rho = 1/(1+\tau)$. They then suggest an alternative sequence $h_i$ which iteratively repeats $h_1$, with the key feature that $\alpha_i = (2 - \frac{1}{i})\frac{1}{i+1}$. See Appendix E.1.1. in Fong et al. [2021] for a derivation of this sequence $\alpha_i$.

### A.1.2 With Autoregression (AR-BP)

For the derivation of the AR-BP update, we can follow the arguments in the previous section until (2) where the factorised kernel assumption applies for the first time. For AR-BP, we instead have

$$\frac{E\left[K(x \mid \theta^*)\, K(x_1 \mid \theta^*)\right]}{p_0(x)\, p_0(x_1)} = \prod_{j=1}^{d} \frac{E\left[K\{x^j \mid \theta^{*j}(x^{1:j-1})\}\, K\{x_1^j \mid \theta^{*j}(x^{1:j-1})\}\right]}{p_0(x^j)\, p_0(x_1^j)}. \tag{3}$$

The factorisation of the denominator follows from

$$p_0(x) = E\left[\prod_{j=1}^{d} K\{x^j \mid \theta^{*j}(x^{1:j-1})\}\right] = \prod_{j=1}^{d} E\left[K\{x^j \mid \theta^{*j}(x^{1:j-1})\}\right]$$

as we have independent GP priors on each function $\theta^{*j}$. For notational convenience we write $\{y, x\}$ in place of $\{x^j, x^{1:j-1}\}$ in the following. With the autoregressive kernel assumption, there is the additional complexity

$$E\left[\mathcal{N}\{y \mid \theta(x), 1\}\, \mathcal{N}\{y_1 \mid \theta(x_1), 1\}\right]$$

where $\theta(\cdot) \sim \text{GP}\{0, \tau^{-1}k\}$. The marginal distribution of the GP is normal, so we have

$$[\theta(x), \theta(x_1)]^{\mathrm{T}} \sim \mathcal{N}_2(x, x_1 \mid 0, \Sigma_{x,x_1})$$

where

$$\Sigma_{x,x_1} = \begin{bmatrix} \tau^{-1} & \tau^{-1}k(x, x_1) \\ \tau^{-1}k(x, x_1) & \tau^{-1} \end{bmatrix}.$$

Again from the conjugacy of the normal, we can show that

$$E\left[\mathcal{N}\{y \mid \theta(x), 1\}\mathcal{N}\{y_1 \mid \theta(x_1), 1\}\right] = \mathcal{N}(y, y_1 \mid 0, K_{x,x_1})$$

where

$$K_{x,x_1} = \begin{bmatrix} 1+\tau^{-1} & \tau^{-1}k(x,x_1) \\ \tau^{-1}k(x,x_1) & 1+\tau^{-1} \end{bmatrix}.$$

Here $p_0(y) = E[\mathcal{N}(y|\theta(x))]$ is the same as above, since marginally $\theta(x) \sim \mathcal{N}(0, \tau^{-1})$. Plugging in $y = P_0^{-1}\{\Phi(z)\}$ again gives us the Gaussian copula density with correlation parameter

$$\rho_1(x) = \rho_0 k(x, x_1)$$

for $\rho_0 = 1/(1+\tau)$.

## A.2 DERIVATION OF GAUSSIAN PROCESS POSTERIOR

In this section, we derive the copula sequence for the Gaussian Process, which is fully tractable. This section is mostly for insight, but it would however be interesting to investigate any potential avenues for methodological development.

### A.2.1 First Update Step

We consider a univariate regression setting with $\{y, x\}$. For the GP, we have the model

$$f_\theta(y \mid x) = \mathcal{N}(y \mid \theta(x), \sigma^2), \quad \theta(\cdot) \sim \mathrm{GP}(0, \tau^{-1}k).$$

Like in the above, we can derive the function $h_1(x, x_1)$. Following a similar argument to the AR-BP derivation, the first step GP copula density is

$$\frac{\mathcal{N}_2\left(y, y_1 \mid 0, K_2 + \sigma^2 I\right)}{p_0(y \mid x)p_0(y_1 \mid x_1)}$$

where $K_i$ is the $i \times i$ Gram matrix, with kernel

$$k(x, x') = \tau^{-1} \exp\left\{-0.5(x - x')^2/\ell\right\}.$$

Writing in terms of $P_0$, we have

$$c\left\{P_0(y \mid x), P_0(y_1 \mid x_1); \rho_1(x)\right\}$$

where $c$ is again the Gaussian copula density, but we have the correlation parameter as

$$\rho_1(x) = \frac{\exp\left\{-0.5(x - x_1)^2/\ell\right\}}{1 + \tau\sigma^2}.$$

From this, we can derive the first step of the update scheme:

$$p_1(y \mid x) = c\{P_0(y \mid x), P_0(y_1 \mid x_1); \rho_1(x)\}\, p_0(y \mid x)$$

where $c(u, v; \rho)$ is again the Gaussian copula density, and $p_0(y \mid x) = \mathcal{N}(y; 0, \sigma^2 + \tau^{-1})$.

### A.2.2 All Update Steps

We can even derive the copula update scheme for $i > 1$, as the Gaussian process posterior is tractable. After observing $i - 1$ observations, we have

$$\pi(\theta_x, \theta_{x_i} \mid y_{1:i-1}, x_{1:i-1}) = \mathcal{N}(\mu_{i-1}, \Sigma_{i-1})$$

where each element of $\Sigma_{i-1}$ has the entry

$$k_{i-1}(x, x') = k(x, x') - k(x, x_{1:i-1}) \left[K_{i-1} + \sigma^2 I\right]^{-1} k(x_{1:i-1}, x')$$

where the subscript $i-1$ indicates it is the posterior kernel and $\mu_{i-1}$ is the posterior mean vector of the GP at $x$ and $x_i$. Marginally, the GP copula after $i-1$ data points is

$$\frac{\mathcal{N}_2\left(y, y_i; \mu_{i-1}, \Sigma_{i-1} + \sigma^2 I\right)}{\mathcal{N}\left\{y; \mu_{i-1}^{y}, k_{i-1}(x,x) + \sigma^2\right\} \mathcal{N}\left\{y_{i+1}; \mu_{i-1}^{y_{i-1}}, k_{i-1}(x_i, x_i) + \sigma^2\right\}}$$

where $\mu_{i-1}^{y}$ is the posterior mean of the GP at $x$ and likewise for $\mu_{i-1}^{y_{i-1}}$. This is equivalent to the bivariate Gaussian copula density $c(u, v; \rho_i(x))$, where as before $u = P_{i-1}(y \mid x)$ and $v = P_{i-1}(y_{i+1} \mid x_{i+1})$. The correlation parameter is now

$$\rho_i(x) = \frac{k_{i-1}(x, x_i)}{\sqrt{\{k_{i-1}(x,x) + \sigma^2\}\{k_{i-1}(x_i, x_i) + \sigma^2\}}}$$

In summary, we have the update

$$p_i(y \mid x) = c\{P_{i-1}(y \mid x), P_{i-1}(y_i \mid x_i); \rho_i(x)\}\, p_{i-1}(y \mid x).$$

This gives the same predictives as fitting a full GP. While this update form does not offer any computational gains, it gives us insight into the GP update. The copula update corresponds to the regular normal update [Hahn et al., 2018] with a data-dependent bandwidth $\rho_i(x)$ which measures the distance between $x$ and $x_i$ based on the posterior kernel. A potential interesting direction of research is to seek approximations of the expensive $\rho_i(x)$ to aid with the computation of the GP.

### A.3 INTUITION FOR AR COPULA

As in the main paper, we consider bivariate data, $(x, y)$. As shown in Fong et al. [2021], the update for the conditional density for R-BP takes the form

$$p_i(y \mid x) = [1 - \alpha_i(x, x_i) + \alpha_i(x, x_i)\, c\{P_{i-1}(y \mid x), P_{i-1}(y_i \mid x_i); \rho\}]\, p_{i-1}(y \mid x), \tag{4}$$

where

$$\alpha_i(x, x_i) = \frac{\alpha_i c\{P_{i-1}(x), P_{i-1}(x_i); \rho\}}{1 - \alpha_i + \alpha_i c\{P_{i-1}(x), P_{i-1}(x_i); \rho\}}.$$

To show the effect of the AR update, we make simplifying assumptions to derive the update for the conditional mean function, $\mu_i(x) = \int y\, p_i(y \mid x) dy$. Let us assume that our predictive densities are normally distributed, that is $P_{i-1}(y \mid x) = \mathcal{N}(y \mid \mu_{i-1}(x), \sigma_y^2)$. This is an accurate approximation if the truth is normal and we have observed sufficient observations. Without loss of generalizability, we assume that $\sigma_y^2 = 1$. This then gives the form $P_{i-1}(y \mid x) = \Phi(y - \mu_{i-1}(x))$, which will help us in the calculation of the bivariate Gaussian copula. If we multiply by $y$ and integrate on both sides of (4), we get

$$\mu_i(x) = [1 - \alpha_i(x, x_i)]\mu_{i-1}(x) + \alpha_i(x, x_i) \int c\left(P_{i-1}(y \mid x), P_{i-1}(y_i \mid x_i); \rho\right)\, y\, p_{i-1}(y \mid x)\, dy.$$

Plugging in $P_{i-1}(y \mid x) = \Phi\{y - \mu_{i-1}(x)\}$ (and similarly for the density) to the above gives

$$\int c\left(P_{i-1}(y \mid x), P_{i-1}(y_i \mid x_i); \rho\right)\, y\, dy = \int \frac{\mathcal{N}(y, y_i \mid [\mu_{i-1}(x), \mu_{i-1}(x_i)], 1,\ \rho)}{\mathcal{N}(y_i \mid \mu_{i-1}(x_i), 1)}\, y\, dy.$$

The above is simply the expectation of a conditional normal distribution, giving us

$$\int c\left(P_{i-1}(y \mid x), P_{i-1}(y_i \mid x_i); \rho\right)\, y\, dy = \mu_{i-1}(x) + \rho(y_i - \mu_{i-1}(x_i)).$$

Putting it all together, we thus have

$$\mu_i(x) = \mu_{i-1}(x) + \alpha_i(x, x_i)\rho(y_i - \mu_{i-1}(x_i)).$$

In the autoregressive case, we have

$$\mu_i(x) = \mu_{i-1}(x) + \alpha_i(x, x_i)\rho(x, x_i)(y_i - \mu_{i-1}(x_i)),$$

where we use the notations $\rho_i(x) = \rho(x, x_i)$ interchangeably to highlight the dependence of $\rho$ on the distance between $x$ and $x_i$. Further assuming $P_{i-1}(x) = \mathcal{N}(x \mid 0, 1)$ returns a tractable form for $\alpha_i(x, x')$, giving us Figure 3 in the main paper.

### A.4 DERIVATION OF COPULA UPDATE FOR SUPERVISED LEARNING

We now derive the predictive density update for supervised learning tasks, closely following the derivations of Fong et al. [2021] for the conditional methods in Supplements E.2 and E.3. We assume fixed design points $x_{1:n} \in \mathbb{R}^{n\times d}$ and random response $y_{1:n} \in \mathbb{R}^n$.

#### A.4.1 Conditional Regression with Dependent Stick-Breaking

We follow Appendix E.2.2 in Fong et al. [2021], and derive the regression copula update inspired by the dependent DP. Consider the general covariate-dependent stick-breaking mixture model

$$f_{G_x}(y) = \int \mathcal{N}(y \mid \theta, 1)\, dG_x(\theta), \quad G_x = \sum_{l=1}^{\infty} w_l(x)\, \delta_{\theta_l^*(x)}. \tag{5}$$

For the weights, we elicit the stick-breaking prior $w_l(x) = v_l(x)\prod_{j<l}\{1 - v_j(x)\}$ where $v_l(x)$ is a stochastic process on $\mathcal{X}$ taking values in $[0, 1]$, and is independent across $l$. For the atoms, which are now dependent on $x$, we assume they are independently drawn from a Gaussian process,

$$\theta_l^*(\cdot) \overset{iid}{\sim} \mathrm{GP}(0, \tau^{-1}k),$$

where $k$ is the covariance function. Once again, we want to compute

$$\frac{E\left[f_{G_x}(y)\, f_{G_{x_1}}(y_1)\right]}{p_0(y \mid x)\, p_0(y_1 \mid x_1)}.$$

Following the stick-breaking argument as in Section A.1.1, we can write the numerator as

$$\{1 - \beta_1(x, x_1)\}\, E\left[K\{y \mid \theta^*(x)\}\right] E\left[K\{y_1 \mid \theta^*(x_1)\}\right] + \beta_1(x, x_1) E\left[K\{y \mid \theta^*(x)\}\, K\{y_1 \mid \theta^*(x_1)\}\right]$$

where

$$K\{y \mid \theta^*(x)\} = \mathcal{N}\{y \mid \theta^*(x), 1\}, \quad \theta^*(\cdot) \sim \mathrm{GP}(0, \tau^{-1}k),$$

and

$$\beta_1(x, x_1) = \sum_{k=1}^{\infty} E\left[w_k(x) w_k(x_1)\right].$$

As before, we have

$$\frac{E\left[f_{G_x}(y)\, f_{G_{x_1}}(y_1)\right]}{p_0(y \mid x)\, p_0(y_1 \mid x_1)} = c\left\{P_0(y \mid x), P_0(y_1 \mid x_1); \rho_1(x)\right\}$$

where $\rho_1(x) = \rho_0 k(x, x_1)$ and $\rho_0 = 1/(1 + \tau)$. We thus have the copula density as a mixture of the independent and Gaussian copula density. This then implies the copula update step of the form

$$p_i(y \mid x) = \left[1 - \beta_i(x, x_i) + \beta_i(x, x_i)\, c\left\{P_{i-1}(y \mid x), P_{i-1}(y_i \mid x_i); \rho_i(x)\right\}\right] p_{i-1}(y \mid x),$$

where we write $\rho_i(x) = \rho_i^{d+1}(x)$. As in Fong et al. [2021], we turn to the multivariate update for inspiration where we do not update $P_n(x)$ and instead keep it fixed at $P_0(x) = \Phi(x)$ (for each dimension). This gives us

$$\beta_i(x, x_i) = \frac{\alpha_i \prod_{j=1}^{d} c\left\{\Phi\left(x^j\right), \Phi\left(x_i^j\right); \rho_i^j(x^{1:j-1})\right\}}{1 - \alpha_i + \alpha_i \prod_{j=1}^{d} c\left\{\Phi\left(x^j\right), \Phi\left(x_i^j\right); \rho_i^j(x^{1:j-1})\right\}}. \tag{6}$$

### A.4.2 Classification with Beta-Bernoulli Copula Update

In the classification setting (Appendix E.3.1 in Fong et al. [2021]), Fong et al. [2021] assume a beta-Bernoulli mixture for $y_i \in \{0, 1\}$. As the derivation is written w.r.t $\rho$, we simply replace $\rho$ with our definition of $\rho_i^j(x^{1:j-1})$, giving the update

$$p_i(y \mid x) = (1 - \beta_i(x, x_i) + \beta_i(x, x_i)\, b\,\{q_{i-1}, r_{i-1}; \rho_i(x)\})\, p_{i-1}(y \mid x)$$

where $q_{i-1} = p_{i-1}(y \mid x), r_i = p_{i-1}(y_i \mid x_i)$, $\rho_i(x)$ as in Equation 9, $\beta_i(x, x_i)$ similarly as in (15), and finally the copula-like function $b$ given by

$$b\{q_{i-1}, r_{i-1}; \rho_i(x)\} = \begin{cases} 1 - \rho_i(x) + \rho_i(x)\, \dfrac{q_{i-1} \wedge r_{i-1}}{q_{i-1}\, r_{i-1}} & \text{if } y = y_i \\ 1 - \rho_i(x) + \rho_i(x)\, \dfrac{q_{i-1} - \{q_{i-1} \wedge (1 - r_{i-1})\}}{q_{i-1}\, r_{i-1}} & \text{if } y \neq y_i. \end{cases}$$

# B METHODOLOGY

In this section, we provide more details on the methodology referred to in the main part of the paper.

## B.1 GENERATIVE MODELLING

First, we consider three approaches to generative modelling

1. Inverse sampling
2. Importance sampling
3. Sequential Monte Carlo (SMC)

### B.1.1 Inverse Sampling

**Univariate setting** As noted by Fong et al. [2021], we can sample from $x^* \sim P_n(x)$ by inverse sampling, that is

$$u \sim \mathcal{U}[0, 1], \;\; x^* \sim P_n^{-1}(u).$$

As we cannot evaluate $P_n^{-1}(u)$ directly, we instead solve an optimisation problem

$$x^* = \underset{x}{\operatorname{argmin}} |P_n(x) - u|$$

**Multivariate setting** The univariate procedure can be repeated iteratively in the multivariate setting given the conditional distribution

$$\begin{aligned} &u^1 \sim \mathcal{U}[0, 1],\; x^1 = P_n^{-1}(u^1) \\ &u^2 \sim \mathcal{U}[0, 1],\; x^2 = P_n^{-1}(u^2 \mid x^1) \\ &\dots \\ &u^d \sim \mathcal{U}[0, 1],\; x^d = P_n^{-1}(u^d \mid x^{1:d-1}) \end{aligned}$$

### B.1.2 Importance Sampling

In practice, inverse sampling is unstable and is highly dependent on the performance of the optimization. An alternative approach to data generation is importance sampling. This includes two steps

1. Sampling a set of particles $z_1, \dots, z_B$ from the initial predictive $p_0$.
2. Re-sampling $z_1, \dots, z_B$ with replacement based on the weights $w_1 = p_n(z_1)/p_0(z_1), \dots, w_B = p_n(z_B)/p_0(z_B)$.

### B.1.3 Sequential Monte Carlo

Importance sampling will perform poorly if $p_n$ and $p_0$ are far apart. Instead, we propose a SMC procedure. A similar SMC sampling scheme has been proposed for univariate imputation of censored survival data by Fong and Lehmann [2022]. Here, the goal is parameter inference, and thus only requires *implicit* sample observations by drawing the marginal CDF $u_n^j$ from a uniform distribution. In our case, we generate new *explicit* data directly by sampling from the data space. Please see Algorithm 6 for a complete overview. As this sampling approach is similar to evaluating the density at test data points (Algorithm 5), we highlighted the differences in blue. In short,

1. We sample a set of particles $z_1, \dots, z_B$ from the initial predictive $p_0$, and set the particle weights to $w_k^{[0]} = 1$ for all $k = 1, \dots, B$
2. We update the predictive $p_{i-1} \to p_i$, and the particle weights $w_k^{[i]} = w_k^{[i-1]} \cdot p_i\left(z_k^{[i-1]}\right) / p_{i-1}\left(z_k^{[i-1]}\right)$ for each training observation
3. If the effective sample size (ESS) is smaller than half of the number of particles, we resample $z_1, \dots, z_B$ and $w_1^{[i]}, \dots, w_1^{[B]}$ based on their weights.

Note that particle diversity can be improved by introducing move steps, for example using Markov kernels Chopin [2002], Gunawan et al. [2020].

In Figure 1, we see that inverse sampling struggles on a simple GMM example. On the other hand, importance sampling and SMC provide reasonable samples. Similar sampling schemes have been proposed for Restricted Boltzmann Machines [Larochelle and Murray, 2011, Salakhutdinov and Murray, 2008] where samples can only be drawn from the model approximately by Gibbs sampling.

## B.2 SUPERVISED LEARNING

We briefly recap how joint density estimation can be extended to conditional supervised learning (regression and classification), as outlined by Fong et al. [2021]. Please see Supplement A.4 for the derivation. Given fixed design points $x_{1:n}$ and random response $y_{1:n}$, the problem at hand is to infer a family of conditional densities $\{f_x(y) : x \in \mathbb{R}^d\}$.

### B.2.1 Regression

For the regression case, Fong et al. [2021] posit a Bayesian model with the nonparametric likelihood being a covariate-dependent stick-breaking Dirichlet Process Mixture Model (DPMM):

$$f_{G_x}(y) = \int \mathcal{N}(y \mid \theta, 1)\, dG_x(\theta), \quad G_x = \sum_{k=1}^{\infty} w_k(x)\, \delta_{\theta_k^*}, \tag{7}$$

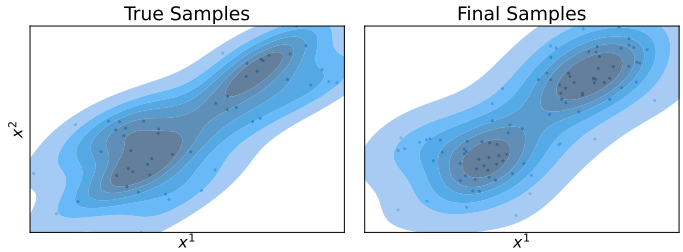
where $w_k(\mathbf{x})$ follows an $x$-dependent stick-breaking process. Our contribution is to assume an autoregressive factorisation of the kernel and independent GP priors on $\theta_k^*$. See Supplement A.4.1 for the derivation of the predictive density update that is now given by

$$p_i(y \mid x) = \left[1 - \beta_i(x, x_i) + \beta_i(x, x_i)\, c\left\{P_{i-1}(y \mid x), P_{i-1}(y_i \mid x_i); \rho_i(x)\right\}\right] p_{i-1}(y \mid x), \tag{8}$$

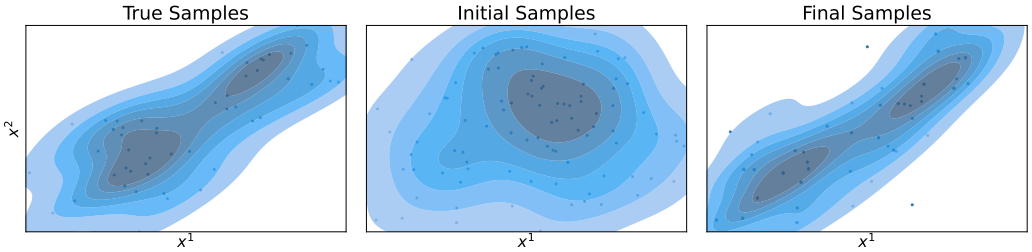
where $\rho_i(x) = \rho_i^{d+1}(x)$ and $\beta$ as in (6).

### B.2.2 Classification

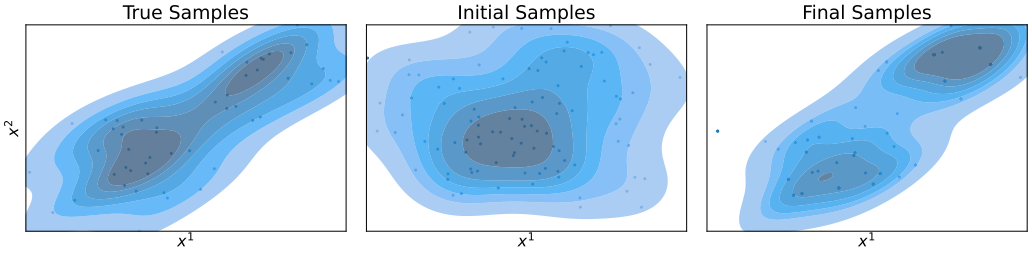
For $y_i \in \{0, 1\}$, Fong et al. [2021] assume a beta-Bernoulli mixture. As explained in Supplement A.4.2 and Fong et al. [2021], this gives the same update as in the regression setting with the difference that the copula $c$ in (8) is




(a) Inverse Sampling




(b) Importance Sampling




(c) Sequential Monte Carlo

Figure 1: 100 samples generated from AR$_d$-BP trained on 50 samples from a GMM with 4 components. All three sampling approaches manage to preserve the multi-modal data distribution.

replaced with

$$b\{q_{i-1}, r_{i-1}; \rho_i(x)\} = \begin{cases} 1 - \rho_i(x) + \rho_i(x) \dfrac{q_{i-1} \wedge r_{i-1}}{q_{i-1}\, r_{i-1}} & \text{if } y = y_i \\ 1 - \rho_i(x) + \rho_i(x) \dfrac{q_{i-1} - \{q_{i-1} \wedge (1 - r_{i-1})\}}{q_{i-1}\, r_{i-1}} & \text{if } y \neq y_i, \end{cases}$$

where $\rho_i(x) = \rho_i^{d+1}(x), q_{i-1} = p_{i-1}(y \mid \mathbf{x}), r_{i-1} = p_{i-1}(y_i \mid \mathbf{x}_i)$ and $\rho_y \in (0, 1)$.

### B.3 IMPLEMENTATION DETAILS

Please see Algorithm 1 for the full estimation procedure, Algorithm 2 for the optimisation of the bandwidth parameters, Algorithm 4 for the fitting procedure of the predictive density updates, and eventually Algorithm 5 for the steps during test-time inference. All algorithms are written for one specific permutation of the dimensions, and are repeated for different permutations.

Note that at both training time and test time, we need to consider the updates on the scale of the CDFs, that is for the terms such as $u_i^j(x^j)$, which appear in the update step (8). Given

$$u_i^j(x^j) = P_i(x^j | x^{1:j-1}) = \int_{-\infty}^{x^j} p_i(x^{1:j-1}, x'^j) / p_i(x^{1:j-1}) dx'^j,$$

and (8), the CDFs $u_i^j(x^j)$ take on the tractable update

$$u_i^j = \left\{ (1 - \alpha_i) u_{i-1}^j + \alpha_i H\left(u_{i-1}^j, v_{i-1}^j; \rho_i^j\right) \prod_{r=1}^{k-1} c\left(u_{i-1}^r, v_{i-1}^r; \rho_i^r\right) \right\} \frac{p_{i-1}\left(x^{1:k-1}\right)}{p_i\left(x^{1:k-1}\right)}, \tag{9}$$

and set $v_{i-1}^j = u_{i-1}^j(x_i)$ which holds by definition, where we dropped the argument $x$ for simplicity from $\rho_i^j$ and $u_i^j$, and $H(u, v; \rho)$ denotes the conditional Gaussian copula distribution with correlation $\rho$, that is

$$H(u, v; \rho) = \int_0^u c(u', v; \rho) du' = \Phi\left\{ \frac{\Phi^{-1}(u) - \rho \Phi^{-1}(v)}{\sqrt{1 - \rho^2}} \right\}.$$

The Gaussian copula density $c(u, v; \rho)$ is given by

$$c(u, v; \rho) = \frac{\mathcal{N}_2\left\{\Phi^{-1}(u), \Phi^{-1}(v) \mid 0, 1, \rho\right\}}{\mathcal{N}\{\Phi^{-1}(u) \mid 0, 1\} \mathcal{N}\{\Phi^{-1}(v) \mid 0, 1\}},$$

where $\Phi$ is the normal CDF, and $\mathcal{N}_2$ is the standard bivariate density with correlation $\rho \in (0, 1)$.

**Ordering** Note that the predictive density update depends on the ordering of both the training data and the dimensions. This permutation dependence is not an additional assumption on the data generative process, and the only implication is that the subset of ordered marginal distributions continue to satisfy (5) (main paper). In the absence of a natural ordering of the training samples or the dimensions, we take multiple random permutations, observing in practice that the resulting averaged density estimate performs better. More precisely, for a given permutation of the dimensions, we first tune the bandwidth parameters, and then calculate density estimates based on multiple random permutations of the training data. We then average over each of the resulting estimates to obtain a single density estimate for each dimension permutation, and subsequently take the average across these estimates to obtain the final density estimate. Importantly, our method is parallelizable over permutations and thus able to exploit modern multi-core computing architectures.

**Algorithm 1** Full density estimation pipeline

**Input:**

$x_{1:n}$: training observations;
$x_{n+1:n+n'}$: test observations;
$M$: number of permutations over samples and features to average over;
$n_\rho$: number of train observations used for the optimisation of bandwidth parameters; **Output:**
$p_n(x_{n+1}), \dots p_n(x_{n+n'})$: density of test points

**procedure** FULL_DENSITY_ESTIMATION
    Compute optimal bandwidth parameters ▷ $\mathcal{O}(Mn_\rho^2 d \cdot \#\text{gradient steps})$
    Compute $v_i^{j,(m)}$ for $i \in \{1, \dots, n\}, j \in \{1, \dots, d\}, m \in \{1, \dots, M\}$ ▷ $\mathcal{O}(Mn^2 d)$
    Evaluate density at test observations $x_{n+1:n+n'}$ ▷ $\mathcal{O}(Mnn'd)$
**end procedure**

**Algorithm 2** Estimate optimal bandwidth parameters

**Input:**

$x_{1:n}$: training observations;
$M$: number of permutations over samples and features to average over;
$n_\rho$: number of train observations used for the optimisation of bandwidth parameters;
`maxiter`: number of iterations;

$\mathcal{R}^{(0)}$: initialisation of bandwidth parameters:
-$\mathcal{R}^{(0)} = \{\rho_0^{(0)}, l_1^{(0)}, \dots, l_{d-1}^{(0)}\}$ (by default, $\rho_0^{(0)} \leftarrow 0.9, l_1^{(0)} \leftarrow 1, \dots, l_{d-1}^{(0)} \leftarrow 1$) for AR-BP,
-$\mathcal{R}^{(0)} = \{\rho_0^{(0)}, w\}$ (by default, $\rho_0^{(0)} \leftarrow 0.9$, and $w$ initialised as implemented in `Haiku` by default) for ARnet-BP

**Output:**

$\mathcal{R}^{(\texttt{maxiter})}$: optimal bandwidth parameters

**procedure** OPTIMAL_BANDWIDTH_AND_LENGTHSCALES
    Subsample $\{x'_1, \dots, x'_{n_\rho}\}$ from $x_{1:n}$
    **for** $s \leftarrow 1$ **to** `maxiter` **do**
        $\_, \{p_{i-1}^{(m)}(x'_i)\}_{i,m} \leftarrow$ FIT_CONDITIONAL_PREDICTIVE_CDF($\mathcal{R}^{(s-1)}, \{x'_1, \dots, x'_{n_\rho}\}, M$, fit_density=True)
        Compute $L(x'_1, \dots, x'_{n_\rho}) = -\sum_{m=1}^{M} \sum_{i=1}^{n_\rho} \log p_{i-1}^{(m)}(x'_i)$
        $\mathcal{R}^{(s)} \leftarrow$ ADAM_STEP($\mathcal{R}^{(s-1)}, L$)
    **end for**
    **return** $\mathcal{R}^{(s)}$
**end procedure**

**Algorithm 3** Single copula update

**Input:**

$z$: observation to update the log density;
$x_i$: observation to update with;
$i$: sample index;
$j$: feature index;
$u_{i-1}^{j}(z)$: predictive CDF for $z$;
$v_{i-1}^{j,(m)}$: prequential CDF;
$\rho_i^j(z^{1:j-1})$=None: bandwidth;
$\mathcal{R}$=None: bandwidth parameters;

**Output:**

$u_i(z)$

1: **procedure** CDF_Update

2: Compute

$$\rho_i^j(z^{1:j-1}) \leftarrow \rho_0 k_{\mathcal{R}}\left(z^{1:j-1}, x_i^{1:j-1}\right)$$

where $k_{\mathcal{R}}$ denotes the user-defined kernel if $\rho$ =None

3: Compute the bivariate Gaussian copula density

$$c\{u_{i-1}^{j}(z), v_{i-1}^{j,(m)}; \rho_j\} \leftarrow \frac{\mathcal{N}_2\left\{\Phi^{-1}(u_{i-1}^{j}(z)), \Phi^{-1}(v_{i-1}^{j,(m)}) \mid 0, 1, \rho_i^j(z^{1:j-1})\right\}}{\mathcal{N}\{\Phi^{-1}(u_{i-1}^{j}(z)) \mid 0, 1\}\mathcal{N}\{\Phi^{-1}(v_{i-1}^{j,(m)}) \mid 0, 1\}}$$

4: Compute the conditional Gaussian copula CDF

$$H\left\{u_{i-1}^{j}(z), v_{i-1}^{j,(m)}; \rho_i^j(z^{1:j-1})\right\} \leftarrow \Phi\left\{\frac{\Phi^{-1}(u_{i-1}^{j}(z)) - \rho_i^j(z^{1:j-1})\Phi^{-1}(v_{i-1}^{j,(m)})}{\sqrt{1-\rho_i^j(z^{1:j-1})^2}}\right\}$$

5: Compute $\alpha_i = (2 - \frac{1}{i})\frac{1}{i+1}$

6: Compute $u_i^j(z) = P_i^j(z|z^{1:j-1})$ by

$$u_i^j(z) \leftarrow \left\{(1-\alpha_i)u_{i-1}^{j}(z) + \alpha_i H\left(u_{i-1}^{j}(z), v_{i-1}^{j,(m)}; \rho_i^j(z)\right)\prod_{l=1}^{j-1} c\left(u_{i-1}^{l}(z), v_{i-1}^{l,(m)}; \rho_i^j(z)\right)\right\}$$
$$\cdot 1 \Big/ \left\{1 - \alpha_i + \alpha_i \prod_{j=1}^{d} c\left(u_{i-1}^{j}(z), v_{i-1}^{j,(m)}; \rho_i^j(z)\right)\right\}$$

7: **return** $u_i(z)$

8: **end procedure**

**Algorithm 4** Estimate prequential CDFs at train observations

**Input:**

$\mathcal{R}$: bandwidth parameters
$x_{1:n}$: training observations;
$M$: number of permutations over features to average over;
compute_density (by default, False);

**Output:**

$\{v_{i-1}^{j,(m)}\}_{i,j,m}, \{p_{i-1}^{(m)}(x_i)\}_{i,m}$ if compute_density, else $\{v_{i-1}^{j,(m)}\}_{i,j,m}$

**procedure** FIT_CONDITIONAL_PREDICTIVE_CDF
  **for** $m \leftarrow 1$ **to** $M$ **do**
    Sample permutation $\pi_1 \in \Pi(n), \pi_2 \in \Pi(d)$

    Change the ordering of the training observations $\{x_1^{(m)}, \ldots, x_n^{(m)}\} \leftarrow \{\pi_1(x_1), \ldots, \pi_1(x_n)\}$ and the features $x \leftarrow [\pi_2(x^1), \ldots, \pi_2(x^d)]$ ▷ For simplicity we will drop the superscript in the following

    **for** $j \leftarrow 1$ **to** $d$ **do**
      **for** $k \leftarrow 1$ **to** $n$ **do**
        Initialise $u_0^j(x_k) \leftarrow \Phi(x_k^j)$ ▷ $u$ also depends on the permutation $m$, but since we do not reuse $u$ after $m$ is updated, we drop the index for simplicity
      **end for**
    **end for**

    **for** $i \leftarrow 1$ **to** $n$ **do**
      Set $v_{i-1}^{j,(m)} \leftarrow u_{i-1}^j(x_i)$ **for** $j \leftarrow 1$ **to** $d$
      **for** $k \leftarrow 1$ **to** $i$ **do**
        **for** $j \leftarrow 1$ **to** $d$ **do**
          $u_i^j(x_k)$=CDF_UPDATE($x_k$, $x_i$, $i$, $j$, $u_{i-1}^j(x_k)$, $v_{i-1}^{j,(m)}$, $\mathcal{R}$)
        **end for**
        **if** compute_density **then**

$$p_i^{(m)}(x_k) \leftarrow \left\{ 1 - \alpha_i + \alpha_i \prod_{j=1}^{d} c\left(u_{i-1}^j(x_k), v_{i-1}^{j,(m)}; \rho_i^j(x_k)\right) \right\} p_{i-1}^{(m)}(x_k)$$

        **end if**
      **end for**
    **end for**
  **end for**
  **return** $\{v_{i-1}^{j,(m)}\}_{i,j,m}, \{p_{i-1}^{(m)}(x_i)\}_{i,m}$ **if** compute_density **else** $\{v_{i-1}^{j,(m)}\}_{i,j,m}$
**end procedure**

**Algorithm 5** Evaluate density at test observations

**Input:**

$\mathcal{R}$: bandwidth parameters

$x_{n+1:n+n'}$: test observations;

$\{\{x_1^{(1)}, ..., x_n^{(1)}\}, \ldots, \{x_1^{(M)}, ..., x_n^{(M)}\}\}$: sets of permuted train observations;

$\{v_i^{j,(m)}\}_{i,j,m}$: prequential conditional CDFs at train observations;

$M$: number of observations over features to average over;

**Output:**

$\{p_n(x_{n+1}), \ldots, p_n(x_{n+n'})\}$

**procedure** EVAL_DENSITY
  **for** $m \leftarrow 1$ **to** $M$ **do**
    **for** $j \leftarrow 1$ **to** $d$ **do**
      **for** $k \leftarrow 1$ **to** $n'$ **do**
        Initialise $u_0^j(x_{n+k}) \leftarrow \Phi(x_{n+k}^j)$
      **end for**
    **end for**

    **for** $i \leftarrow 1$ **to** $n$ **do**
      **for** $k \leftarrow 1$ **to** $n'$ **do**
        **for** $j \leftarrow 1$ **to** $d$ **do**
          $u_i^j(x_k)$=CDF_UPDATE($x_{n+k}$, $x_i$, $i$, $j$, $u_{i-1}^j(x_{n+k})$, $v_{i-1}^{j,(m)}$, $\mathcal{R}$)
        **end for**
        Compute density

$$p_i^{(m)}(x_{n+k}) \leftarrow \left\{1 - \alpha_i + \alpha_i \prod_{j=1}^{d} c\left(u_{i-1}^j(x_{n+k}), v_{i-1}^{j,(m)}; \rho_i^j(x_{n+k})\right)\right\} p_{i-1}^{(m)}(x_{n+k})$$

      **end for**
    **end for**
  **end for**

  ▷ Average density over permutations
  **for** $i \leftarrow n+1$ **to** $n+n'$ **do**
    $p_n(x_i) \leftarrow \frac{1}{M} \sum_{m=1}^{M} p_n^{(m)}(x_i)$
  **end for**
  **return** $\{p_n(x_{n+1}), \ldots, p_n(x_{n+n'})\}$
**end procedure**

**Algorithm 6** Sample new observations

**Input:**

$\mathcal{R}$: bandwidth parameters

$\{z_1^{[0]}, \ldots, z_B^{[0]}\}$: initial samples from proposal distribution;

$\{q(z_1^{[0]}), \ldots, q(z_B^{[0]})\}$: proposal density evaluated at initial samples;

$\{\{x_1^{(1)}, ..., x_n^{(1)}\}, \ldots, \{x_1^{(M)}, ..., x_n^{(M)}\}\}$: sets of permuted train observations;

$\{v_i^{j,(m)}\}_{i,j,m}$: prequential conditional CDFs at train observations;

**Output:**

$\{z_1^{[n]}, \ldots, z_B^{[n]}\}$ and $\{p_n(z_1^{[n]}), \ldots, p_n(z_B^{[n]})\}$

**procedure** SAMPLE
  **for** $m \leftarrow 1$ **to** $M$ **do**
    **for** $k \leftarrow 1$ **to** $B$ **do**
      **for** $j \leftarrow 1$ **to** $d$ **do**
        Initialise $u_0^j(z_k^{[0]}) \leftarrow \Phi(z_k^{[0]})$
      **end for**
      Initialise $w_k^{[0]} \leftarrow p_0(z_k^{[0]})/q(z_k^{[0]})$
    **end for**
  **end for**

  **for** $i \leftarrow 1$ **to** $n$ **do**
    **for** $k \leftarrow 1$ **to** $B$ **do**
      **for** $m \leftarrow 1$ **to** $M$ **do**
        **for** $j \leftarrow 1$ **to** $d$ **do**
          $u_i^j(x_k)$=CDF_UPDATE($z_k^{[i-1]}$, $x_i$, $i$, $j$, $u_{i-1}^j(z_k^{[i-1]})$, $v_{i-1}^{j,(m)}$, $\mathcal{R}$)
        **end for**
        Compute density

$$p_i^{(m)}\left(z_k^{[i-1]}\right) \leftarrow \left\{1 - \alpha_i + \alpha_i \prod_{j=1}^{d} c\left(u_{i-1}^j(x_{n+k}), v_{i-1}^{j,(m)}; \rho_i^j(z_k^{[i-1]})\right)\right\} p_{i-1}^{(m)}\left(z_k^{[i-1]}\right)$$

      **end for**
      $p_i(z_k^{[i-1]}) \leftarrow \frac{1}{M}\sum_{m=1}^{M} p_i^{(m)}(z_k^{[i-1]})$
      $w_k^{[i]} \leftarrow w_k^{[i-1]} \cdot p_i\left(z_k^{[i-1]}\right)/p_{i-1}\left(z_k^{[i-1]}\right)$
    **end for**
    ESS $\leftarrow \left(\sum_k w_k^{[i]}\right)^2 / \left(\sum_k w_k^{[i]^2}\right)$
    **if** ESS $< 0.5 \cdot B$ **then**
      $\{z_k^{[i]}\}_k \leftarrow$RESAMPLE_WITH_REPLACEMENT$\left(\{z_k^{[i-1]}\}_k, \{w_k^{[i]}\}_k\right)$
      $w_k^{[i]} \leftarrow 1$ **for** $k = 1, \ldots, B$
    **else**
      $z_k^{[i]} \leftarrow z_k^{[i-1]}$ **for** $k = 1, \ldots, B$
    **end if**
  **end for**
  **return** $\{z_k^{[i]}\}_k$
**end procedure**

# C EXPERIMENTS

## C.1 EXPERIMENTAL DETAILS

The UCI data sets [Asuncion and Newman, 2007] we used are: wine, breast, parkinson (PARKIN), ionosphere (IONO), boston housing (BOSTON), concrete (CONCR), diabetes (DIAB), and digits.

**Code** We downloaded the code for MAF and NSF from `https://github.com/bayesiains/nsf`, and the code for R-BP from `https://github.com/edfong/MP/tree/main/pr_copula`, and implemented EarlyStopping with patience 50, and 200 minimal, and 2000 maximal iterations. Note that we chose the autoregressive version of RQ-NSF over the coupling variant as the former seemed to generally outperform the latter in Durkan et al. [2019]. The neural network in ARnet-BP was implemented with `Haiku` [Hennigan et al., 2020]. The remaining methods are implemented in `sklearn`. For the DPMM with VI (mean-field approximation), we use both the diagonal and full covariance function, with default hyperparameters for the priors. The code used to generate these results is available as an additional supplementary directory.

**Initialisation** We initialise the predictive densities with a standard normal, the bandwidth parameter with $\rho_0 = 0.9$, the length scales with $l_2 = 1, ..., l_{d-1} = 1$, and the neural network weights inside ARnet-BP by sampling from a truncated normal with variance proportional to the number of input nodes of the layer.

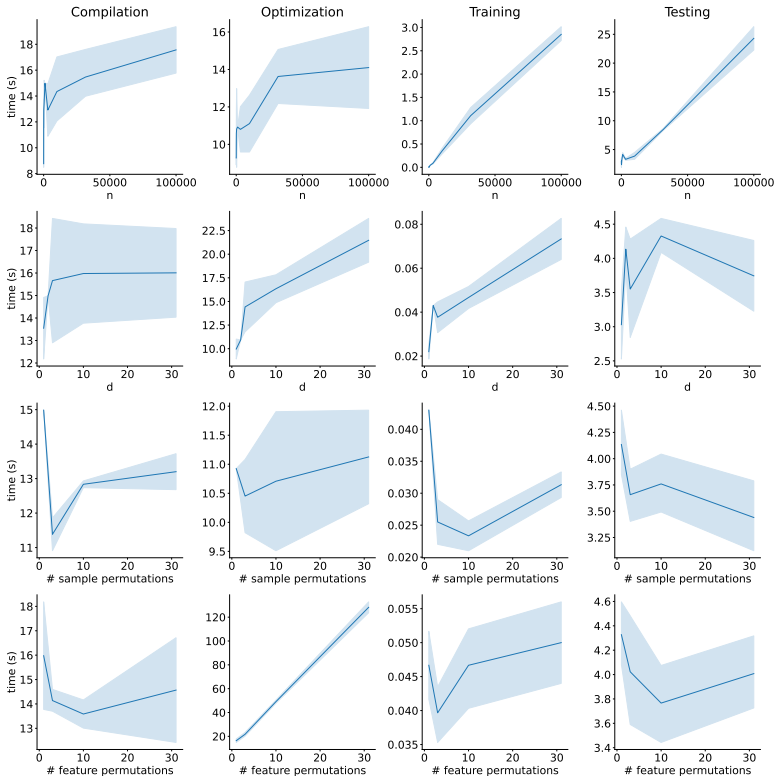
**Data pre-processing** For each dataset, we standardized each of the attributes by mean-centering and rescaling to have a sample standard deviation of one. Following Papamakarios et al. [2017], we eliminated discrete-valued attributes. To avoid issues arising from collinearity, we also eliminated one attribute from each pair of attributes with a Pearson correlation coefficient greater than 0.98.

**Hyperparameter tuning** We average over $M = 10$ permutations over samples and features. The bandwidth of the kernel density estimations (KDEs) was found by five-fold cross validation over a grid of 80 log-scale-equidistant values from $\rho = 0.1$ to 100. For the DPMM, we considered versions with a diagonal (Diag) and full (Full) covariance matrix for each mixture component. We optimized over the weight concentration prior of the DPMM by five-fold cross validation with values ranging from $10^{-40}$ to 1. The model was trained with variational inference using `sklearn`. The hyperparameters of masked autoregressive flows (MAFs) and rational-quadratic neural spline flows (RQ-NSFs) were found with a Bayesian optimisation search. For MAF and RQ-NSF, we applied a Bayesian optimisation search over the learning rate $\{3 \cdot 10^{-4}, 4 \cdot 10^{-4}, 5 \cdot 10^{-4}\}$, the batch size $\{512, 1024\}$, the flow steps $\{10, 20\}$, the hidden features $\{256, 512\}$, the number of bins $\{4, 8\}$, the number of transform blocks $\{1, 2\}$ and the dropout probability $\{0, 0.1, 0.2\}$. On each data set, the hyperparameter search ran for more than 5 days. Please see Table 1 for the optimal parameters found. For the benchmark UCI data sets, we did not tune the hyperparameters for neither MAF nor RQ-NSF but instead used the standard parameters given by Durkan et al. [2019]. The kernel parameters of the Gaussian process (GP) are optimised during training, the $\alpha$ resp. $\lambda$ intialization parameter of the linear model over the range from 1 to 2 resp. 0.01 to 0.1, and the hidden layer sizes of the MLP over the values $\{64, 128, 256\}$.

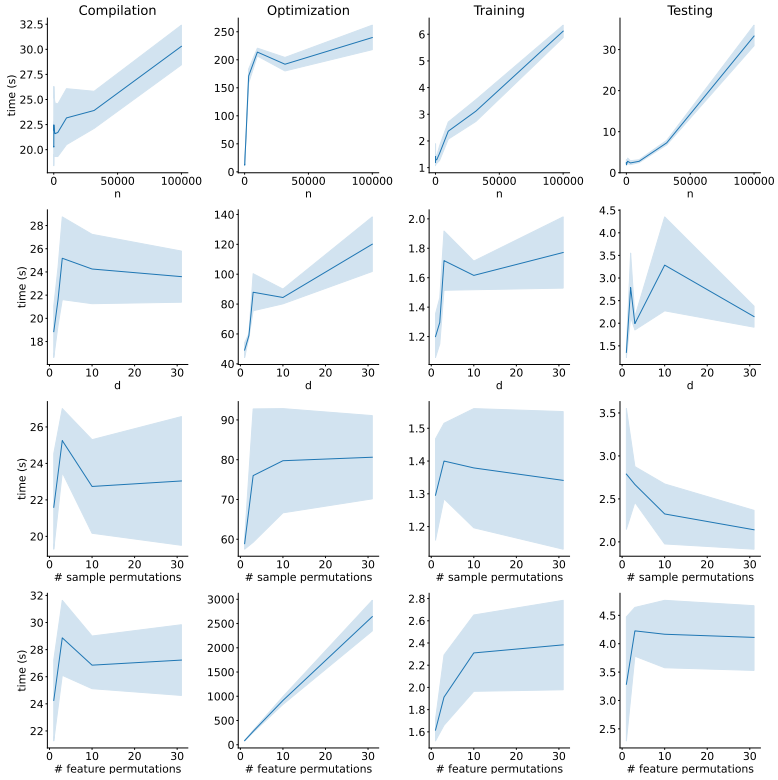
**Compute** We run all BP and neural network experiments on a single Tesla V100 GPU, as provided in the internal cluster of our department. In total, these experiments required compute of approximately 4000 GPU hours. The remaining experiments were run on a single core of an Intel(R) Xeon(R) Gold 6240 CPU @ 2.60GHz, using up a total of 100 hours.

## C.2 ADDITIONAL EXPERIMENTAL RESULTS

**Computational analysis** For the computational study, we consider data sampled from a Gaussian mixture model (GMM). By default, we set the number of training samples to $n = 500$, the number of test samples to $n' = 500$, the number of features to $d = 2$, the number of mixture components to $K = 2$, and the number of feature and samples permutations to 1. In Figure 2, we plot the compute in elapsed seconds w.r.t changes in these parameters.




(a) AR-BP




(b) ARnet-BP

Figure 2: Computational study: computational time measured in elapsed seconds for a simple GMM example. Note that R-BP has the same computational complexity and only saves an indiscernible constant time factor.

Table 1: Hyperparameters for MAF and RQ-NSF

| | data | batch size | learning rate | flow steps | hidden nodes | bins | transform blocks | dropout |
|---|---|---|---|---|---|---|---|---|
| MAF | WINE | 10000 | 0.0003 | 20 | 512 | - | 1 | 0.2 |
| | BREAST | 10000 | 0.0004 | 20 | 512 | - | 1 | 0.2 |
| | PARKINSONS | 10000 | 0.0004 | 20 | 512 | - | 1 | 0.2 |
| | IONOSPHERE | 10000 | 0.0003 | 20 | 512 | - | 1 | 0.2 |
| | BOSTON | 10000 | 0.0003 | 10 | 512 | - | 1 | 0.2 |
| | CONCRETE | 1024 | 0.0003 | 10 | 512 | - | 1 | 0.2 |
| | DIABETES | 10000 | 0.0004 | 20 | 512 | - | 1 | 0.2 |
| | CHECKERBOARD | 10000 | 0.0003 | 20 | 512 | - | 1 | 0.2 |
| RQ-NSF | WINE | 10000 | 0.0004 | 20 | 512 | 8 | 1 | 0.2 |
| | BREAST | 10000 | 0.0005 | 10 | 512 | 8 | 1 | 0.2 |
| | PARKINSONS | 10000 | 0.0005 | 20 | 512 | 8 | 1 | 0.2 |
| | IONOSPHERE | 10000 | 0.0003 | 10 | 512 | 8 | 1 | 0.2 |
| | BOSTON | 10000 | 0.0003 | 10 | 512 | 8 | 1 | 0.2 |
| | CONCRETE | 1024 | 0.0004 | 20 | 256 | 8 | 2 | 0.1 |
| | DIABETES | 256 | 0.0004 | 10 | 512 | 8 | 2 | 0.2 |
| | CHECKERBOARD | 1024 | 0.0004 | 10 | 512 | 8 | 2 | 0.1 |

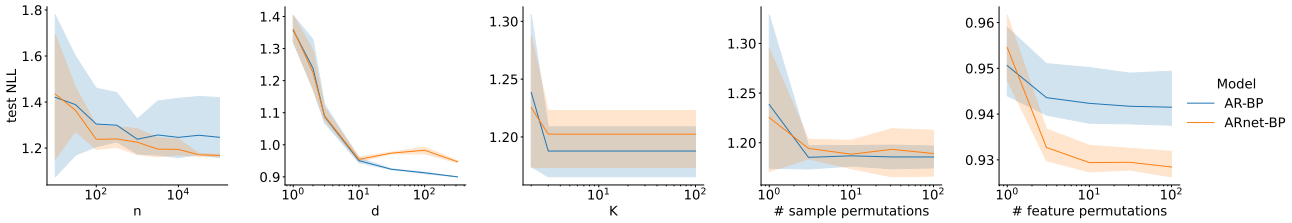



Figure 3: Sensitivity analyis: Average test NLL over 5 runs reported with standard error for a simple GMM example over a range of simulation and parameter settings.

**Sensitivity analysis** For the sensitivity study, we consider the same simulated GMM data as in the computational study, and plot the results in Figure 3. As expected, we observe that the test negative log-likelihood (NLL) decreases in $n$, and in the number of permutations. It also decreases in the number of mixture components. One possible explanation for this is that, as noted by Hahn et al. [2018], R-BP can be interpreted as a mixture of $n$ normal distributions. The NLL decreases in $d$, as the mixture components are easier to distinguish in higher dimensional covariate spaces.

**Ablation study** Figure 3 shows the test NLL of ARnet-BP and AR-BP for the above GMM example, as a function of the number of sample permutations, and number of feature permutations. We see that averaging over multiple permutations is crucial to the performance of AR-BP. In Table 2, we also show results on the small UCI datasets for:

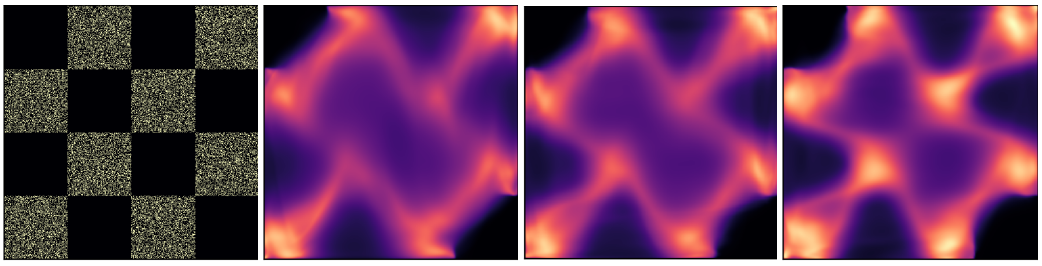
- a different choice of covariance function, namely a rational quadratic covariance function, defined by $k(x, x_i) = \left(1 + \frac{||x-x_i||_2^2}{2\gamma\ell^2}\right)^{-\gamma}$, where $\ell, \gamma > 0$ and
- a different choice of initial distribution, namely a uniform distribution (unif).

We observe that none of these ablations consistently outperforms $\text{AR}_d$-BP.

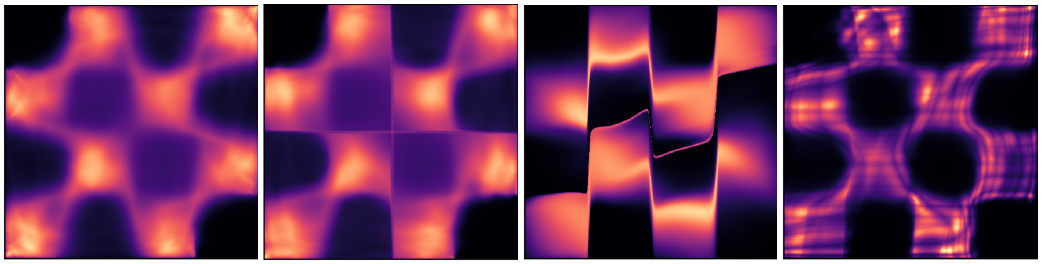
**Benchmark UCI data sets** As we only presented a subset of the results on the benchmark data sets introduced by Papamakarios et al. [2017] in Section 5, we present more results for density estimation on the complete data set in Table 3. These results underscore that 1) MAF and RQ-NSF outperform any other baseline, the more data is available; 2) KDE underperforms in high-dimensional settings; 3) DPMM is not suitable for every data distribution. Note that evaluation of the R-BP variants take at least 4 days to run on any of the data sets with

Table 2: Average NLL with standard error over five runs on five UCI data sets of small-to-moderate size

| | WINE | BREAST | PARKIN | IONO | BOSTON |
|---|---|---|---|---|---|
| n/d | 89/12 | 97/14 | 97/16 | 175/30 | 506/13 |
| $AR_d$-BP | $\mathbf{13.22_{\pm 0.04}}$ | $\mathbf{6.11_{\pm 0.04}}$ | $\mathbf{7.21_{\pm 0.12}}$ | $\mathbf{16.48_{\pm 0.26}}$ | $-14.75_{\pm 0.89}$ |
| AR-BP (RQ) | $13.53_{\pm 0.02}$ | $7.39_{\pm 0.06}$ | $8.79_{\pm 0.08}$ | $21.26_{\pm 0.08}$ | $4.49_{\pm 0.00}$ |
| $AR_d$-BP (RQ) | $13.36_{\pm 0.04}$ | $6.18_{\pm 0.03}$ | $7.85_{\pm 0.08}$ | $20.25_{\pm 0.09}$ | $\mathbf{-20.41_{\pm 1.28}}$ |
| $AR_d$-BP (unif) | $-5.18_{\pm 0.04}$ | $-15.51_{\pm 0.11}$ | $-16.58_{\pm 0.06}$ | $-47.77_{\pm 3.77}$ | $-10.73_{\pm 1.63}$ |



(a) True data (b) R-BP (c) $R_d$-BP (d) AR-BP



(e) $AR_d$-BP (f) ARnet-BP (g) MAF (h) RQ-NSF

Figure 4: Scatter plot and density estimates of 60,000 observations sampled from a chessboard data distribution. Test log likelihoods are R-BP: $2.25_{\pm 0.0}$, $R_d$-BP : $2.19_{\pm 0.0}$, AR-BP: $2.21_{\pm 0.0}$, $AR_d$-BP: $2.10_{\pm 0.0}$, ARnet BP : $2.19_{\pm 0.0}$, MAF : $2.09_{\pm 0.0}$, RQ-NSF : $2.05_{\pm 0.0}$.

more than 800,000 observations which is why we omitted those results here.

Table 3: Average NLL with standard error over five runs on benchmark UCI data from Papamakarios et al. [2017]

| | POWER | GAS | HEPMASS | MINIBOONE | BSDS300 |
|---|---|---|---|---|---|
| n/d | 1,659,917/6 | 852,174/8 | 315,123/21 | 29,556/43 | 1,000,000/ 63 |
| Gaussian | $7.73_{\pm 0.00}$ | $3.59_{\pm 0.00}$ | $27.93_{\pm 0.00}$ | $37.20_{\pm 0.00}$ | $56.45_{\pm 0.00}$ |
| KDE | $29.39_{\pm 0.00}$ | $\mathbf{-9.61_{\pm 0.00}}$ | $26.44_{\pm 0.00}$ | $43.88_{\pm 7.52}$ | $63.70_{\pm 10.00}$ |
| DPMM (Diag) | $0.51_{\pm 0.01}$ | $1.20_{\pm 0.02}$ | $25.80_{\pm 0.00}$ | $39.16_{\pm 0.01}$ | $37.55_{\pm 0.02}$ |
| DPMM (Full) | $0.33_{\pm 0.00}$ | $-5.57_{\pm 0.04}$ | $23.40_{\pm 0.02}$ | $18.82_{\pm 0.01}$ | $4.47_{\pm 0.00}$ |
| MAF | $0.52_{\pm 0.00}$ | $-2.21_{\pm 0.54}$ | $21.10_{\pm 0.04}$ | $12.81_{\pm 0.08}$ | $2.76_{\pm 0.17}$ |
| RQ-NSF | $\mathbf{0.00_{\pm 0.01}}$ | $-6.41_{\pm 0.14}$ | $\mathbf{19.46_{\pm 0.08}}$ | $\mathbf{12.51_{\pm 0.19}}$ | $\mathbf{2.44_{\pm 0.56}}$ |

**Image examples** We provide preliminary results on two image datasets, digits and MNIST, in Table 4. Note that the AR-BP copula updates investigated here were not designed with computer vision tasks in mind. The rich parameterization allows the model to overfit to the data leading to a prequential negative log-likelihood of at least -684 at train time while the test NLL is considerably higher. ARnet-BP, on the other hand, helps to model the complex data structure more efficiently. We expect that further extensions based on, for instance, convolutional covariance functions [Van der Wilk et al., 2017] may prove fruitful.

**Toy examples** Figure 4 shows density estimates for the introductory example of the checkerboard distribution in a large data regime. We observe that neural-network-based methods outperform the AR-BP alternatives. Nevertheless, AR-BP performs better than the baseline R-BP. An illustration of this behaviour on another

Table 4: Image datasets: average test NLL over five runs displayed with standard error

| | DIGITS | MNIST |
|---|---|---|
| MAF | $\mathbf{-8.76_{\pm 0.10}}$ | $-7.14_{\pm 0.48}$ |
| RQ-NSF | $-6.17_{\pm 0.13}$ | $-8.49_{\pm 0.03}$ |
| R-BP | $-8.80_{\pm 0.00}$ | $-9.04_{\pm 0.07}$ |
| $\text{R}_d$-BP | $-7.46_{\pm 0.12}$ | $-7.73_{\pm 0.07}$ |
| AR-BP | $-8.66_{\pm 0.03}$ | $-7.31_{\pm 42.54}$ |
| $\text{AR}_d$-BP | $-7.46_{\pm 0.18}$ | $-8.32_{\pm 61.92}$ |
| ARnet-BP | $-7.72_{\pm 0.28}$ | $\mathbf{-9.20_{\pm 0.10}}$ |

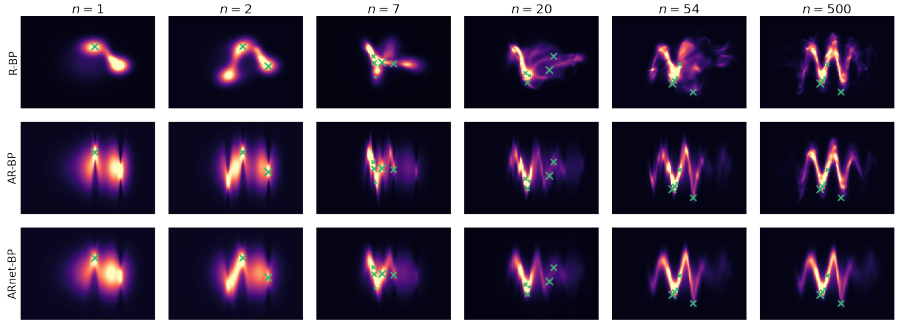



Figure 5: Illustration of the importance of an autoregressive kernel. We trained the models on 500 data points sampled according to a sine wave distribution (given in Figure 6). We visualise the predictive density after observing a different number, $n$, of observations, highlighting the last five points with ×. We observe that for highly non-linear relationships between $x^1$ and $x^2$, the optimal bandwidth of R-BP is quite high ($\rho = 0.93$) which results in strong overfitting. Even when we choose $\rho_0 = 0.93$ for AR-BP and ARnet-BP, we observe that these models learn the true data distribution with fewer samples than R-BP does.

toy example is also given in Figure 5. Figure 6 shows density estimates from AR-BP on a number of complex distributions.

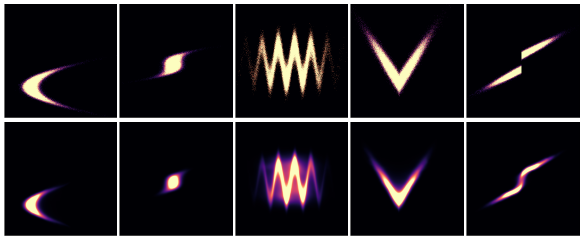


Figure 6: Scatter plots of 60,000 samples from different data distributions in the first row, and corresponding autoregressive predictive density estimates in the second row.

## REFERENCES


Arthur Asuncion and David Newman. UCI machine learning repository, 2007.

Nicolas Chopin. A sequential particle filter method for static models. *Biometrika*, 89(3):539–552, 2002.

Conor Durkan, Artur Bekasov, Iain Murray, and George Papamakarios. Neural spline flows. *Advances in neural information processing systems*, 32, 2019.

Edwin Fong and Brieuc Lehmann. A predictive approach to bayesian nonparametric survival analysis. In Gustau Camps-Valls, Francisco J. R. Ruiz, and Isabel Valera, editors, *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics*, volume 151 of *Proceedings of Machine Learning Research*, pages 6990–7013. PMLR, 28–30 Mar 2022. URL `https://proceedings.mlr.press/v151/fong22a.html`.

Edwin Fong, Chris Holmes, and Stephen G Walker. Martingale posterior distributions. *To appear at the Journal of the Royal Statistical Society: Series B (with discussion)*, 2021.

David Gunawan, Khue-Dung Dang, Matias Quiroz, Robert Kohn, and Minh-Ngoc Tran. Subsampling sequential monte carlo for static bayesian models. *Statistics and Computing*, 30(6):1741–1758, 2020.

P Richard Hahn, Ryan Martin, and Stephen G Walker. On recursive Bayesian predictive distributions. *Journal of the American Statistical Association*, 113(523):1085–1093, 2018.

Tom Hennigan, Trevor Cai, Tamara Norman, and Igor Babuschkin. Haiku: Sonnet for JAX, 2020. URL `http://github.com/deepmind/dm-haiku`.

Hugo Larochelle and Iain Murray. The neural autoregressive distribution estimator. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 29–37. JMLR Workshop and Conference Proceedings, 2011.

George Papamakarios, Theo Pavlakou, and Iain Murray. Masked autoregressive flow for density estimation. *Advances in neural information processing systems*, 30, 2017.

Ruslan Salakhutdinov and Iain Murray. On the quantitative analysis of deep belief networks. In *Proceedings of the 25th international conference on Machine learning*, pages 872–879, 2008.

Mark Van der Wilk, Carl Edward Rasmussen, and James Hensman. Convolutional gaussian processes. *Advances in Neural Information Processing Systems*, 30, 2017.

## Statement of Authorship for joint/multi-authored papers for PGR thesis

To appear at the end of each thesis chapter submitted as an article/paper

The statement shall describe the candidate's and co-authors' independent research contributions in the thesis publications. For each publication there should exist a complete statement that is to be filled out and signed by the candidate and supervisor **(only required where there isn't already a statement of contribution within the paper itself).**

| | |
|---|---|
| Title of Paper | Quasi-Bayesian nonparametric density estimation via autoregressive predictive updates |
| Publication Status | xPublished □ Accepted for Publication<br>□Submitted for Publication □Unpublished and unsubmitted work written in a manuscript style |
| Publication Details | Ghalebikesabi, S., Holmes, C. C., Fong, E., & Lehmann, B. (2023, July). Quasi-Bayesian nonparametric density estimation via autoregressive predictive updates. In Uncertainty in Artificial Intelligence (pp. 658-668). PMLR. |

### Student Confirmation

| | | | |
|---|---|---|---|
| Student Name: | Sahra Ghalebikesabi | | |
| Contribution to the Paper | I conceived the implementation and experiments, and took the lead in the writing. The idea and derivation were conceived by Edwin Fong. During frequent meetings, my collaborators helped out with helpful suggestions on methodology and feedback. They also contributed to writing, checking the results, and proof-reading. | | |
| Signature |  | Date | 28.10.2023 |

### Supervisor Confirmation

By signing the Statement of Authorship, you are certifying that the candidate made a substantial contribution to the publication, and that the description described above is accurate.

| | | |
|---|---|---|
| Supervisor name and title: Chris Holmes | | |
| Supervisor comments<br><br>I am in agreement with the description of the contributions. | | |
| Signature | Date | 28.10.2023 |

This completed form should be included in the thesis, at the end of the relevant chapter.

# 4

# Deep Generative Pattern-Set Mixture Models for Nonignorable Missingness

# Deep Generative Pattern-Set Mixture Models for Nonignorable Missingness

**Sahra Ghalebikesabi**
University of Oxford

**Rob Cornish**
University of Oxford

**Luke J. Kelly**
CEREMADE, CNRS
Université Paris-Dauphine

**Chris Holmes**
University of Oxford

## Abstract

We propose a variational autoencoder architecture to model both ignorable and nonignorable missing data using pattern-set mixtures as proposed by Little (1993). Our model explicitly learns to cluster the missing data into missingness pattern sets based on the observed data and missingness masks. Underpinning our approach is the assumption that the data distribution under missingness is probabilistically semi-supervised by samples from the observed data distribution. Our setup trades off the characteristics of ignorable and nonignorable missingness and can thus be applied to data of both types. We evaluate our method on a wide range of data sets with different types of missingness and achieve state-of-the-art imputation performance. Our model outperforms many common imputation algorithms, especially when the amount of missing data is high and the missingness mechanism is nonignorable.

## 1 INTRODUCTION

Missing data is a ubiquitous problem in real-world applications. In imaging, for instance, inpainting algorithms are used to restore damaged images or to remove selected objects (Bertalmio et al., 2000). In medical applications, some patient records never get collected because of missed appointments or because information acquisition was too cost intensive. In other cases, some of the data may have simply been lost. Since the 1970s, a rapidly growing body of literature has developed imputation algorithms that appropriately fill in the missing data. Missing data is commonly classified into three categories: missing completely at random (MCAR), missing at random (MAR) and missing not at random (MNAR) (Rubin, 1976). When data are MCAR, the missingness is independent of both the observed and missing variables. Data are MAR when the missingness depends on the observed variables only. And they are said to be MNAR when the missingness mechanism depends not only on the observed variables but also on the missing values. Imputing MNAR data bears the highest difficulties as it requires modeling the missing data mechanism. It is thus not sufficiently analyzed in the current machine learning literature.

In this paper, we exemplary explain some existing deep generative imputation methods in the framework of nonignorable missingness models (Rubin, 1976). We propose a VAE-based imputation algorithm derived from one of these nonignorable missingness models, the pattern-set mixture model, which was introduced by Little (1993). Our model encodes the nonignorable missing data mechanism in a latent variable. To prevent underidentification, we impose probabilistic semi-supervision. This approach prevents overfitting to the observed data. An additional benefit of our method is that it explicitly provides meaningful clusters of the data that can be exploited to cluster the population based on the similarity of their missing data mechanism. We show that the model achieves state-of-the-art performance on a variety of data sets. Furthermore, we provide an implementation of our algorithm and all corresponding experiments online.[2]

## 2 RELATED WORK

Common imputation approaches include the EM algorithm (Dempster et al., 1977), MICE (Buuren and Groothuis-Oudshoorn, 2010), matrix completion (Mazumder et al., 2010) and MissForest (Stekhoven

Proceedings of the 24[th] International Conference on Artificial Intelligence and Statistics (AISTATS) 2021, San Diego, California, USA. PMLR: Volume 130. 

[2] `https://github.com/sghalebikesabi/PSMVAE`

and Bühlmann, 2012). With the recent development of deep generative models, there has been an increase in generative imputation algorithms. Deep generative models such as generative adversarial nets (GANs) (Goodfellow et al., 2014) or variational autoencoders (VAEs) (Kingma and Welling, 2013; Rezende et al., 2014) have been used for missingness imputation since their introduction.

One of the most popular examples is GAIN (Yoon et al., 2018). It uses a generative adversarial network in which the generator takes the observed data vector as an input and outputs an imputed data vector. The discriminator determines which variables were actually observed based on partial information of the true missingness mask. Another GAN-based imputation algorithm, MisGAN (Li et al., 2019), learns one generator and one discriminator solely for the missingness mask, and simultaneously learns a separate generator and discriminator for the data and the missingness mask.

Nazabal et al. (2020) introduce an objective function in order to directly train VAEs with a Gaussian Mixture prior on data with missingness. MIWAE (Mattei and Frellsen, 2019) is an importance-weighted autoencoder that achieves state-of-the-art imputation performance by maximizing a tight lower bound of the log-likelihood of the observed data.

All these imputation algorithms have in common that they do not incorporate assumptions on the missingness mechanisms and thus cannot be applied when the data is MNAR. To tackle this problem, Ipsen et al. (2021) introduced an extension of MIWAE, the not-MIWAE, that handles MNAR data by explicitly modeling the conditional distribution of the missingness mask given the data. Such an approach is, however, not robust to the misspecification of the missingness mechanism. Collier et al. (2020) develop a latent variable model that considers the observed data to be the result of a corruption process which depends on a binary missingness mask and can be applied on data with ignorable and nonignorable missingness. Nonetheless, the model architecture has to be chosen dependent on the underlying missingness mechanism. Recently, Sportisse et al. (2020) have shown the identifiability of the parameters of probabilistic principal components analysis for a class of MNAR mechanisms. Missingness not at random has also been tackled using discriminative approaches such as matrix completion in the recommender system literature (Wang et al., 2019).

# 3 PROBLEM FORMULATION

Let $\mathbf{x} = (x_1, ..., x_d)$ be a random variable taking values in the $d$-dimensional space $\mathcal{X} = \mathcal{X}_1 \times ... \times \mathcal{X}_d$. We further assume that the missingness mask $\mathbf{m}$ is a random variable defined on $\{0, 1\}^d$. The missingness mask is defined such that $x_j$ is observed for $m_j = 1$, and missing otherwise. We denote the joint distribution of $\mathbf{x}$ and $\mathbf{m}$ by $P_\theta(\mathbf{x}, \mathbf{m})$. Following Yoon et al. (2018), we also introduce a random variable $\mathbf{x_{obs}} = (x_{obs,1}, ..., x_{obs,d})$ which takes values in $\mathcal{X}_{obs} = (\mathcal{X}_1 \cup \{*\}) \times \cdots \times (\mathcal{X}_d \cup \{*\})$ where $*$ is a point not in $\mathcal{X}_1 \cup .... \cup \mathcal{X}_d$ and represents unobserved data points. The random variable $\mathbf{x_{obs}}$ is then defined by

$$x_{obs,j} = \begin{cases} x_j, & \text{if } m_j = 1 \\ *, & \text{otherwise.} \end{cases}$$

Building upon this, we introduce another random variable $\mathbf{x_{mis}}$ with $x_{mis,j} = x_j$ if $m_j = 0$ and $x_{mis,j} = *$ otherwise. We can now retrieve $\mathbf{x}$ as

$$\mathbf{x} = \mathbf{m} \odot \mathbf{x_{obs}} + (\mathbf{1} - \mathbf{m}) \odot \mathbf{x_{mis}}, \tag{1}$$

where $\odot$ denotes the Hadamard product. Note that both $\mathbf{x_{mis}}$ and $\mathbf{x_{obs}}$ are defined to be $d$-dimensional random variables. We can interpret such an approach by imagining self-masked missingness, which means that the missingness probability of each covariate only depends on the covariate itself such that $P_\theta(m_j|\mathbf{x}) = P_\theta(m_j|x_j)$. Then $x_{mis,j}$ denotes the outcome of that covariate under the treatment $m_j = 0$ and $x_{obs,j}$ the outcome under the treatment $m_j = 1$. When we want to refer to only the observed or missing observations, we instead write $\mathbf{x'_{obs}} = \{x_j \mid m_j = 1 \text{ for } j \in \{1, ..., d\}\}$ and $\mathbf{x'_{mis}} = \{x_j \mid m_j = 0 \text{ for } j \in \{1, ..., d\}\}$.

In the imputation setting, we assume that we are given $n$ i.i.d. copies $\mathbf{x^1_{obs}}, ..., \mathbf{x^n_{obs}}$ of $\mathbf{x_{obs}}$. From these observations, we can infer the missingness masks $\mathbf{m^1}, ...., \mathbf{m^n}$ as realizations of $\mathbf{m}$. We write the observed data as $\mathcal{D} = \{\mathbf{x^1_{obs}}, ..., \mathbf{x^n_{obs}}, \mathbf{m^1}, ...., \mathbf{m^n}\}$. When we impute the missing data, we aim to recover $P_\theta(\mathbf{x_{mis}}|\mathbf{x_{obs}}, \mathbf{m})$.

# 4 NONIGNORABLE MISSINGNESS MODELS

When data are MCAR (i.e. $P_\theta(\mathbf{x}, \mathbf{m}) = P_\theta(\mathbf{x})P_\theta(\mathbf{m})$) or MAR (i.e. $P_\theta(\mathbf{x}, \mathbf{m}) = P_\theta(\mathbf{x})P_\theta(\mathbf{m}|\mathbf{x'_{obs}})$), we can maximize the data likelihood without modelling the missing-data mechanism since

$$\begin{aligned} P_\theta(\mathbf{x'_{obs}}, \mathbf{m}) &= \int P_\theta(\mathbf{x'_{obs}}, \mathbf{x'_{mis}}) P_\theta(\mathbf{m}|\mathbf{x'_{obs}}, \mathbf{x'_{mis}}) d\mathbf{x'_{mis}} \\ &= P_\theta(\mathbf{x'_{obs}}) P_\theta(\mathbf{m}|\mathbf{x'_{obs}}). \end{aligned}$$

When data are MNAR, the missing-data mechanism is nonignorable and has to be modeled within a maximum likelihood framework. Little and Rubin (2019) differentiate three ways of modelling the joint distribution of $\mathbf{x}$ and $\mathbf{m}$ in this case.

*Selection models* factorize the joint distribution as $P_\theta(\mathbf{x}, \mathbf{m}) = P_\theta(\mathbf{x})P_\theta(\mathbf{m}|\mathbf{x})$. This factorization goes along with intuitive missing-data mechanisms: In the MNAR case, the complete data $x$ can be seen as the cause why some variables are missing. Not-MIWAE (Ipsen et al., 2021) can be categorized as a selection model that uses MIWAE to model $P_\theta(\mathbf{x})$ and an additional layer (through a neural network layer or a logistic regression) on top of $P_\theta(\mathbf{x})$ to model $P_\theta(\mathbf{m}|\mathbf{x})$. GAIN (Yoon et al., 2018) makes an MCAR assumption in the setting of a selection model. As such it imputes the missing values by modeling the data distribution $P_\theta(\mathbf{x})$ using a neural network with one hidden layer, and then estimates $P_\theta(\mathbf{m}|\mathbf{x}, \mathbf{h})$ where $\mathbf{h}$ is a hint variable that indicates what $\mathbf{m}$ looks like.

*Pattern mixture models* factorize the joint distribution as $P_\theta(\mathbf{x}, \mathbf{m}) = P_\theta(\mathbf{x}|\mathbf{m})P_\theta(\mathbf{m})$, where $P_\theta(\mathbf{m})$ is a categorical distribution and $P_\theta(\mathbf{x}, \mathbf{m})$ is as a result a mixture of distributions. The drawback of such a parameterization is that $P_\theta(\mathbf{x}|\mathbf{m})$ is conditional on a high-dimensional categorical variable whose categories are often not completely observed which leads to the distribution of the missing data being underidentified without any additional assumptions (Little, 1993). Despite this, pattern mixture models are applied when there is an interest in $P_\theta(\mathbf{x}|\mathbf{m})$. For instance, a drug company might be interested in predicting the lab values of patients that dropped out of a clinical trial. Collier et al. (2020) propose a latent variable model that optimizes the lower bound of $P_\theta(\mathbf{x_{obs}}|\mathbf{m})$ and as such constitutes a special type of pattern mixture model.

In order to combine the benefits of both factorizations, Little (1993) introduced *pattern-set mixture models*. These models have an additional latent variable $\mathbf{r}$ with realizations in $\{1, ..., k\}$ that clusters the missingness patterns into $k$ missingness pattern-sets. In each missingness pattern-set, the missing data mechanism is modeled using a selection model. The joint distribution can then be written as

$$P_\theta(\mathbf{x}, \mathbf{m}, \mathbf{r}) = P_\theta(\mathbf{r})P_\theta(\mathbf{x}|\mathbf{r})P_\theta(\mathbf{m}|\mathbf{x}, \mathbf{r}). \tag{2}$$

For $k = 1$ this model reduces to a selection model. Compared to pattern mixture models, it requires fewer parameters (because we are borrowing strengths across the clusters), is less prone to underidentification and has thus more statistical power. Regardless of this, it still allows us to cluster the population into interesting categories. In a clinical trial, there might be some patients that dropped out for reasons unrelated to the study, while other patients dropped out because of adverse reactions to the treatment. The objective is that the missingness pattern-set $\mathbf{r}$ corresponds to these different missingness mechanisms. It summarizes similar missingness patterns and thus simplifies downstream tasks. The HIVAE model (Nazabal et al., 2020), a Gaussian Mixture VAE, can be seen as a pattern-set mixture model for data that are MAR and where $P_\theta(\mathbf{m}|\mathbf{x}, \mathbf{r})$ is ignored in the maximum likelihood inference.

Wu and Carroll (1988) propose *shared-parameter models* which build upon the assumption that there exists a latent random variable $\mathbf{z} \in \mathbb{R}^b$ for some $b < n$ conditional on which the missing model and the data model are independent: $P_\theta(\mathbf{x}, \mathbf{m}|\mathbf{z}) = P_\theta(\mathbf{x}|\mathbf{z})P_\theta(\mathbf{m}|\mathbf{z})$.

These specifications are equivalent only when the data is MCAR (Little and Rubin, 2019). The choice of the model should be made based on the underlying data problem. We argue that in the case of high-dimensional data sets, such as images, where machine learning algorithms proved to be more useful than classical statistical approaches, pattern mixture models are not feasible considering the high-dimensional space of $\mathbf{m}$. As selection models fall within the class of pattern-set mixture models, we use a combination of pattern-set mixture models and shared-parameter models for building a deep generative pattern-set mixture model.

# 5 DEEP GENERATIVE PATTERN-SET MIXTURE MODEL

We now introduce an imputation approach that combines ideas of variational autoencoders and pattern-set mixture models. In contrast to other machine learning imputation methods such as HIVAE (Nazabal et al., 2020) or MIWAE (Mattei and Frellsen, 2019), we thus aim to model the joint distribution $P_\theta(\mathbf{x}, \mathbf{m})$ instead of the marginal $P_\theta(\mathbf{x})$.

## 5.1 Generative Model

We will now define a generative model for $P_\theta(\mathbf{x}, \mathbf{m})$. More specifically, we will model $P_\theta(\mathbf{x_{obs}}, \mathbf{m}, \mathbf{x_{mis}})$ where we assume for now that $\mathbf{x_{mis}}$ is a latent variable. Then, $P_\theta(\mathbf{x}, \mathbf{m})$ follows from Equation 1. Assuming the pattern-set mixture model holds, we introduce an additional latent categorical variable $\mathbf{r}$ which groups the missingness patterns into sets. Following Equation 2 and assuming that $P_\theta(\mathbf{m}|\mathbf{x_{obs}}, \mathbf{x_{mis}}, \mathbf{r}) =$

$P_\theta(\mathbf{m}|\mathbf{x},\mathbf{r})$, we can now write the joint distribution as

$$P_\theta(\mathbf{x_{obs}},\mathbf{m},\mathbf{x_{mis}},\mathbf{r}) = P_\theta(\mathbf{r})P_\theta(\mathbf{x_{obs}},\mathbf{x_{mis}}|\mathbf{r})P_\theta(\mathbf{m}|\mathbf{x},\mathbf{r}).$$

Since $\mathbf{r}$ is a categorical variable that only captures the pattern-set of an observation, we introduce an additional continuous latent variable $\mathbf{z}$ that models the latent interaction of $\mathbf{x_{mis}}$ and $\mathbf{x_{obs}}$. Given $\mathbf{z}$ and $\mathbf{r}$, we then assume the joint of $\mathbf{x_{mis}}$ and $\mathbf{x_{obs}}$ to fully factorize implying

$$P_\theta(\mathbf{x_{mis}},\mathbf{x_{obs}}) = \prod_{j=1}^{d} P_\theta(x_{mis,j}|\mathbf{z},\mathbf{r})P_\theta(x_{obs,j}|\mathbf{z},\mathbf{r}).$$

If additional information on the missing data mechanism $P_\theta(\mathbf{m}|\mathbf{x},\mathbf{r})$ is available, we can write the generative model as

$$P_\theta(\mathbf{x_{obs}},\mathbf{m},\mathbf{x_{mis}},\mathbf{z},\mathbf{r}) = P_\theta(\mathbf{m}|\mathbf{x},\mathbf{r})P_\theta(\mathbf{x_{obs}}|\mathbf{r},\mathbf{z})P_\theta(\mathbf{x_{mis}}|\mathbf{r},\mathbf{z})P_\theta(\mathbf{z}|\mathbf{r})P_\theta(\mathbf{r}), \quad (3)$$

as is visualized in Figure 1a. While Ipsen et al. (2021) only show how one uniform missing model can be parameterized for the whole population, our approach allows to account for different missingness models. This can be interesting when part of the data is assumed to be MCAR while some observations can be MNAR.

We parameterize the generative model of the VAE using a neural network for improved data fit. Allowing the missing model $P_\theta(\mathbf{m}|\mathbf{x},\mathbf{r})$ to be parameterized by a neural network has the disadvantage that the fit of $P_\theta(\mathbf{x_{obs}},\mathbf{x_{mis}}|\mathbf{r})$ suffers from the flexibility of the missing model. This statement is strengthened by the empirical results of Ipsen et al. (2021). Since for any MNAR model there is an MAR model with equal fit to the observed data, it is not possible to test for MNAR without any further assumptions on the missing data mechanism (Molenberghs et al., 2008). For this reason, it is important to choose an imputation algorithm that finds a trade off between flexibility of the missingness model and the distortion it induces into the data model when the data are MCAR. We find that this dilemma can be solved by the assumption made when using shared parameter models that $\mathbf{x_{obs}},\mathbf{x_{mis}}$ and $\mathbf{m}$ are independent conditional on the pattern-set indicator $\mathbf{r}$ and the continuous latent representation $\mathbf{z}$. Such a model has been proven to be more robust to model specification (Harel and Schafer, 2009). When no additional information on $P_\theta(\mathbf{m}|\mathbf{x})$ is available, we thus formalize the generative model as

$$P_\theta(\mathbf{x_{obs}},\mathbf{m},\mathbf{x_{mis}},\mathbf{z},\mathbf{r}) = P_\theta(\mathbf{r})P_\theta(\mathbf{z}|\mathbf{r})P_\theta(\mathbf{x_{mis}}|\mathbf{r},\mathbf{z})P_\theta(\mathbf{x_{obs}}|\mathbf{r},\mathbf{z})P_\theta(\mathbf{m}|\mathbf{r},\mathbf{z}), \quad (4)$$

as illustrated in Figure 1b. The missing values are imputed by sampling from $P_\theta(\mathbf{x_{mis}}|\mathbf{x_{obs}},\mathbf{m}) = P_\theta(\mathbf{x_{obs}}|\mathbf{r},\mathbf{z})$ which follows from the conditional independence of $\mathbf{x_{obs}},\mathbf{m}$ and $\mathbf{x_{mis}}$ given $\mathbf{r}$ and $\mathbf{z}$.

### 5.2 Probabilistic Semi-Supervision

We use likelihood-based variational inference to learn the joint distribution $P_\theta(\mathbf{x},\mathbf{m})$. As we do not observe the missing data, we can however only optimize for the parameters of $P_\theta(\mathbf{x_{obs}},\mathbf{m})$. A sensible choice is to treat $\mathbf{x_{mis}}$ as a latent factor learned from $\mathbf{x_{obs}}$ and $\mathbf{m}$ (Nazabal et al., 2020). Without any further assumptions, the distribution of $\mathbf{x_{mis}}$ would be underidentified given the observed data as, among other reasons, any orthogonal transformation of $\mathbf{x_{mis}}$ would not affect the observed data likelihood $P_\theta(\mathbf{x_{obs}},\mathbf{m})$ (Khemakhem et al., 2020).

We can regularize the problem and introduce smoothness through a novel process involving information sharing and semi-supervision (Kingma et al., 2014; Joy et al., 2020). For any covariate of any sample $x^i_{obs,j}$ with $j \in \{1,...,d\}$ and $i \in \{1,...,n\}$, we assume $x^i_{obs,j}$ is not only an observation of $x_{obs,j}$, but also of $x_{mis,j}$ with probability $1-\pi_{i,j}$ if $m_{i,j}=1$. Put simply, we sample each $x^i_{mis,j}$ from

$$y_{i,j}P_\theta(\widetilde{x}_{mis,j}) + (1-y_{i,j})\mathbb{1}(x^i_{obs,j}),$$

where $\mathbb{1}$ is the indicator function, $\widetilde{x}_{mis,j}$ is a latent auxiliary variable that describes the unobserved dynamics of the missing data, and $y_j$ is an independent Bernoulli random variable with known success probability $\pi'$ if $m_j=1$, and with probability 1 otherwise. We then define the augmented data set as

$$\mathcal{D}_{\boldsymbol{\pi}}(\mathbf{y^1},...,\mathbf{y^n}) := \mathcal{D} \cup \{x^i_{mis,j}; y^i_j = 0\}_{i,j}. \quad (5)$$

For simplicity, let us assume for now that we observe a single univariate data point $x^0_{obs,0}$ with $m^0_0 = 0$ such that $\mathcal{D} = \{x^0_{obs,0}, m^0_0\}$. With probability $1-\pi'$, it holds that $y^0_0 = 0$. We then assume that $x^0_{mis,0}$ is also observed with value $x^0_{obs,0}$ and the augmented data set is thus $\mathcal{D}_\pi(y^0_0=0) = \{x^0_{obs,0}, m^0_0, x^0_{mis,0}\}$. In this case, we maximize the likelihood $P_\theta(x_{obs,0}, m_0, x_{mis,0}|\mathcal{D}_\pi(y^0_0=0))$. With probability $\pi'$, however, it holds that $y^0_0=1$ and $x^0_{mis,0}$ is assumed to be unobserved. We then have $\mathcal{D}_\pi(y^0_0=1) = \{x^0_{obs,0}, m^0_0\} = \mathcal{D}$. We now maximize the likelihood $P_\theta(x_{obs,0}, m_0|\mathcal{D}_\pi(y^0_0=1))$. Since we know the true distribution of $y^0_0$, we can also marginalize out $y_0$ and maximize the weighted likelihood

$$\pi' P_\theta(x_{obs,0}, m_0|\mathcal{D}_\pi(y^0_0=1)) + (1-\pi')P_\theta(x_{obs,0}, m_0, x_{mis,0}|\mathcal{D}_\pi(y^0_0=0)).$$

In a more general setting, we can write the expected

likelihood given the augmented data set as

$$\begin{aligned}&\mathbb{E}_{\mathbf{y}}[P_\theta(\mathbf{x_{obs}}, \mathbf{x_{mis,1-y}}, \mathbf{m}|\mathcal{D}_{\boldsymbol{\pi}}(\mathbf{y}))]\\&\quad= \boldsymbol{\pi} P_\theta(\mathbf{x_{obs}}, \mathbf{m}|\mathcal{D}_{\boldsymbol{\pi}}(\mathbf{1}))\\&\qquad+ (\mathbf{1}-\boldsymbol{\pi})P_\theta(\mathbf{x_{obs}}, \mathbf{m}, \mathbf{x_{mis}}|\mathcal{D}_{\boldsymbol{\pi}}(\mathbf{0})),\end{aligned}$$

where $\mathbf{x_{mis,1-y}} := \{x_{mis,j}|y_j = 0 \text{ for } j \in \{1, ..., d\}\}$, and $\mathbf{1}$ and $\mathbf{0}$ are $d$-vectors of ones and zeros respectively. We only assume semi-supervision for the covariates $x_{mis,j}|(m_j = 1) \in \{*\}$ which drop out in the generation process of $x_j$ (1). The parameter $\pi$ can thus be interpreted as confidence on the ignorability of the missing model: the greater $\pi$ is, the less likely $x_{mis,j}$ stems from an observed distribution. Note that this approach is equivalent to a biased data augmentation approach and that we do not modify the generative model here. As proven in the supplements, it holds:

**Proposition 1.** *Assume that we observe data $\mathcal{D}$ sampled from one of the generative models defined according to* (3) *or* (4)*. For $\pi' < 1$, it holds that the distributional parameter $\mu$ of $\mathbf{x_{mis}}|\mathbb{E}_{\mathbf{y}}(\mathcal{D}_\pi(\mathbf{y}))) \sim \mathcal{N}(\mu, c)$ for some fixed constant $c$ and $\mathcal{D}_\pi(\mathbf{y})$ as defined in* (5) *is identifiable under the maximum likelihood estimation method.*

Without any additional model assumptions such as the specification of the missing model, the distribution of $\widetilde{\mathbf{x}}_{\mathbf{mis}}$ will be underidentified and the estimator of $\mathbf{x_{mis}}$ will be biased. In that case, the question arises why we should not just assume $P_\theta(\mathbf{x_{mis}}) = P_\theta(\mathbf{x_{obs}})$ from the beginning. We argue that our unconventional approach to model both $\mathbf{x_{obs}}$ and $\mathbf{x_{mis}}$ as $d$-dimensional vectors and include such a semi-supervised approach introduces smoothing and prevents overfitting to the observed data distribution which is especially beneficial when missingness is high. A similar methodology has been proposed by Szegedy et al. (2016) under the term label-smoothing regularization. Contrary to our approach, Nazabal et al. (2020) assume the union of $\mathbf{x_{obs}}$ and $\mathbf{x_{mis}}$ to be $d$-dimensional. In other words, they assume that only the missing components of $\mathbf{x}$ are latent. As a result the missing data imputation of the HIVAE model can be seen as a special case of our model for $\pi' = 0$.

### 5.3 Recognition Model

As already noted, the generative model is learned by maximum likelihood inference. The marginal likelihood of the observed variables, $P_\theta(\mathbf{x_{obs}}, \mathbf{m}, \mathbf{x_{mis,1-y}})$, is intractable but we can follow Kingma and Welling (2013) and define a recognition model $Q_\phi$ for modelling the unobserved latent variables $\mathbf{x_{mis,y}}, \mathbf{z}$ and $\mathbf{r}$ given the observed observations $\mathbf{x_{obs}}, \mathbf{m}$ and $\mathbf{x_{mis,1-y}}$ assuming a simple parametric form:

$$\begin{aligned}Q' :=& Q_\phi(\mathbf{x_{mis,y}}, \mathbf{z}, \mathbf{r}|\mathbf{x_{obs}}, \mathbf{m}, \mathbf{x_{mis,1-y}})\\=& Q_\phi(\mathbf{r}|\mathbf{x_{obs}}, \mathbf{m}) Q_\phi(\mathbf{z}|\mathbf{r}, \mathbf{x_{obs}}, \mathbf{m}, \mathbf{x_{mis,1-y}})\\&\times Q_\phi(\mathbf{x_{mis,y}}|\mathbf{z}, \mathbf{r}, \mathbf{x_{mis,1-y}}, \mathbf{x_{obs}}, \mathbf{m}).\end{aligned}$$

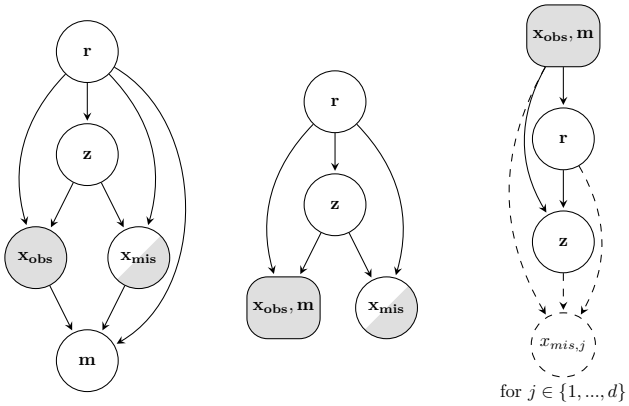
Using Jensen's equality it follows that

$$\begin{aligned}&\log P_\theta(\mathbf{x_{obs}}, \mathbf{m}, \mathbf{x_{mis,1-y}})\\&\quad\geq \mathbb{E}_{Q'}[-\log Q' + \log P_\theta(\mathbf{x_{obs}}, \mathbf{m}, \mathbf{x_{mis}}, \mathbf{z}, \mathbf{r})].\end{aligned}$$

We can thus fit the model by maximizing this evidence lower bound (ELBO).

Since the data sampled for $\mathbf{x_{mis,1-y}}$ is not more informative than $\mathbf{x_{obs}}$ and $\mathbf{y}$, and $\mathbf{y}$ is defined independent of the missing data mechanism, we will assume in the following that $Q_\phi(\cdot|\mathbf{x_{obs}}, \mathbf{m}, \mathbf{x_{mis,1-y}}) = Q_\phi(\cdot|\mathbf{x_{obs}}, \mathbf{m})$. The recognition model can be seen in Figure 1c. Please refer to Table 1 for the ELBO that results from integrating out $\mathbf{y}$. This result is proved in the supplementary material.

Since we parameterize the recognition model $Q_\phi(\mathbf{x_{mis,y}}, \mathbf{z}, \mathbf{r}, |\mathbf{x_{obs}}, \mathbf{m})$ as a neural network, the input has to be of a fixed size. We thus implement an input-dropout layer (Nazabal et al., 2020) which takes all data (missing or observed) as input and drops out all the missing observations. The mechanics of this approach are the same as mean imputing the standardized input before applying the VAE model.




(a) Generative model with $P_\theta(\mathbf{m}|\mathbf{x}, \mathbf{r})$

(b) Generative model with $P_\theta(\mathbf{m}|\mathbf{z}, \mathbf{r})$

(c) Recognition model.

Figure 1: The architecture of the proposed models. Dashed lines and nodes exist with probability $\pi_j = 1$ if $m_j = 0$ and probability $\pi'$ if $m_j = 1$

### 5.4 Imputation

The VAE model defined above allows us to learn the predictive distribution of $P_\theta(\mathbf{x_{mis}}|\mathbf{x_{obs}}, \mathbf{m})$. When a

Table 1: ELBO of the proposed model.

$$\begin{aligned}\mathcal{L}(\mathbf{x_{obs}}, \mathbf{m}) = \mathbb{E}_{Q_\phi(\mathbf{r},\mathbf{z}|\mathbf{x_{obs}},\mathbf{m})}\Bigg[\log P_\theta(\mathbf{x_{obs}} \mid \mathbf{r}, \mathbf{z}) + \log \frac{P_\theta(\mathbf{z} \mid \mathbf{r})}{Q_\phi(\mathbf{z} \mid \mathbf{r}, \mathbf{x_{obs}}, \mathbf{m})} + \log \frac{P_\theta(\mathbf{r})}{Q_\phi(\mathbf{r} \mid \mathbf{x_{obs}}, \mathbf{m})}\Bigg] \\ + \pi \cdot \mathbb{E}_{Q_\phi(\mathbf{r},\mathbf{z},\mathbf{x_{mis}}|\mathbf{x_{obs}},\mathbf{m})}\Bigg[\log P_\theta(\mathbf{m} \mid \mathbf{r}, \mathbf{z}, \mathbf{x_{obs}}, \mathbf{x_{mis}}) + \log \frac{P_\theta(\mathbf{x_{mis}} \mid \mathbf{r}, \mathbf{z})}{Q_\phi(\mathbf{x_{mis}} \mid \mathbf{x_{obs}}, \mathbf{m}, \mathbf{z}, \mathbf{r})}\Bigg] \\ + (1-\pi) \cdot \mathbb{E}_{Q_\phi(\mathbf{r},\mathbf{z}|\mathbf{x_{obs}},\mathbf{m},\mathbf{x_{mis}})}\Bigg[\log P_\theta(\mathbf{m} \mid \mathbf{r}, \mathbf{z}, \mathbf{x_{obs}}, \mathbf{x_{mis}}) + \log P_\theta(\mathbf{x_{mis}} \mid \mathbf{r}, \mathbf{z})\Bigg]\end{aligned}$$

single value is used for imputing the missing data, we speak of single imputation. For continuous data and when the $l_2$ norm is a relevant error metric, we follow Burda et al. (2015) and Mattei and Frellsen (2019) and impute the missing data by estimating $\mathbb{E}(\mathbf{x_{mis}}|\mathbf{x_{obs}}, \mathbf{m})$ using importance sampling. This procedure will reduce the noise in our estimation compared to sampling a single value from $P_\theta(\mathbf{x_{mis}}|\mathbf{z}, \mathbf{r})$. The above expectation can also be written as

$$\begin{aligned}&\mathbb{E}(h(\mathbf{x_{mis}})|\mathbf{x_{obs}}, \mathbf{m}) \\ &= \int h(\mathbf{x_{mis}}) P_\theta(\mathbf{x_{mis}}|\mathbf{x_{obs}}, \mathbf{m}) d\mathbf{x_{mis}} \\ &= \int h(\mathbf{x_{mis}}) P_\theta(\mathbf{x_{mis}}|\mathbf{z}) P_\theta(\mathbf{z}|\mathbf{x_{obs}}, \mathbf{m})\, d\mathbf{z}\; d\mathbf{x_{mis}},\end{aligned}$$

where $h$ is an absolutely integrable function of $\mathbf{x_{mis}}$ and equal to the identity function in the case of single imputation. We use a self-normalized importance weighted estimator with the importance distribution $P_\theta(\mathbf{x_{mis}}|\mathbf{z})Q_\phi(\mathbf{z}|\mathbf{x_{obs}}, \mathbf{m})$ and the resulting estimator

$$\mathbb{E}(h(\mathbf{x_{mis}})|\mathbf{x_{obs}}, \mathbf{m}) \approx \frac{1}{\sum_{l=1}^{L} w^{(l)}} \sum_{l=1}^{L} w^{(l)} h(\mathbf{x_{mis}^{(l)}}),$$

where $(\mathbf{x_{mis}}^{(l)}, \mathbf{z}^{(l)})_{l=1}^{L}$ are samples from $P_\theta(\mathbf{x_{mis}}|\mathbf{z})Q_\phi(\mathbf{z}|\mathbf{x_{obs}})$ obtained by ancestral sampling. More specifically, we get samples from $Q_\phi(\mathbf{z}|\mathbf{x_{obs}})$ by marginalizing out $\mathbf{r}$ in $Q_\phi(\mathbf{z}|\mathbf{r}, \mathbf{x_{obs}}, \mathbf{m})$ and $Q_\phi(\mathbf{x_{mis}}|\mathbf{z}, \mathbf{r}, \mathbf{x_{obs}}, \mathbf{m})$. The importance weights are then defined by

$$w^{(l)} = \sum_{\mathbf{r}=1}^{k} \frac{P_\theta(\mathbf{x_{mis}}^{(l)}|\mathbf{z}^{(l)}, \mathbf{r}) P_\theta(\mathbf{z}^{(l)}|\mathbf{r}) P_\theta(\mathbf{r})}{Q_\phi(\mathbf{z}^{(l)}|\mathbf{r}, \mathbf{x_{obs}}) Q_\phi(\mathbf{r}|\mathbf{x_{obs}}, \mathbf{m})}.$$

Note that, in contrast to Mattei and Frellsen (2019), we do not rely on such an IWAE-based approach in the training stage. This stems from the fact that we propose a complex variational distribution and thus do not need the additional complexity induced by an importance-weighted approach (Cremer et al., 2017; Mattei and Frellsen, 2019). This hypothesis is further supported by our results in the Experiments section.

While single imputation returns a single estimator for the missing data, it is usually more interesting to sample multiple values from the predictive distribution for uncertainty quantification and statistically robust inference in downstream tasks (Rubin, 1996). Multiple imputations can be obtained by sampling multiple times from the generative model. Again we follow Mattei and Frellsen (2019) and use sampling importance resampling. For this, we generate a set of imputations and weight them using the weights defined above. The multiple imputations are then sampled from this weighted set.

## 6 EXPERIMENTS

In this section, we quantitatively evaluate the imputation performance of our model (from now on PSMVAE) and several state-of-the-art imputation approaches on UCI data sets (Lichman et al., 2013) commonly used in the machine learning for imputation literature (Yoon et al., 2018; Nazabal et al., 2020). Each experiment is repeated five times with different seeds. We report the root mean squared error (RMSE) of the estimators of the missing data along with its standard deviations across the experiments. Unless otherwise specified, we choose $\pi' = 0.5$. We further use the same hyperparameter constellation for each data set. Please see the supplementary material for details, the complete results and additional experiments such as the multiple imputation setting. The complete code for all experiments can be found online and figures are created using Weights & Biases (Biewald, 2020).

We synthetically introduce missingness into our data sets. The MNAR data are generated by self-masking one of the features. Every time this randomly selected variable is larger than its median we set it to zero with probability equal to the missingness rate, similar to Ipsen et al. (2021). The MCAR characteristic is introduced by randomly setting a feature to be missing with probability equal to the missingness rate. We present an overview of some results in Table 2 and Table 4. Note that the best score of each group is highlighted in bold, but that some of the approximate

Table 2: RMSE (Average±Std of RMSE) for single imputation on data with a missingness rate of 20%.

| Model class | Adult | | Letter | | Wine | |
|---|---|---|---|---|---|---|
| Missingness | MCAR | MNAR | MCAR | MNAR | MCAR | MNAR |
| PSMVAE(a) | .2494±.0021 | .4943±.3187 | .0964±.0013 | **.0835±.0153** | .0958±.0054 | **.1034±.0026** |
| PSMVAE(b) | .2426±.0019 | .5249±.2387 | .0936±.0008 | .0864±.0152 | .0890±.0029 | .1158±.0095 |
| ↳ K=10.000 | **.2306±.0019** | .4981±.2176 | **.0879±.0006** | .0854±.0162 | **.0832±.0022** | .1069±.0100 |
| ↳ w/o M | .2450±0.031 | **.4815±.2778** | .0941±.0009 | .0908±.0185 | .0885±.0024 | .1167±.0096 |
| DLGM | .2467±.0033 | .5468±.2328 | .0947±.0172 | .0947±.0172 | .0923±.0036 | .1234±.0104 |
| HIVAE | .2693±.0338 | .4907±.2072 | .1023±.0008 | .0947±.0179 | .0940±.0032 | .1246±.0156 |
| VAE | .2562±.0027 | .5012±.2313 | .1119±.0008 | .1061±.0194 | .1067±.0035 | .1255±.0105 |
| MIWAE | .2845±.0121 | .6081±.2423 | .1183±.0018 | **.1024±.0191** | .1129±.0034 | .1253±.0259 |
| ↳ K=10.000 | **.2373±.0015** | .5872±.3065 | **.1149±.0004** | .1242±.0063 | **.0915±.0017** | .0803±.0117 |
| not-MIWAE | .2374±.0011 | **.5201±.2640** | .1153±.0005 | .1192±.0317 | .0928±.0022 | **.0756±.0089** |
| GAIN | .2570±.0084 | .5940±.3744 | .1518±.0074 | .1316±.0061 | .1749±.0042 | .1151±.0175 |
| MICE | .2383±.0013 | .5879±.3079 | .1167±.0008 | .1235±.0069 | .0881±.0030 | .0782±.0117 |
| ↳ sample | .3166±.0015 | .6100±.2375 | .1575±.0007 | .1664±.0128 | .1264±.0025 | .1073±.0135 |
| MissForest | **.2246±.0026** | **.4513±.1774** | **.0650±.0018** | **.0534±.0113** | **.0738±.0023** | **.0698±.0035** |
| mean | .2510±.0012 | .6676±.3350 | .1560±.0005 | .1938±.0341 | .1765±.0041 | .1591±.0292 |

confidence intervals overlap.

### 6.1 Properties Of The Model

First, we analyze how the performance of our model changes due to incremental modifications in its architecture. For this purpose we compare a basic VAE, a simplified HIVAE model (HIVAE without different losses for different categories), and a deep latent Gaussian Variable model (DLGM, which is a VAE with one categorical and two Gaussian latent variables) with our model. We train our algorithm once with one importance sample and no missingness mask as input and output (PSMVAE w/o M), once with one importance sample and generative model as specified in Figure 1b (PSMVAE(b)), once as PSMVAE(b) with 10,000 importance samples, and finally with generative model as specified in Figure 1a (PSMVAE(a)) and 10,000 importance samples.

We note in our experiments that adding a categorical variable to a basic VAE always leads to a reduction in the RMSE loss of up to 1.4 percentage points. Moreover, we find that the PSMVAE w/o M is performing better than the DLGM. The main difference between the models is the assumed probabilistic semi-supervision of $\mathbf{x_{mis}}$ in the PSMVAE(b) while $\pi'$ is equal to 0 in the DLGM.[3] The average improvement of approximately 4.78% compared to the DLGM loss across all data sets speaks for the semi-supervised approach. Our model is also consistently performing better than the HIVAE which is a special case of our model when the union of $\mathbf{x_{mis}}$ and $\mathbf{x_{obs}}$ are assumed to be $d$-dimensional and $\pi' = 0$. In four of the displayed twelve experiments, from which three were run on data with MNAR, including the missingness mask led to a higher loss. We hypothesize that this stems from the class imbalance: While only one of the variables is set to be MNAR, the other variables are always observed.

In Table 3, we see that our models learn to cluster the data into meaningful missingness pattern-sets. For this purpose we ran our models and a HIVAE, whose input and output was a concatenation of the observed data and the corresponding missingness masks, on a variety of data sets. We did not only model MNAR, but also created a version of the data sets where all variables were MCAR and one variable was additionally MNAR. We assume that the model learns to cluster the data if the categories of the (here two-dimensional) latent variable $\mathbf{r}$ correspond to the cases when the variable that is MNAR is higher resp. lower than its median, which is the threshold for the assignment to the different missingness mechanisms. The pattern-set accuracy was then determined by computing the accuracy of a permutation $\widetilde{p}$ of the learned categories $\mathbf{r}|\mathbf{x_{obs}}, \mathbf{m}$ where the permutation was chosen such that $\widetilde{p}(\mathbf{r}|\mathbf{x_{obs}}, \mathbf{m}) = 0$ contained the highest fraction of observations of the MNAR pattern-set. Even though we only induce 20% missingness we see that all models learn to cluster the data.

### 6.2 Comparison With Other Imputation Approaches

We compare our proposed method with common imputation methods from the deep learning and statistical literature. As the losses of MIWAE and PSM-

[3]Please refer to the supplementary material for a detailed description of the DLGM.

Table 3: Pattern-set accuracy on data with a missingness rate of 20%.

| | Breast | | Wine | | Spam | |
|---|---|---|---|---|---|---|
| Algorithm | MNAR | MNAR+MCAR | MNAR | MNAR+MCAR | MNAR | MNAR+MCAR |
| PSMVAE (a) | **.7819±.0937** | **.7116±.1110** | .6113±.0564 | .6435±.0659 | .7349±.0638 | .8137±.0965 |
| PSMVAE (b) | .7709±.0824 | .6975±.1104 | .6068±.0535 | .6476±.0843 | **.7375±.0622** | .8123±.1019 |
| HIVAE w.M | .7050±.1161 | .6887±.1079 | **.6544±.0813** | **.7267±.0725** | .7244±.0833 | **.8145±.0949** |

Table 4: RMSE (Average±Std of RMSE) for single imputation on data with a missingness rate of 80%.

| | Breast | | Credit | | Spam | |
|---|---|---|---|---|---|---|
| Algorithm | MCAR | MNAR | MCAR | MNAR | MCAR | MNAR |
| PSMVAE(a) | **.1003±.0035** | .1114±.0417 | .1715±.0023 | **.0948±.061** | .0603±.0022 | .0801±.0232 |
| PSMVAE(b) | .1004±.0013 | .1085±.0258 | .1712±.0010 | .1095±.0533 | .0591±.0023 | .0889±.0427 |
| ↳ K=10,000 | .1125±.0034 | **.0729±.0184** | **.1701±.0011** | .0973±.0437 | **.0552±.0035** | **.0800±.0377** |
| ↳ w/o M | .1058±.0025 | .1025±.0276 | .1730±.0033 | .1202±.0555 | .0594±.0023 | .0879±.0433 |
| DLGM | .1179±.0011 | .1303±.0363 | .1742±.0026 | .1458±.0507 | .0624±.0027 | .0943±.0415 |
| HIVAE | .1207±.0023 | .1406±.0537 | .1743±.11 | .1301±.0542 | .0621±.0020 | .0924±.0426 |
| VAE | .1214±.0013 | .1214±.0475 | .1748±.0011 | .1267±.0576 | .0614±.0023 | .0929±.0425 |
| MIWAE | .1926±.0256 | .1452±.0316 | .1807±.0009 | .1716±.1262 | .0690±.0025 | .1133±.0709 |
| ↳ K=10,000 | **.0953±.0020** | **.1229±.0381** | **.1582±.0004** | .1076±.0391 | **.0579±.0021** | **.0858±.0364** |
| not-MIWAE | .2795±.0496 | .1393±.0425 | .2744±.0045 | **.1027±.0293** | .1112±.0035 | .0875±.0456 |
| GAIN | .6115±.1014 | .9170±.7307 | .1654±.0024 | .1334±.0649 | .0603±.0020 | .0886±.0423 |
| MICE | .1343±.0044 | .1445±.0424 | .1671±.0036 | **.1113±.0356** | .0619±.0023 | **.0886±.0372** |
| ↳ sample | .1419±.0021 | .1526±.0392 | .1961±.0010 | .1664±.0128 | .0804±.0027 | .1000±.0418 |
| MissForest | **.1155±.0020** | **.1260±.0269** | .1759±.0028 | .1145±.0384 | .0732±.0026 | .0909±.0397 |
| Mean | .1496±.0010 | .1545±.0297 | **.1640±.0005** | .1666±.0976 | **.0600±.0020** | .0926±.046 |

VAE(b) indicate, importance sampling is essential in the imputation step of VAEs to reduce the noise of the imputations. We further note that traditional methods, such as MissForest and MICE, often outperform neural network based methods. We claim that this roots from the fact that single imputations in linear models, such as MICE learned with Bayesian Ridge, rely on consistent estimators of the expectation over the missing data distributions. When we sample a single value from the predictive distribution of MICE instead, we note that the single sample imputation of our method outperforms MICE. In Table 4 we see that the deep generative models perform better than MICE and MissForest on multiple data sets when the missingness rate is high. Another drawback of MICE and MissForest towards deep generative methods is that they are typically not scalable to high-dimensional data sets while our proposed models are.

We further note that our model achieves state-of-the-art imputation performance on both MCAR and MNAR data while MIWAE and Not-MIWAE typically only outperform other methods on one of the missingness types and are thus less robust to the misspecification of the missingness mechanism. In Figure 2 we see that the PSMVAE model with $\pi'(m_j) = \frac{P_\theta(m_j=1)}{1+P_\theta(m_j=1)}$ (blue line) is the only method with a decreasing imputation loss the longer the method learns. This choice ensures that the distributions of $x_{obs,j}$ and $x_{mis,j}$ are closer whenever $x_j$ was more likely to be missing and that $x_{mis,j}$ is always observed with a probability of at least 50%. Choosing a small constant $\pi'$ leads to high volatility in the results. Note that we choose a misspecified missing model (learned by self-masked logistic regression) for training PSMVAE(a) and not-MIWAE. While the loss of both models starts to increase after 500 steps, the loss of PSMVAE(a) increases less steeply than the the loss of not-MIWAE.

# 7 CONCLUSION AND FUTURE WORK

We propose a generative model combining ideas from VAEs and pattern-set mixture models for missing data imputation. This architecture specifies a variational autoencoder that prevents overfitting to data by introducing the assumption of probabilistic semi-supervision of the missing data. We also allow for the specification of a missing model. We see in various experiments with real-world data sets that our model achieves state-oft-the-art imputation performance for different missingness types and explicitly models meaningful clusters which add to the inter-

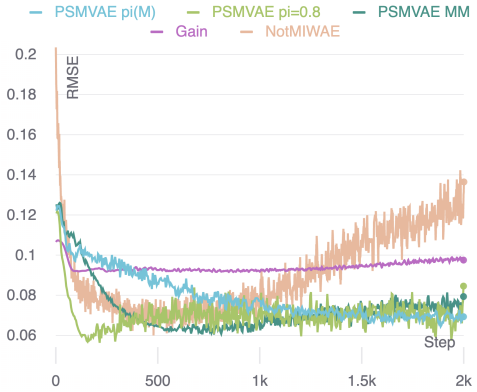



Figure 2: RMSE for single imputation (K=10,000) of various models as a function of iteration steps trained on the Breast data set with 80% induced MNAR missingness.

pretability of the model when no information on the missingness mechanism is available. We recommend our model for use when the underlying data mechanisms are unknown, the missingness is high and/or the data is high-dimensional. Future research will focus on incorporating assumptions on the differences between the missing models of distinct missingness pattern-sets.

### Acknowledgments

We thank the reviewers for their constructive feedback. SG is a student of the EPSRC CDT in Modern Statistics and Statistical Machine Learning (EP/S023151/1) and receives funding from the Oxford Radcliffe Scholarship and Novartis. RC is supported by the Engineering and Physical Sciences Research Council (EPSRC) through the Bayes4Health programme Grant EP/R018561/1. LJK is supported by the French government under management of Agence Nationale de la Recherche as part of the "ABSint" programme, reference ANR-18-CE40-0034. CH is supported by The Alan Turing Institute, Health Data Research UK, the Medical Research Council UK, the EPSRC through the Bayes4Health programme Grant EP/R018561/1, and AI for Science and Government UK Research and Innovation (UKRI).

### References

Bertalmio, M., Sapiro, G., Caselles, V., and Ballester, C. (2000). Image inpainting. In *SIGGRAPH '00: Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, pages 417–424.

Biewald, L. (2020). Experiment tracking with weights and biases. Software available from wandb.com.

Burda, Y., Grosse, R., and Salakhutdinov, R. (2015). Importance weighted autoencoders. *arXiv preprint arXiv:1509.00519*.

Buuren, S. v. and Groothuis-Oudshoorn, K. (2010). MICE: Multivariate imputation by chained equations in R. *Journal of Statistical Software*, pages 1–68.

Collier, M., Nazabal, A., and Williams, C. K. (2020). VAEs in the presence of missing data. *arXiv preprint arXiv:2006.05301*.

Cremer, C., Morris, Q., and Duvenaud, D. (2017). Reinterpreting importance-weighted autoencoders. *arXiv preprint arXiv:1704.02916*.

Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)*, 39(1):1–22.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial nets. In *Advances in Neural Information Processing Systems*, pages 2672–2680.

Harel, O. and Schafer, J. L. (2009). Partial and latent ignorability in missing-data problems. *Biometrika*, 96(1):37–50.

Ipsen, N. B., Mattei, P.-A., and Frellsen, J. (2021). not-MIWAE: Deep Generative Modelling with Missing not at Random Data. In *International Conference on Learning Representations*.

Joy, T., Schmon, S. M., Torr, P. H., Siddharth, N., and Rainforth, T. (2020). Rethinking Semi-Supervised Learning in VAEs. *arXiv preprint arXiv:2006.10102*.

Khemakhem, I., Kingma, D., Monti, R., and Hyvarinen, A. (2020). Variational Autoencoders and Nonlinear ICA: A Unifying Framework. In *International Conference on Artificial Intelligence and Statistics*, pages 2207–2217.

Kingma, D. P., Mohamed, S., Rezende, D. J., and Welling, M. (2014). Semi-supervised Learning with Deep Generative Models. In *Advances in Neural Information Processing Systems*, pages 3581–3589.

Kingma, D. P. and Welling, M. (2013). Auto-Encoding Variational Bayes. *arXiv preprint arXiv:1312.6114*.

Li, S. C.-X., Jiang, B., and Marlin, B. (2019). Learning from Incomplete Data with Generative Adversarial Networks. In *International Conference on Learning Representations*.

Lichman, M. et al. (2013). UCI machine learning repository.

Little, R. J. (1993). Pattern-mixture models for multivariate incomplete data. *Journal of the American Statistical Association*, 88(421):125–134.

Little, R. J. and Rubin, D. B. (2019). *Statistical analysis with missing data*, volume 793. John Wiley & Sons.

Mattei, P.-A. and Frellsen, J. (2019). MIWAE: deep generative modelling and imputation of incomplete data sets. In *International Conference on Machine Learning*, pages 4413–4423.

Mazumder, R., Hastie, T., and Tibshirani, R. (2010). Spectral regularization algorithms for learning large incomplete matrices. *The Journal of Machine Learning Research*, 11:2287–2322.

Molenberghs, G., Beunckens, C., Sotto, C., and Kenward, M. G. (2008). Every missingness not at random model has a missingness at random counterpart with equal fit. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 70(2):371–388.

Nazabal, A., Olmos, P. M., Ghahramani, Z., and Valera, I. (2020). Handling Incomplete Heterogeneous Data using VAEs. *Pattern Recognition*, page 107501.

Rezende, D. J., Mohamed, S., and Wierstra, D. (2014). Stochastic Backpropagation and Approximate Inference in Deep Generative Models. In *International Conference on Machine Learning*.

Rubin, D. B. (1976). Inference and missing data. *Biometrika*, 63(3):581–592.

Rubin, D. B. (1996). Multiple imputation after 18+ years. *Journal of the American Statistical Association*, 91(434):473–489.

Sportisse, A., Boyer, C., and Josse, J. (2020). Estimation and imputation in probabilistic principal component analysis with missing not at random data. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M. F., and Lin, H., editors, *Advances in Neural Information Processing Systems*, volume 33, pages 7067–7077. Curran Associates, Inc.

Stekhoven, D. J. and Bühlmann, P. (2012). MissForest—non-parametric missing value imputation for mixed-type data. *Bioinformatics*, 28(1):112–118.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., and Wojna, Z. (2016). Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826.

Wang, X., Zhang, R., Sun, Y., and Qi, J. (2019). Doubly robust joint learning for recommendation on data missing not at random. In *International Conference on Machine Learning*, pages 6638–6647.

Wu, M. C. and Carroll, R. J. (1988). Estimation and comparison of changes in the presence of informative right censoring by modeling the censoring process. *Biometrics*, pages 175–188.

Yoon, J., Jordon, J., and Van Der Schaar, M. (2018). GAIN: Missing Data Imputation using Generative Adversarial Nets. In Dy, J. and Krause, A., editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80, pages 5689–5698.

# Deep Generative Pattern-Set Mixture Models for Nonignorable Missingness: Supplementary Materials

**Sahra Ghalebikesabi**
University of Oxford

**Rob Cornish**
University of Oxford

**Luke J. Kelly**
CEREMADE, CNRS
Université Paris-Dauphine

**Chris Holmes**
University of Oxford

## 1 PROOFS

*Proof of Proposition 1.* Let $\mu$ denote the posterior distributional parameter of $(\mathbf{x_{mis}} \mid \mathbf{x_{obs}}, \mathcal{M}) \sim \mathcal{N}(\mu, c)$ for some fixed constant $c$. In the following, we assume that the distributional parameter is found by optimization of the expected likelihood given the augmented dataset

$$\mathbf{E}_{\mathbf{y}}[P_\theta(\mathbf{x_{obs}}, \mathbf{x_{mis,1-y}}, \mathbf{m} \mid \mathcal{D}_{\boldsymbol{\pi}}(\mathbf{y}))] = \boldsymbol{\pi} P_\theta(\mathbf{x_{obs}}, \mathbf{m} \mid \mathcal{D}_{\boldsymbol{\pi}}(\mathbf{1})) + (1-\boldsymbol{\pi}) P_\theta(\mathbf{x_{obs}}, \mathbf{m}, \mathbf{x_{mis}} \mid \mathcal{D}_{\boldsymbol{\pi}}(\mathbf{0})). \tag{1}$$

Let $\theta$ denote the parameter set of the generative model. We write $\theta_1$ for the parameter set where $\mu = \mu_1$ and $\theta_2$ for the parameter set where $\mu = \mu_2$. For any two parameters $\mu_1$ and $\mu_2$ with

$$\begin{aligned}\boldsymbol{\pi} P_{\theta_1}(\mathbf{x_{obs}}, \mathbf{m} \mid \mathcal{D}_{\boldsymbol{\pi}}(\mathbf{1})) + (1-\boldsymbol{\pi}) P_{\theta_1}(\mathbf{x_{obs}}, \mathbf{m}, \mathbf{x_{mis}} \mid \mathcal{D}_{\boldsymbol{\pi}}(\mathbf{0})) =& \boldsymbol{\pi} P_{\theta_2}(\mathbf{x_{obs}}, \mathbf{m} \mid \mathcal{D}_{\boldsymbol{\pi}}(\mathbf{1})) \\ &+ (1-\boldsymbol{\pi}) P_{\theta_2}(\mathbf{x_{obs}}, \mathbf{m}, \mathbf{x_{mis}} \mid \mathcal{D}_{\boldsymbol{\pi}}(\mathbf{0})),\end{aligned}$$

we have to show that $\mu_1 = \mu_2$.

From the identifiability of the mean parameter of Gaussian distributions as sample mean in maximum likelihood estimation, it follows that

$$\begin{aligned}\boldsymbol{\pi} P_{\theta_1}(\mathbf{x_{obs}}, \mathbf{m} \mid \mathcal{D}_{\boldsymbol{\pi}}(\mathbf{1})) &= \boldsymbol{\pi} P_{\theta_2}(\mathbf{x_{obs}}, \mathbf{m} \mid \mathcal{D}_{\boldsymbol{\pi}}(\mathbf{1})) \text{ and thus} \\ (1-\boldsymbol{\pi}) P_{\theta_1}(\mathbf{x_{obs}}, \mathbf{m}, \mathbf{x_{mis}} \mid \mathcal{D}_{\boldsymbol{\pi}}(\mathbf{0})) &= (1-\boldsymbol{\pi}) P_{\theta_2}(\mathbf{x_{obs}}, \mathbf{m}, \mathbf{x_{mis}} \mid \mathcal{D}_{\boldsymbol{\pi}}(\mathbf{0})),\end{aligned} \tag{2}$$

where only the probabilities in Equation (2) depend on $\mu$. The maximization of $P_{\theta_2}(\mathbf{x_{obs}}, \mathbf{m}, \mathbf{x_{mis}} \mid \mathcal{D}_{\boldsymbol{\pi}}(\mathbf{0}))$ equals a maximum likelihood estimation with a fully observed dataset of $\mathbf{x_{mis}}$. Again, from the identifiability of the mean parameter of Gaussian distributions, it follows that $\mu_1 = \mu_2$ from (2). □

*Proof of ELBO in Table 1.* The expected likelihood given the augmented dataset (1) is a weighted sum over the likelihood of the observed model $P_\theta(\mathbf{x_{obs}}, \mathbf{m}, \mathbf{x_{mis}} \mid \mathcal{D}_{\boldsymbol{\pi}}(\mathbf{0}))$ and the likelihood of the unobserved model $P_\theta(\mathbf{x_{obs}}, \mathbf{m} \mid \mathcal{D}_{\boldsymbol{\pi}}(\mathbf{1}))$.

The observed model can be learned by maximizing the ELBO which we can compute using Jensen's inequality:

$$\begin{aligned}\log P_\theta(\mathbf{x_{obs}}, \mathbf{m}, \mathbf{x_{mis}} \mid \mathcal{D}_{\boldsymbol{\pi}}(\mathbf{0})) &= \log P_\theta(\mathbf{x_{obs}}, \mathbf{m}, \mathbf{x_{mis}}, \mathbf{r}, \mathbf{z} \mid \mathcal{D}_{\boldsymbol{\pi}}(\mathbf{0})) - \log P_\theta(\mathbf{r}, \mathbf{z} \mid \mathcal{D}_{\boldsymbol{\pi}}(\mathbf{0})) \\ &\geq \mathbb{E}_{Q_\phi(\mathbf{r},\mathbf{z} \mid \mathbf{x_{obs}}, \mathbf{m}, \mathbf{x_{mis}})}\Big[\log P_\theta(\mathbf{x_{obs}}, \mathbf{m}, \mathbf{x_{mis}}, \mathbf{r}, \mathbf{z} \mid \mathbf{x_{obs}}, \mathbf{m}, \mathbf{x_{mis}}) - \log Q_\phi(\mathbf{r}, \mathbf{z} \mid \mathbf{x_{obs}}, \mathbf{m}, \mathbf{x_{mis}})\Big] \\ &= \mathbb{E}_{Q_\phi(\mathbf{r},\mathbf{z} \mid \mathbf{x_{obs}}, \mathbf{m}, \mathbf{x_{mis}})}\Big[\log P_\theta(\mathbf{m} \mid \mathbf{r}, \mathbf{z}, \mathbf{x_{obs}}, \mathbf{x_{mis}}) + \log P_\theta(\mathbf{x_{obs}} \mid \mathbf{r}, \mathbf{z}) + \log P_\theta(\mathbf{x_{mis}} \mid \mathbf{r}, \mathbf{z}) \\ &\qquad + \log \frac{P_\theta(\mathbf{z} \mid \mathbf{r})}{Q_\phi(\mathbf{z} \mid \mathbf{r}, \mathbf{x_{obs}}, \mathbf{m}, \mathbf{x_{mis}})} + \log \frac{P_\theta(\mathbf{r})}{Q_\phi(\mathbf{r} \mid \mathbf{x_{obs}}, \mathbf{m}, \mathbf{x_{mis}})}\Big]\end{aligned}$$

[1] denotes senior author

$$= \mathbb{E}_{Q_\phi(\mathbf{r},\mathbf{z}|\mathbf{x_{obs}},\mathbf{m})}\Bigg[\log P_\theta(\mathbf{m} \mid \mathbf{r},\mathbf{z},\mathbf{x_{obs}},\mathbf{x_{mis}}) + \log P_\theta(\mathbf{x_{obs}} \mid \mathbf{r},\mathbf{z}) + \log P_\theta(\mathbf{x_{mis}} \mid \mathbf{r},\mathbf{z}) + \log \frac{P_\theta(\mathbf{z} \mid \mathbf{r})}{Q_\phi(\mathbf{z} \mid \mathbf{r},\mathbf{x_{obs}},\mathbf{m})} + \log \frac{P_\theta(\mathbf{r})}{Q_\phi(\mathbf{r} \mid \mathbf{x_{obs}},\mathbf{m})}\Bigg],$$

where $\log P_\theta(\mathbf{m} \mid \mathbf{r},\mathbf{z},\mathbf{x_{obs}},\mathbf{x_{mis}}) = \log P_\theta(\mathbf{m} \mid \mathbf{r},\mathbf{x_{obs}},\mathbf{x_{mis}})$ for PSMVAE(a) and $\log P_\theta(\mathbf{m} \mid \mathbf{r},\mathbf{z},\mathbf{x_{obs}},\mathbf{x_{mis}}) = \log P_\theta(\mathbf{m} \mid \mathbf{r},\mathbf{z})$ for PSMVAE(b). The last equation follows from the assumption that $Q_\phi(\cdot|\mathbf{x_{obs}},\mathbf{m},\mathbf{x_{mis}}) = Q_\phi(\cdot|\mathbf{x_{obs}},\mathbf{m})$.

Similarly we obtain the lower bound of the unobserved model:

$$\begin{aligned}
\log P_\theta(\mathbf{x_{obs}},\mathbf{m} \mid \mathcal{D}_{\boldsymbol{\pi}}(\mathbf{1})) &= \log P_\theta(\mathbf{x_{obs}},\mathbf{m},\mathbf{x_{mis}},\mathbf{r},\mathbf{z} \mid \mathcal{D}_{\boldsymbol{\pi}}(\mathbf{1})) - \log P_\theta(\mathbf{r},\mathbf{z},\mathbf{x_{mis}} \mid \mathcal{D}_{\boldsymbol{\pi}}(\mathbf{1})) \\
&\geq \mathbb{E}_{Q_\phi(\mathbf{r},\mathbf{z},\mathbf{x_{mis}}|\mathbf{x_{obs}},\mathbf{m})}\Big[\log P_\theta(\mathbf{x_{obs}},\mathbf{m},\mathbf{x_{mis}},\mathbf{r},\mathbf{z} \mid \mathbf{x_{obs}},\mathbf{m}) - \log Q_\phi(\mathbf{r},\mathbf{z},\mathbf{x_{mis}} \mid \mathbf{x_{obs}},\mathbf{m})\Big] \\
&= \mathbb{E}_{Q_\phi(\mathbf{r},\mathbf{z},\mathbf{x_{mis}}|\mathbf{x_{obs}},\mathbf{m})}\Bigg[\log P_\theta(\mathbf{m} \mid \mathbf{r},\mathbf{z},\mathbf{x_{obs}},\mathbf{x_{mis}}) + \log P_\theta(\mathbf{x_{obs}} \mid \mathbf{r},\mathbf{z}) + \log \frac{P_\theta(\mathbf{x_{mis}} \mid \mathbf{r},\mathbf{z})}{Q_\phi(\mathbf{x_{mis}} \mid \mathbf{x_{obs}},\mathbf{m},\mathbf{z},\mathbf{r})} \\
&\qquad + \log \frac{P_\theta(\mathbf{z} \mid \mathbf{r})}{Q_\phi(\mathbf{z} \mid \mathbf{r},\mathbf{x_{obs}},\mathbf{m},\mathbf{x_{mis}})} + \log \frac{P_\theta(\mathbf{r})}{Q_\phi(\mathbf{r} \mid \mathbf{x_{obs}},\mathbf{m},\mathbf{x_{mis}})}\Bigg],
\end{aligned}$$

where $\log P_\theta(\mathbf{m} \mid \mathbf{r},\mathbf{z},\mathbf{x_{obs}},\mathbf{x_{mis}})$ is defined as before.

The weighted sum of the ELBOs (denoted by $\mathcal{L}(\mathbf{x_{obs}},\mathbf{m})$ in the following) can then be written as

$$\begin{aligned}
\mathcal{L}(\mathbf{x_{obs}},\mathbf{m}) = {} & \mathbb{E}_{Q_\phi(\mathbf{r},\mathbf{z}|\mathbf{x_{obs}},\mathbf{m})}\Bigg[\log P_\theta(\mathbf{x_{obs}} \mid \mathbf{r},\mathbf{z}) + \log \frac{P_\theta(\mathbf{z} \mid \mathbf{r})}{Q_\phi(\mathbf{z} \mid \mathbf{r},\mathbf{x_{obs}},\mathbf{m})} + \log \frac{P_\theta(\mathbf{r})}{Q_\phi(\mathbf{r} \mid \mathbf{x_{obs}},\mathbf{m})}\Bigg] \\
& + \pi\mathbb{E}_{Q_\phi(\mathbf{r},\mathbf{z},\mathbf{x_{mis}}|\mathbf{x_{obs}},\mathbf{m})}\Bigg[\log P_\theta(\mathbf{m} \mid \mathbf{r},\mathbf{z},\mathbf{x_{obs}},\mathbf{x_{mis}}) + \log \frac{P_\theta(\mathbf{x_{mis}} \mid \mathbf{r},\mathbf{z})}{Q_\phi(\mathbf{x_{mis}} \mid \mathbf{x_{obs}},\mathbf{m},\mathbf{z},\mathbf{r})}\Bigg] \\
& + (1-\pi)\mathbb{E}_{Q_\phi(\mathbf{r},\mathbf{z}|\mathbf{x_{obs}},\mathbf{m},\mathbf{x_{mis}})}\Big[\log P_\theta(\mathbf{m} \mid \mathbf{r},\mathbf{z},\mathbf{x_{obs}},\mathbf{x_{mis}}) + \log P_\theta(\mathbf{x_{mis}} \mid \mathbf{r},\mathbf{z})\Big].
\end{aligned}$$

Specifying $\pi = 1$ when $m_j = 0$ yields the ELBO, as shown in Table 1 of the main paper. □

# 2 DETAILS OF DATASETS AND EXPERIMENTS

## 2.1 Data

We split the datasets into a train, a validation and a test dataset respectively based on a 8:1:1 data split. The data is then normalized and mean imputed. The datasets can be found online in the GitHub repository. We compute the normalized RMSE which corresponds to the RMSE on the normalized data. We predict categorical and nominal variables by rounding the predictions of the models. Please refer to Table 1. Note that our results deviate from those of other papers because we do a train test split for the data which results in a smaller dataset in the training stage.

| Dataset | Sample size | # continuous features | # discrete features |
|---|---|---|---|
| Adult | 32.561 | 3 | 8 |
| Breast | 569 | 30 | 0 |
| Credit | 30.000 | 14 | 9 |
| Letter | 20.000 | 0 | 16 |
| Spam | 4.601 | 57 | 0 |
| Wine | 6.497 | 11 | 1 |

Table 1: Statistics of the datasets

### 2.2 Implementation Details

We use PyTorch to implement the VAE models. The tensorflow code for GAIN can be found online (`https://github.com/jsyoon0823/GAIN`) as well as the PyTorch implementation of MIWAE (`https://github.com/pamattei/miwae`). The code for not-MIWAE was provided by the main author of the corresponding paper, Niels Bruun Ipsen. We use the MissForest implementation of sklearn (IterativeImputer with ExtraTreesRegressor as estimator) and the MICE implementation of sklearn (IterativeImputer with BayesianRidgeRegressor).

We train all benchmark models with default parameters as specified in our code. We run all deep learning methods for 1,000 epochs and train them using the Adam optimizer (Kingma and Ba, 2014). All subgraphs of the VAE-based approaches have one hidden layer with 128 nodes. Unless otherwise stated, we choose the dimension of the latent Gaussian variable $\mathbf{z}$ equal to 20 and the number of categories $k$ of the latent categorical variable $\mathbf{r}$ as 10. Only MIWAE has two hidden layers each with 128 hidden units as specified by Mattei and Frellsen (2019). We use a rectifier as activation function between two layers. We chose a batch size of 200 for the datasets used in the main paper and a batch size of 512 for MNIST. The number of trees used for MissForest is set to 10. We use the self-masking approach of not-MIWAE where $P(\mathbf{m} \mid \mathbf{x})$ is learned by a logistic regression which is independent for each feature.

The benchmark VAE architectures have been modeled as follows:

- *VAE*: a basic VAE with one Gaussian latent variable learned on the observed data (and not on the missingness mask).
- *GMVAE*: a VAE with a Gaussian Mixture prior. That is a VAE with one categorical latent variable $\mathbf{r}$ taking values from $\{1, ..., k\}$ and one conditionally normal distributed latent variable $\mathbf{z} \mid \mathbf{r}$. It learns the generative model $P(\mathbf{x_{obs}}|\mathbf{m}, \mathbf{z}, \mathbf{r}) = P(\mathbf{x_{obs}}, \mathbf{m} \mid \mathbf{z}, \mathbf{r})P(\mathbf{z} \mid \mathbf{r})P(\mathbf{r})$. This model framework is similar to the HIVAE model presented in (Nazabal et al., 2020). Instead of sampling $\mathbf{r}$ from a Gumbel-Softmax distribution (Jang et al., 2017), we instead learn $P(\cdot \mid \mathbf{r} = r)$ for each $r \in \{1, ..., k\}$ and weight the resulting loss for each category with its posterior probability.
- *DLGM*: a deep latent Gaussian model which extends the GMVAE by a second latent Gaussian variable $\mathbf{w}$ with the same dimensionality as $\mathbf{x}$. It learns the generative distribution $P(\mathbf{x_{obs}}, \mathbf{m}, \mathbf{z}, \mathbf{w}, \mathbf{r}) = P(\mathbf{x_{obs}}, \mathbf{m} \mid \mathbf{z}, \mathbf{r})P(\mathbf{w} \mid \mathbf{z}, \mathbf{r})P(\mathbf{z} \mid \mathbf{r})P(\mathbf{r})$. The inference model is structured in the same way as the inference model for the PSMVAE where $\mathbf{x_{mis}}$ is replaced by $\mathbf{w}$.

We then impute $\mathbf{x_{mis}}$ by sampling from the conditional distribution of $\mathbf{x_{obs}}$.

In contrast to Nazabal et al. (2020), we do not use the Gumbel softmax distribution for sampling the categorical variable $\mathbf{r}$ (Jang et al., 2017), but instead integrate out $\mathbf{r}$ in the loss function. See Dilokthanakul et al. (2016) for a similar approach. In the implementation, we weight the log-likelihood of the observed data with the inverse of 1-*missingness rate*. Otherwise increasing the missingness will lead to a higher weight of the log-likelihood of the missingness mask compared to the log-likelihood of the observed data.

## 3 ADDITIONAL EXPERIMENTS

### 3.1 Multiple Imputation

We follow Mattei and Frellsen (2019) and assess the performance of our models in the multiple imputation setting by computing the test accuracy of the predicted target variable when a one layer classification network is trained on the dataset where each observation with missing entries was imputed 20 times. The test set is again made up of 20 multiple imputations.

### 3.2 Preliminary Results On Image Inpainting

We also show that our method can be used for image inpainting using the MNIST dataset (LeCun et al., 2010). We induced missingness completely random as before. We then trained our method using 500 epochs, a batch size of 512, $\pi = 0$ and only one importance sample. Please refer to Figure 1 for the results. Missing pixels were highlighted in red in the first row of each subfigure. The second row shows inpainted images using

| Algorithm | 20% MCAR | 80% MCAR | 20% MNAR | 80% MNAR |
|---|---|---|---|---|
| PSMVAE(a) | .9482 ± .0211 | .8466 ± .0627 | .9489 ± .0116 | .9501 ± .0101 |
| PSMVAE(b) | .9429 ± .0180 | .8791 ± .0511 | .9577 ± .0063 | .9496 ± .0157 |
| MICE | .9545 ± .0141 | .8105 ± .0337 | .9578 ± .0096 | .9578 ± .0096 |

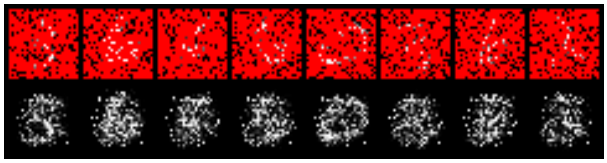
Table 2: Test accuracy of a one layer neural network for the prediction of breast cancer using the breast dataset which was imputed multiple times

the corresponding imputation algorithm. As we see, the PSMVAE(b) yields sharper images than GAIN does. Especially when the missingness is high, the imputations of our model suffer from considerably less noise than those of GAIN.

(a) 20% MCAR inpainted using PSMVAE(b)

(b) 20% MCAR inpainted using GAIN

(c) 80% MCAR inpainted using PSMVAE(b)

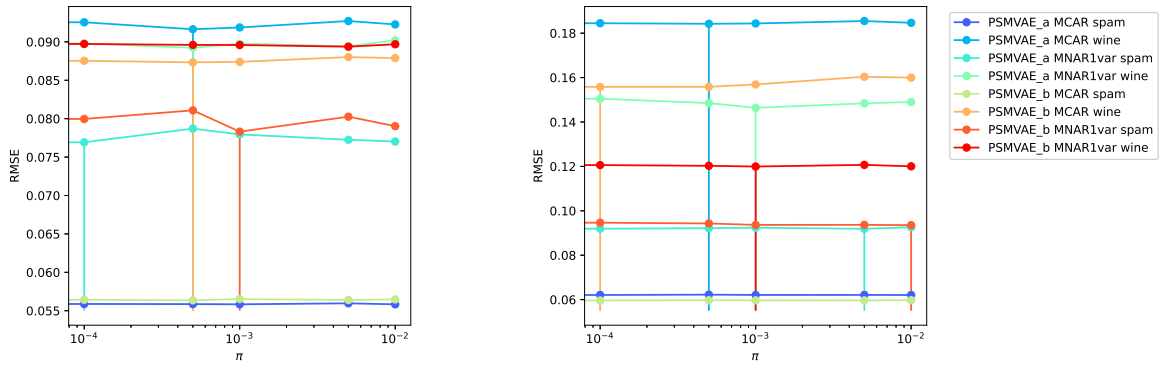


(d) 80% MCAR inpainted using GAIN

Figure 1: MNIST images with pixels MCAR at different rates inpainted with different imputation algorithms

### 3.3 Robustness Study

We now assess the robustness of our method. Please refer to Figure 2 for an illustration of the relationship between the RMSE and the hyperparameter $\pi$. The vertical lines highlight the minimum of the loss curves. We notice that the optimal value of $\pi$ is usually on a similar scale as the optimal value of the weight decay parameter. Only on the MCAR spam dataset it is optimal to have a $\pi = 0$. Please note that the decay of the curves seems infinitesimal because we compare different methods and datasets within the same figure.



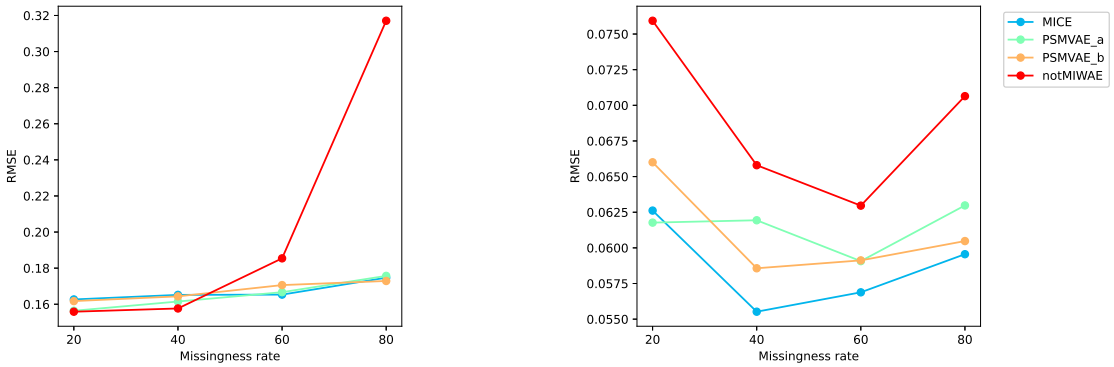

(a) 20% missingness rate

(b) 80% missingness rate

Figure 2: RMSE of imputation for different values of $\pi$ for a missingness rate of 20% (left) and a missingness rate of 80% (right)

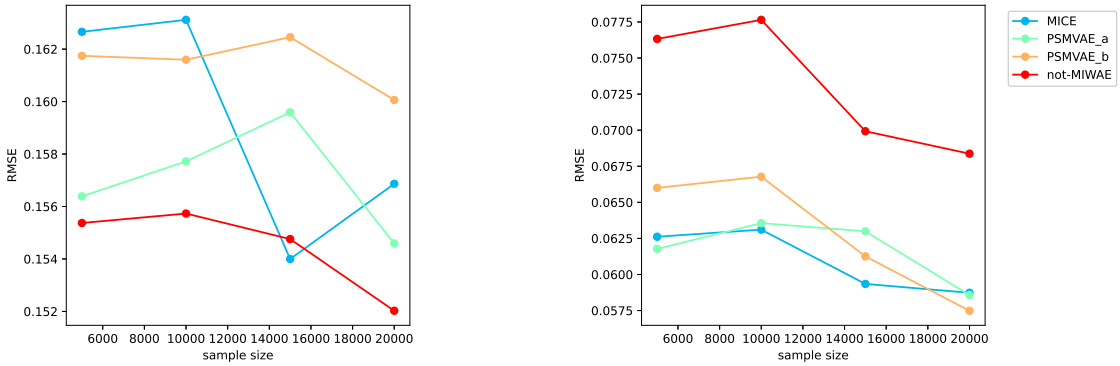
In the following we compare the robustness of our models and two benchmarks on the credit dataset. When assessing the relationship of the RMSE and the missingness rate (see Figure 3), we note that not-MIWAE is

not robust to increasing the missingness rate. This could stem from the fact that increasing the missingness increases the fraction of the optimization loss that comes from the likelihood of the missingness mask compared to the likelihood of the observed data. The larger the missingness, the better do our models compare relative to MICE.



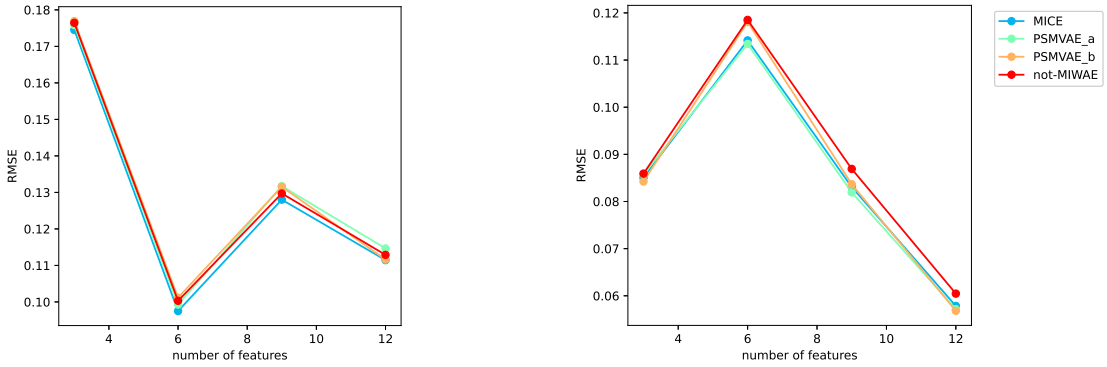

(a) MCAR (b) MNAR

Figure 3: RMSE of imputation for different missingness rates (in %) when data of the credit dataset are missing




(a) MCAR (b) MNAR

Figure 4: RMSE of imputation for different sample sizes when 20% of the data of the credit dataset are missing




(a) MCAR (b) MNAR

Figure 5: RMSE of imputation for different number of features when 20% of the data of the credit dataset are missing

When we change the sample size of the training dataset (see Figure 4), we note that the performance of the deep learning methods improves strictly and that the PSMVAE(b) performs better than MICE for large sample sizes.

Eventually, we compare the robustness of the RMSE when the number of features is changed (Figure 5). We see that the performance of all methods in general decreases when the number of features increases (except for one increase for all methods). The relative performance of our models hereby improves, the more features there are.

## References

Dilokthanakul, N., Mediano, P. A., Garnelo, M., Lee, M. C., Salimbeni, H., Arulkumaran, K., and Shanahan, M. (2016). Deep Unsupervised Clustering with Gaussian Mixture Variational Autoencoders. *arXiv preprint arXiv:1611.02648*.

Jang, E., Gu, S., and Poole, B. (2017). Categorical Reparameterization with Gumbel-Softmax. In *International Conference on Learning Representations*.

Kingma, D. P. and Ba, J. (2014). Adam: A Method for Stochastic Optimization. *arXiv preprint arXiv:1412.6980*.

LeCun, Y., Cortes, C., and Burges, C. (2010). MNIST handwritten digit database.

Mattei, P.-A. and Frellsen, J. (2019). MIWAE: deep generative modelling and imputation of incomplete data sets. In *International Conference on Machine Learning*, pages 4413–4423.

Nazabal, A., Olmos, P. M., Ghahramani, Z., and Valera, I. (2020). Handling Incomplete Heterogeneous Data using VAEs. *Pattern Recognition*, page 107501.

## Statement of Authorship for joint/multi-authored papers for PGR thesis

To appear at the end of each thesis chapter submitted as an article/paper

The statement shall describe the candidate's and co-authors' independent research contributions in the thesis publications. For each publication there should exist a complete statement that is to be filled out and signed by the candidate and supervisor **(only required where there isn't already a statement of contribution within the paper itself).**

| | |
|---|---|
| Title of Paper | Deep generative missingness pattern-set mixture models |
| Publication Status | xPublished □ Accepted for Publication<br>□Submitted for Publication □Unpublished and unsubmitted work written in a manuscript style |
| Publication Details | Ghalebikesabi, S., Cornish, R., Holmes, C., & Kelly, L. (2021, March). Deep generative missingness pattern-set mixture models. In International Conference on Artificial Intelligence and Statistics (pp. 3727-3735). PMLR. |

### Student Confirmation

| | | | |
|---|---|---|---|
| Student Name: | Sahra Ghalebikesabi | | |
| Contribution to the Paper | I conceived methodology, implementation and experiments. During the meetings my collaborators helped out with helpful suggestions on methodology and feedback. They also contributed by checking the results, and proof-reading. | | |
| Signature |  | Date | 28.10.2023 |

### Supervisor Confirmation

By signing the Statement of Authorship, you are certifying that the candidate made a substantial contribution to the publication, and that the description described above is accurate.

| | | |
|---|---|---|
| Supervisor name and title: Chris Holmes | | |
| Supervisor comments<br><br>I am in agreement with the description of the contributions. | | |
| Signature | Date | 28.10.2023 |

This completed form should be included in the thesis, at the end of the relevant chapter.

# 5

# Mitigating Statistical Bias within Differentially Private Synthetic Data

# Mitigating Statistical Bias within Differentially Private Synthetic Data

**Sahra Ghalebikesabi**[1] **Harrison Wilde**[2] **Jack Jewson**[3] **Arnaud Doucet**[1]
**Sebastian Vollmer**[5] **Chris Holmes**[1]

[1]University of Oxford
[2]University of Warwick
[3]Universitat Pompeu Fabra
[5]University of Kaiserslautern, German Research Centre for Artificial Intelligence (DFKI)

## Abstract

Increasing interest in privacy-preserving machine learning has led to new and evolved approaches for generating private synthetic data from undisclosed real data. However, mechanisms of privacy preservation can significantly reduce the utility of synthetic data, which in turn impacts downstream tasks such as learning predictive models or inference. We propose several re-weighting strategies using privatised likelihood ratios that not only mitigate statistical bias of downstream estimators but also have general applicability to differentially private generative models. Through large-scale empirical evaluation, we show that private importance weighting provides simple and effective privacy-compliant augmentation for general applications of synthetic data.

## 1 INTRODUCTION

The prevalence of sensitive datasets, such as electronic health records, contributes to a growing concern for violations of an individual's privacy. In recent years, the notion of Differential Privacy (Dwork et al., 2006) has gained popularity as a privacy metric offering statistical guarantees. This framework bounds how much the likelihood of a randomised algorithm can differ under neighbouring real datasets. We say two datasets $\mathcal{D}$ and $\mathcal{D}'$ are neighbouring when they differ by at most one observation. A randomised algorithm $g : \mathcal{M} \to \mathcal{R}$ satisfies $(\epsilon, \delta)$-differential privacy for $\epsilon, \delta \geq 0$ if and only if for all neighbouring datasets $\mathcal{D}, \mathcal{D}'$ and all subsets $S \subseteq \mathcal{R}$, we have

$$\Pr(g(\mathcal{D}) \in S) \leq \delta + e^{\epsilon} \Pr(g(\mathcal{D}') \in S).$$

The parameter $\epsilon$ is referred to as the privacy budget; smaller $\epsilon$ quantities imply more private algorithms.

Injecting noise into sensitive data according to this paradigm allows for datasets to be published in a private manner. With the rise of generative modelling approaches, such as Generative Adversarial Networks (GANs) (Goodfellow et al., 2014), there has been a surge of literature proposing generative models for differentially private (DP) synthetic data generation and release (Jordon et al., 2019; Xie et al., 2018; Zhang et al., 2017). These generative models often fail to capture the true underlying distribution of the real data, possibly due to flawed parametric assumptions and the injection of noise into their training and release mechanisms. The constraints imposed by privacy-preservation can lead to significant differences between nature's true data generating process (DGP) and the induced synthetic data generating process (SDGP) (Wilde et al., 2020). This increases the bias of estimators trained on data from the SDGP which reduces their utility.

Recent literature has proposed techniques to decrease this bias by modifying the training processes of private algorithms. These approaches are specific to a particular synthetic data generating method (Zhang et al., 2018; Frigerio et al., 2019; Neunhoeffer et al., 2020), or are query-based (Hardt and Rothblum, 2010; Liu et al., 2021) and are thus not generally applicable. Hence, we propose several post-processing approaches that aid mitigating the bias induced by the DP synthetic data.

While there has been extensive research into estimating models directly on protected data without leaking privacy, we argue that releasing DP synthetic data is crucial for rigorous statistical analysis. This makes providing a framework to debias inference on this an important direction of future research that goes beyond the applicability of any particular DP estimator. Because of the post-processing theorem (Dwork et al., 2014), any function on the DP synthetic data is itself DP. This allows deployment of standard statistical analysis tooling that may otherwise be unavailable for DP estimation. These include 1) exploratory data analysis, 2) model verification and analysis of model diagnostics, 3) private release of (newly developed) models for which no

*Proceedings of the 38th Conference on Uncertainty in Artificial Intelligence* (UAI 2022), PMLR 180:696–705.

DP analogue has been derived, 4) the computation of confidence intervals of downstream estimators through the non-parametric bootstrap, and 5) the public release of a data set to a research community whose individual requests would otherwise overload the data curator. This endeavour could facilitate the release of data on public platforms like the UCI Machine Learning Repository (Lichman, 2013) or the creation of data competitions, fuelling research growth for specific modelling areas.

This motivates our main contributions, namely the formulation of multiple approaches to generating DP importance weights that correct for synthetic data's issues. In particular:

- The bias estimation of an existing DP importance weight estimation method, and the introduction of an unbiased extension with smaller variance (Section 3.3).
- An adjustment to DP Stochastic Gradient Descent's sampling probability and noise injection to facilitate its use in the training of DP-compliant neural network-based classifiers to estimate importance weights from combinations of real and synthetic data (Section 3.4).
- The use of discriminator outputs of DP GANs as importance weights that do not require any additional privacy budget (Section 3.5).
- An application of importance weighting to correcting for the bias incurred in Bayesian posterior belief updating with synthetic data motivated by the results from (Wilde et al., 2020) and to exhibit our methods' wide applicability in frequentist and Bayesian contexts (Section 3.1).

# 2 BACKGROUND

Before we proceed, we provide some brief background on bias mitigation in non-private synthetic data generation.

## 2.1 DENSITY RATIOS FOR NON-PRIVATE GANS

Since their introduction, GANs have become a popular tool for synthetic data generation in semi-supervised and unsupervised settings. GANs produce realistic synthetic data by trading off the learning of a generator $Ge$ to produce synthetic observations, with that of a classifier $Di$ learning to correctly classify the training and generated data as real or fake. The generator $Ge$ takes samples from the prior $u \sim p_u$ as an input and generates samples $Ge(u) \in X$. The discriminator $Di$ takes an observation $x \in X$ as input and outputs the probability $Di(x)$ of this observation being drawn from the true DGP. The classification network $Di$ distinguishes between samples from the DGP with label $y = 1$ and distribution $p_D$, and data from the SDGP with label $y = 0$ and distribution $p_G$. Following Bayes' rule we can show that the output of $Di(x)$, namely the probabilities $\widehat{p}(y = 1|x)$ and $\widehat{p}(y = 0|x)$, can be used for importance weight estimation:

$$\frac{\widehat{p}_D(x)}{\widehat{p}_G(x)} = \frac{\widehat{p}(x|y = 1)}{\widehat{p}(x|y = 0)} = \frac{\widehat{p}(y = 1|x)}{\widehat{p}(y = 0|x)} \frac{\widehat{p}(y = 0)}{\widehat{p}(y = 1)}. \quad (1)$$

This observation has been exploited in a stream of literature focusing on importance weighting (IW) based sampling approaches for GANs. Grover et al. (2019) analyse how importance weights of the GAN's outputs can lead to performance gains; extensions include their proposed usage in rejection sampling on the GAN's outputs (Azadi et al., 2018), and Metropolis–Hastings sampling from the GAN alongside improvements to the robustness of this sampling via calibration of the discriminator (Turner et al., 2019). To date, no one has leveraged these discriminator-based IW approaches in DP settings where the weights can mitigate the increased bias induced by privatised data models.

## 2.2 DIFFERENTIAL PRIVACY IN SYNTHETIC DATA GENERATION

Private synthetic data generation through DP GANs is built upon the post processing theorem: If $Di$ is $(\epsilon, \delta)$- DP, then any composition $Di \circ Ge$ is also $(\epsilon, \delta)$-DP (Dwork et al., 2014) since $Ge$ does not query the protected data. Hence, to train private GANs, we only need to privatise the training of their discriminators, see e.g. Hyland et al. (2018). Xie et al. (2018) propose DPGAN, a Wasserstein GAN which is trained by injecting noise to the gradients of the discriminator's parameters. In contrast, Jordon et al. (2019) privatise the GAN discriminator by using the Private Aggregation of Teacher Ensembles algorithm, leading to a model architecture called PATE-GAN. Recently, Torkzadehmahani et al. (2019) proposed DPCGAN as a conditional variant to DP-GAN that uses an efficient moments accountant. In contrast, PrivBayes (Zhang et al., 2017) learns a DP Bayesian network and does not rely on a GAN-architecture. Other generative approaches, for instance, include Chen et al. (2018); Acs et al. (2018). See Abay et al. (2018); Fan (2020) for an extensive overview of more DP generative approaches.

**Differentially private bias mitigation** In this paper, we offer an augmentation to the usual release procedure for synthetic data by leveraging true and estimated importance weights. Most related to our work are the contributions from Elkan (2010) and Ji and Elkan (2013) who train a regularised logistic regression model and assign weights based on the Laplace-noise-contaminated coefficients of the logistic regression. In follow up work, Ji et al. (2014) propose to modify the update step of the Newton-Raphson optimisation algorithm used in fitting the logistic regression classifier to achieve DP. However, neither of these generalise well to more complex and high dimensional settings because of the linearity of the classifier. Further, the authors assume the existence of a *public dataset* while we consider the

case where we first generate DP *synthetic data* and then weight them a posteriori, providing a generic and universally applicable approach. The benefit of learning a generative model over using public data include on the one hand that there is no requirement for the existence of a public data set, and on the other hand the possibility to generate new data points. This distinction necessitates additional analysis as the privacy budget splits between the budget spent on fitting the SDGP and the budget for estimating the IW approach. Furthermore, we show that the approach from Ji and Elkan (2013) leads to statistically biased estimation and formulate an unbiased extension with improved properties.

# 3 DIFFERENTIAL PRIVACY AND IMPORTANCE WEIGHTING

From a decision theoretic perspective, the goal of statistics is estimating expectations of functions $h : X \mapsto \mathbb{R}$, e.g. loss or utility functions, w.r.t the distribution of future uncertainties $x \sim p_D$. Given data from $\{x'_1, \ldots, x'_{N_D}\} =: x'_{1:N_D} \overset{\text{i.i.d.}}{\sim} p_D$ the data analyst can estimate these expectations consistently via the strong law of large numbers as $\mathbb{E}_{x\sim p_D}(h(x)) \approx \frac{1}{N_D}\sum_{i=1}^{N_D} h(x'_i)$. However, under DP constraints the data analyst is no longer presented with a sample from the true DGP $x'_{1:N_D} \overset{\text{i.i.d.}}{\sim} p_D$ but with a synthetic data sample $x_{1:N_G}$ from the SDGP $p_G$. Applying the naive estimator in this scenario biases the downstream tasks as $\frac{1}{N_G}\sum_{i=1}^{N_G} h(x_i) \to \mathbb{E}_{x\sim p_G}(h(x))$ almost surely.

This bias can be mitigated using a standard Monte Carlo method known as importance weighting (IW). Suppose we had access to the weights $w(x) := \frac{p_D(x)}{p_G(x)}$. If $p_G(\cdot) > 0$ whenever $h(\cdot)p_D(\cdot) > 0$, then IW relies on

$$\mathbb{E}_{x\sim p_D}[h(x)] = \mathbb{E}_{x\sim p_G}\left[w(x)h(x)\right]. \tag{2}$$

So we have almost surely for $x_{1:N_G} \overset{\text{i.i.d.}}{\sim} p_G$ the convergence

$$\mathbb{E}_{x\sim p_D}[h(x)] = \mathbb{E}_{x\sim p_G}\left[w(x)h(x)\right].$$

So we have almost surely for $x_{1:N_G} \overset{\text{i.i.d.}}{\sim} p_G$ the convergence

$$I_N(h|w) := \frac{1}{N_G}\sum_{i=1}^{N_G} w(x_i)h(x_i) \overset{N_G\to\infty}{\longrightarrow} \mathbb{E}_{x\sim p_D}[h(x)].$$

## 3.1 IMPORTANCE WEIGHTED EMPIRICAL RISK MINIMISATION

A downstream task of particular interest is the use of $x'_{1:N_D} \sim p_D$ to learn a predictive model, $f(\cdot) \in \mathcal{F}$, for the data generating distribution $p_D$ based on empirical risk minimisation. Given a loss function $h : \mathcal{F} \times X \mapsto \mathbb{R}$ comparing models $f(\cdot) \in \mathcal{F}$ with observations $x \in X$ and data $x'_{1:N_D} \sim p_D$, the principle of empirical risk minimisation (Vapnik, 1991) states that the optimal $\widehat{f}$ is given by the minimisation of

$$\frac{1}{N_D}\sum_{i=1}^{N_D} h(f(\cdot), x'_i) \approx \mathbb{E}_{x\sim p_D}\left[h(f(\cdot), x)\right]$$

over $f$. Maximum likelihood estimation (MLE) is a special case of the above with $h(f(\cdot), x_i) = -\log f(x_i|\theta)$ for a class of densities $f$ parameterised by $\theta$. Given synthetic data $x_{1:N_G} \sim p_G$, Equation (2) can be used to debias the learning of $f$.

**Remark 1** (Supplement B.5)**.** *Minimisation of the importance weight adjusted log-likelihood, $-w(x_i)\log f(x_i|\theta)$, can be viewed as an $M$-estimator (e.g. Van der Vaart, 2000) with clear relations to the standard MLE.*

**Bayesian updating.** Wilde et al. (2020) showed that naively conducting Bayesian updating using DP synthetic data without any adjustment could have negative consequences for inference. To show the versatility of our approach and to address the issues they pointed out, we demonstrate how IW can help mitigate this. The posterior distribution for parameter $\theta$ given $\tilde{x}' := x'_{1:N_D} \sim p_D$ is

$$\pi(\theta|\tilde{x}') \propto \pi(\theta)\prod_{i=1}^{N_D} f(x'_i|\theta) = \pi(\theta)\exp\left(\sum_{i=1}^{N_D}\log f(x'_i|\theta)\right)$$

where $\pi(\theta)$ denotes the prior distribution for $\theta$. This posterior is known to learn about model parameter $\theta^{\text{KLD}}_{p_D} := \arg\min_\theta \text{KLD}\left(p_D||f(\cdot|\theta)\right)$ (Berk, 1966; Bissiri et al., 2016) where KLD denotes the Kullback-Leibler divergence.

Given only synthetic data $\tilde{x} := x_{1:N_G}$ from the 'proposal distribution' $p_G$, we can use the importance weights defined in Equation (2) to construct the (generalised) posterior distribution

$$\pi_{IW}(\theta|\tilde{x}) \propto \pi(\theta)\exp\left(\sum_{i=1}^{N_G} w(x_i)\log f(x_i|\theta)\right). \tag{3}$$

In fact, Equation (3) corresponds to a generalised Bayesian posterior (Bissiri et al., 2016) with $\ell_{IW}(x_i;\theta) := -w(x_i)\log f(x_i|\theta)$, providing a coherent updating of beliefs about parameter $\theta^{\text{KLD}}_{p_D}$ using only data from the SDGP.

**Theorem 1** (Supplement B.6)**.** *The importance weighted Bayesian posterior $\pi_{IW}(\theta|x_{1:N_G})$, defined in Equation (3) for $x_{1:N_G} \overset{\text{i.i.d.}}{\sim} p_G$, admits the same limiting Gaussian distribution as the Bayesian posterior $\pi(\theta|x'_{1:N_D})$ where $x'_{1:N_D} \overset{\text{i.i.d.}}{\sim} p_D$, under regularity conditions as in (Chernozhukov and Hong, 2003; Lyddon et al., 2018).*

It is necessary here to acknowledge the existence of methods to directly conduct privatised Bayesian updating (e.g.

Dimitrakakis et al., 2014; Foulds et al., 2016; Wang et al., 2015) or M-estimation (Avella-Medina, 2021). We refer the reader Section 1 for why the attention of this paper focuses on downstream tasks for private synthetic data. We consider the application of DP IW to Bayesian updating as a natural example of such a task.

### 3.2 ESTIMATING THE IMPORTANCE WEIGHTS

The previous section shows that IW can be used to recalibrate inference for synthetic data. Unfortunately, both the DGP $p_D$ and SDGP $p_G$ densities are typically unknown, e.g. due to the intractability of GAN generation, and thus the 'perfect' weight $w(x)$ cannot be calculated. Instead, we must rely on estimates of these weights, $\widehat{w}(x)$. In this section, we show that the existing approach to DP importance weight estimation is biased, and how the data curator can correct it.

Using the same reasoning as in Section 2.1, we argue that any calibrated classification method that learns to distinguish between data from the DGP, labelled thenceforth with $y = 1$, and from the SDGP, labelled with $y = 0$, can be used to estimate the likelihood ratio (Sugiyama et al., 2012). Using Equation (1), we compute

$$\widehat{w}(x) = \frac{\widehat{p}(y = 1|x)}{\widehat{p}(y = 0|x)} \frac{N_D}{N_G}$$

where $\widehat{p}$ are the probabilities estimated by such a classification algorithm. To improve numerical stability, we can also express the log weights as

$$\log \widehat{w}(x) = \sigma^{-1}(\widehat{p}(y = 1|x)) + \log \frac{N_D}{N_G},$$

where $\sigma(x) := (1 + \exp(-x))^{-1}$ is the logistic function and $\sigma^{-1}(\widehat{p}(y = 1|x))$ are the logits of the classification method. We will now discuss two such classifiers: logistic regression and neural networks.

### 3.3 PRIVATISING LOGISTIC REGRESSION

DP guarantees for a classification algorithm $g$ can be achieved by adding noise to the training procedure. The scale of this noise is determined by how much the algorithm differs when one observation of the dataset changes. In more formal terms, the sensitivity of $g$ w.r.t a norm $|\cdot|$ is defined by the smallest number $S(g)$ such that for any two neighbouring datasets $\mathcal{D}$ and $\mathcal{D}'$ it holds that

$$|g(\mathcal{D}) - g(\mathcal{D}')| \leq S(g).$$

Dwork et al. (2006) show that to ensure the differential privacy of $g$, it suffices to add Laplacian noise with standard deviation $S(g)/\epsilon$ to $g$.

Possibly the simplest classifier $g$ one could use to estimate the importance weights is logistic regression with $L_2$ regularisation. It turns out this also has a convenient form for its sensitivity. If the data is scaled to a range from 0 to 1 such that $X \subset [0, 1]^d$, Chaudhuri et al. (2011) show that the $L_2$ sensitivity of the optimal coefficient vector estimated by $\widehat{\beta}$ in a regularised logistic regression with model

$$\widehat{p}(y = 1|x_i) = \sigma(\widehat{\beta}^T x_i) = \left(1 + e^{-\widehat{\beta}^T x_i}\right)^{-1}$$

is $S(\widehat{\beta}) = 2\sqrt{d}/(N_D\lambda)$ where $\lambda$ is the coefficient of the $L_2$ regularisation term added to the loss during training. For completeness, when the logistic regression contains an intercept parameter, we let $x_i$ denote the concatenation of the feature vector and the constant 1.

Ji and Elkan (2013) propose to compute DP importance weights by training such an $L_2$ regularised logistic classifier on the private and the synthetic data, and perturb the coefficient vector $\widehat{\beta}$ with Laplacian noise. For a $d$ dimensional noise vector $\zeta$ with $\zeta_j \overset{i.i.d.}{\sim} \text{Laplace}(0, \rho)$ with $\rho = 2\sqrt{d}/(N_D\lambda\epsilon)$ for $j \in \{1, \ldots, d\}$, the private regression coefficient is then $\overline{\beta} = \widehat{\beta} + \zeta$, akin to adding heteroscedastic noise to the private estimates of the log weights

$$\log \overline{w}(x_i) = \overline{\beta}^T x_i = \widehat{\beta}^T x_i + \zeta x_i. \tag{4}$$

The resulting privatised importance weights can be shown to lead to statistically biased estimation.

**Proposition 1** (Supplement B.1)**.** *Let $\overline{w}$ denote the importance weights computed by noise perturbing regression coefficients as in Equation* (4) *(Ji and Elkan, 2013, Algorithm 1). The IS estimator $I_N(h|\overline{w})$ is biased.*

Introducing bias on downstream estimators of sensitive information is undesirable as it can lead to an increased expected loss. To address this issue, we propose a fast and effective way for the data curator to debias the weights after computation, without requirement for an additional privacy budget.

**Proposition 2** (Supplement B.2)**.** *Let $\overline{w}$ denote the importance weights computed by noise perturbing the regression coefficients as in Equation* (4) *(Ji and Elkan, 2013, Algorithm 1) where $\zeta$ can be sampled from any noise distribution that ensures $(\epsilon, \delta)$-differential privacy of $\overline{\beta}$. Define*

$$b(x_i) := 1/\mathbb{E}_{p_\zeta}[\exp\left(\zeta^T x_i\right)],$$

*and adjusted importance weight*

$$\overline{w}^*(x_i) = \overline{w}(x_i)b(x_i) = \widehat{w}(x_i) \exp\left(\zeta^T x_i\right) b(x_i). \tag{5}$$

*The importance sampling estimator $I_N(h|\overline{w}^*)$ is unbiased and $(\epsilon, \delta)$-DP for $\mathbb{E}_{p_\zeta}[\exp\left(\zeta^T x_i\right)] > 0$.*

In Supplement B.2.4, we further show that our approach does not only decrease the bias, but also the variance of the importance weighted estimators.

For the case of component-wise independent Laplace perturbations $\zeta_j \overset{i.i.d.}{\sim} \text{Laplace}(0, \rho)$, we show that the bias correction term can be computed as

$$b(x_i) = \prod_{j=1}^{d} \left(1 - \rho^2 x_{ij}^2\right), \text{ provided } |x_{ij}| < 1/\rho \quad \forall j.$$

In practice, e.g. as we observe empirically in Section 4, the optimal choice of the regularisation term $\lambda$ is sufficiently large such that $\rho < 1$. Since the data is scaled to a range of 0 to 1 (Chaudhuri et al., 2011), this bias correction method is not limited by the restriction $|x_{ij}| < 1/\rho, \forall j$. If the data curator still encounters a case where this condition is not fulfilled, they can choose to perturb the weights with Gaussian noise instead, in which case the bias correction term always exists (see Supplement B.2.2). Laplacian perturbations are however preferred as the required noise scale can be expressed analytically without additional optimisation (Balle and Wang, 2018), and as they give stricter privacy guarantees with $\delta = 0$.

Alternatively, unbiased importance weighted estimates can be computed directly by noising the weights instead of the coefficients of the logistic regression. While this procedure removes the bias of the estimates and can also be shown to be consistent, it increases the variance to a greater extent than noising the coefficients does, and is thus only sustainable when small amounts of data are released. Please refer to Supplement A.1 for more details.

### 3.4 PRIVATISING NEURAL NETWORKS

If logistic regression fails to give accurate density ratio estimates, for example because of biases introduced by the classifier's linearity assumptions, a more complex discriminator in the form of a neural network can be trained. We can train DP classification neural networks for the aim of likelihood ratio estimation with stochastic gradient decent (SGD) by clipping the gradients and adding calibrated Gaussian noise at each step of the SGD, see e.g. Abadi et al. (2016). The noised gradients are then added up in a *lot* before the descent step where lots resemble mini-batches.

These optimisation algorithms are commonly formulated for the case when the complete dataset is private. However, in our setting, $N_D$ observations are private and $N_G$ observations are non-private. Thus, we can define a relaxed version of DP SGD. Algorithm 1 provides an overview of our proposed method. We highlight the modifications to Algorithm 1 from Abadi et al. (2016) in blue.

**Proposition 3.** *Each step in the SGD outlined in Algorithm 1 is $(\epsilon, \delta)$-differentially private w.r.t the lot and $(\mathcal{O}(q\epsilon), \delta)$ differentially private w.r.t the full dataset where $q = \frac{L}{N_D+N_G}$ and $\sigma = \sqrt{2\log\left(\frac{1.25}{\delta}\right)}/\epsilon$.*

**Algorithm 1:** Relaxed DP SGD

**Input:** Examples $x_{1:N_D}, y_{1:N_D}$ from the DGP and $x_{N_D+1:N_D+N_G}, y_{N_D+1:N_D+N_G}$ from the SDGP, loss function $\mathcal{L}(\theta) = \frac{1}{N_G+N_D}\sum_i \mathcal{L}(\theta, x_i, y_i)$. Parameters: learning rate $\eta_t$, noise scale $\sigma$, expected lot size $L$, gradient norm bound $C$.

**Initialise** $\theta_0$ randomly

**for** $t \in [T]$ **do**

  Construct a random subset $L_t \subset \{1, \ldots, N_D + N_G\}$ by including each index independently at random with probability $\frac{L}{N_D+N_G}$

  **Compute gradient**

  For each $i \in L_t$, compute $g_t(x_i, y_i) \leftarrow \Delta_{\theta_t}\mathcal{L}(\theta_t, x_i, y_i)$

  **Clip gradient**

  $\bar{g}_t(x_i, y_i) \leftarrow g_t(x_i, y_i)/\max(1, \frac{||g_t(x_i,y_i)||_2}{C})$

  **Add noise**

  $\tilde{g}_t \leftarrow \frac{1}{L}\sum_{i\in L_t}(\bar{g}_t(x_i, y_i) + N(0, \sigma^2 C^2 \mathbf{I})\mathbb{1}_{(y_i=1)})$, where $\mathbb{1}_{(y_i=1)}$ is 1 if $y_i = 1$ and 0 otherwise

  **Descent**

  $\theta_{t+1} \leftarrow \theta_t + \eta_t \tilde{g}_t$

**Output:** $\theta_T$ and the overall privacy cost $(\epsilon, \delta)$ using the moment's accountant of Abadi et al. (2016) with sampling probability $q = \frac{L}{N_D+N_G}$.

The differential privacy w.r.t a lot follows directly from the observation that the gradients of the synthetic data are already private. Further, the labels of the synthetic data are public knowledge. Lastly, the differential privacy w.r.t the dataset follows from the amplification theorem (Kasiviswanathan et al., 2011), the fact that sampling one particular private observation within a lot of size $L$ is $q = \frac{L}{N_D+N_G}$, and the reasoning behind the moment accountant of Abadi et al. (2016). We still clip the gradients of the public dataset as their influence will otherwise be overproportional under strong maximum norm assumptions.

### 3.5 GAN DISCRIMINATOR WEIGHTS

The downside of the aforementioned likelihood ratio estimators (Equation (4), Equation (5), and Algorithm 1) is that their training requires an additional privacy budget which has to be added to the privacy budget used to learn the SDGP. If we however use a GAN such as DPGAN or PATE-GAN for private synthetic data generation, we can use the GAN's discriminator for the computation of the importance weights. According to the post processing theorem, these importance weights can be released without requiring an additional pri-

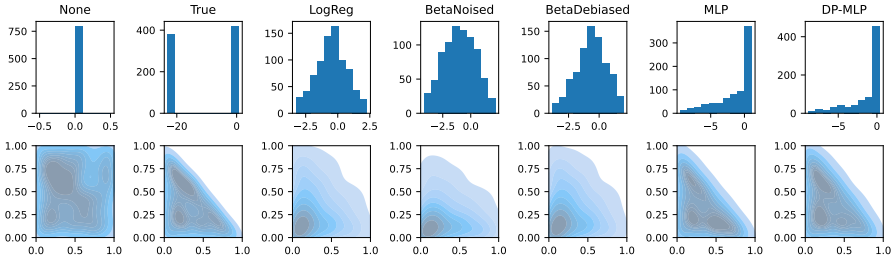



Figure 1: Kernel density plots of 100 observations sampled from a two dimensional uniform square distribution as SDGP (bottom left) and a uniform triangle distribution as DGP (second figure in second row). The first row depicts histograms of the computed weights starting with the true importance weights (True). The DP weights were privatised with $\epsilon = 1$, and the regularisation was chosen as $\lambda = 0.1$. The second row illustrates the importance weighted synthetic observations. We observe that while BetaDebiased corrects the weights of the logistic regression, the complex nature of the MLPs allows a better modelling of the DGP even in this simple setting.

vacy budget. In contrast to the weights computed from DP classification networks, the weights from this approach are thus more robust and require no additional hyperparameter tuning (confer to Section 4).

# 4 EXPERIMENTS

We demonstrate the benefits of using debiased IW for DP data release with a large-scale experimental study comparing three different SDGPs (DPGAN, DPCGAN, PrivBayes) on six real-world data sets (Iris, TGFB, Boston, Breast, Banknote, MNIST) for two different privacy budgets, $\epsilon \in \{1, 6\}$. We stress that debiasing comes with little overhead to the actual computations. As we see in Supplement C.2, the computations of the logistic regression and neural network importance weight estimates take less than one and a half minutes to train, even on MNIST. These weight estimators can be applied to any kind of synthetic data generation model, while the importance weights of the GAN discriminator can be computed in a single line of Python code and do not require any additional concerns regarding the privacy budget. Please see `https://github.com/sghalebikesabi/importance-weighted-differential-privacy` for the implementation.

**Computation of importance weights** After fitting the SDGP on the scaled true data, we weight each synthetic observation with importance weights. Based on the train and the synthetic data, we apply one of the following IW approaches: weights computed from a non-private logistic regression (LogReg), its DP alternative introduced by Ji and Elkan (2013) (BetaNoised), or our debiased proposal (BetaDebiased), and likelihood ratios estimated by a non-private multi-layer perceptron (MLP), or a DP-MLP trained using Algorithm 1. We also compare to the naive estimator using uniform weights without IW (called 'None').

Please refer to Supplement C.1 for more details on the implementation and the hyperparameters used in our experiments. In Supplement C.8, we provide a comparison to the experimental results reported by related papers. Because of the large scale of our experimental study, we present only the most important results in this section, and give a complete overview in Supplement C. The code and data for all experiments can be found online.

## 4.1 TOY EXAMPLE

We start our analysis with a simple example to illustrate the benefits of the different weighting schemes. We assume that the synthetic data is sampled from a two-dimensional uniform distribution from 0 to 1 whereas the true data follows a uniform distribution on the lower triangle given by $x_1 + x_2 < 1$ for $x_1, x_2 \in [0, 1]$. This illustrative toy example was chosen for a fairer comparison of the logistic regression and the neural network based approaches. As we see in Figure 1, the weighted kernel density estimate (KDE) of BetaDebiased is closer to the LogReg weighted KDE, and also the true KDE compared to the BetaNoised KDE.

## 4.2 UCI DATA SETS

**Datasets and preprocessing** We performed additional experiments on four UCI datasets of different characteristics as decribed in Supplement C.1: Iris, Banknote, Boston, and Breast. Similarly to Chaudhuri et al. (2011); Ji and Elkan (2013), we scale all data to a feature range from 0 to 1. We use a train-test split of 80%. In all experiments we fix $\delta$ to $N_D^{-1} - 10^{-6}$, and choose $\epsilon \in \{1, 6\}$. We refer to Supplement C.7 for a complete overview of the results.

**Synthetic data generators** We used DPCGAN (Torkzadehmahani et al., 2019), DPGAN (Xie et al., 2018), and their corresponding non-DP analogues (CGAN and CGAN) to generate DP synthetic data of the same size as the training

| | IW | Breast | | | Banknote | | |
|---|---|---|---|---|---|---|---|
| | | DPGAN | DPCGAN | PrivBayes | DPGAN | DPCGAN | PrivBayes |
| WST ↓ | None | $2.3665_{\pm 0.0982}$ | $1.5853_{\pm 0.1333}$ | $2.1117_{\pm 0.1740}$ | $0.4746_{\pm 0.0214}$ | $0.7442_{\pm 0.0333}$ | $0.3237_{\pm 0.0162}$ |
| | BetaNoised | $1.4337_{\pm 0.1114}$ | $2.2232_{\pm 0.2325}$ | $1.2322_{\pm 0.0823}$ | $0.2509_{\pm 0.0436}$ | $0.4355_{\pm 0.0456}$ | $0.2318_{\pm 0.0035}$ |
| | BetaDebiased | $1.8922_{\pm 0.1237}$ | $1.9913_{\pm 0.3507}$ | $\mathbf{1.1825_{\pm 0.0933}}$ | $0.4015_{\pm 0.0766}$ | $0.4618_{\pm 0.0832}$ | $0.2369_{\pm 0.0061}$ |
| | DP-MLP | $1.4570_{\pm 0.1492}$ | $1.0315_{\pm 0.1415}$ | $1.2190_{\pm 0.0795}$ | $\mathbf{0.2035_{\pm 0.0427}}$ | $0.4298_{\pm 0.0433}$ | $\mathbf{0.0456_{\pm 0.0061}}$ |
| | Discriminator | $\mathbf{1.0007_{\pm 0.0004}}$ | $\mathbf{1.0001_{\pm 0.0001}}$ | - | $0.3382_{\pm 0.0399}$ | $\mathbf{0.1087_{\pm 0.0415}}$ | - |
| | LogReg | $1.6451_{\pm 0.1168}$ | $2.2953_{\pm 0.2121}$ | $1.4663_{\pm 0.1152}$ | $0.2508_{\pm 0.0432}$ | $0.4348_{\pm 0.0460}$ | $0.2348_{\pm 0.0034}$ |
| | MLP | $1.6129_{\pm 0.1404}$ | $1.0709_{\pm 0.1579}$ | $1.4141_{\pm 0.1216}$ | $0.0913_{\pm 0.0259}$ | $0.3860_{\pm 0.0452}$ | $0.0021_{\pm 0.0004}$ |
| $\beta$ MSE ↓ | None | $2.0643_{\pm 0.2012}$ | $4.9828_{\pm 1.5701}$ | $2.3904_{\pm 0.1050}$ | $11.0215_{\pm 1.8377}$ | $19.3243_{\pm 3.7708}$ | $8.1724_{\pm 0.3987}$ |
| | BetaNoised | $2.7532_{\pm 0.2650}$ | $2.5025_{\pm 0.3763}$ | $2.1144_{\pm 0.2400}$ | $8.4298_{\pm 1.0383}$ | $15.2862_{\pm 4.0365}$ | $5.7001_{\pm 0.1885}$ |
| | BetaDebiased | $2.8337_{\pm 0.3842}$ | $\mathbf{2.2324_{\pm 1.0446}}$ | $\mathbf{1.8266_{\pm 0.2392}}$ | $\mathbf{8.3508_{\pm 2.3127}}$ | $12.9909_{\pm 5.9024}$ | $6.6862_{\pm 0.1458}$ |
| | DP-MLP | $2.3965_{\pm 0.2083}$ | $3.8865_{\pm 0.6043}$ | $2.3130_{\pm 0.2195}$ | $17.1597_{\pm 2.5448}$ | $16.4618_{\pm 4.1011}$ | $\mathbf{3.5519_{\pm 0.2895}}$ |
| | Discriminator | $\mathbf{1.4591_{\pm 0.1837}}$ | $4.0612_{\pm 0.9523}$ | - | $12.5471_{\pm 2.3124}$ | $\mathbf{10.9282_{\pm 5.4283}}$ | - |
| | LogReg | $2.6934_{\pm 0.2667}$ | $2.2156_{\pm 0.3366}$ | $1.5333_{\pm 0.2138}$ | $8.4760_{\pm 1.0406}$ | $15.2964_{\pm 4.0396}$ | $5.6751_{\pm 0.1785}$ |
| | MLP | $2.3999_{\pm 0.2040}$ | $3.8343_{\pm 0.7032}$ | $1.6581_{\pm 0.2020}$ | $17.9390_{\pm 2.4926}$ | $15.5211_{\pm 4.2147}$ | $2.6286_{\pm 0.3761}$ |
| MLP ROC-AUC ↑ | None | $0.6374_{\pm 0.0421}$ | $0.6791_{\pm 0.0966}$ | $0.8366_{\pm 0.0579}$ | $0.8546_{\pm 0.0213}$ | $0.6863_{\pm 0.0436}$ | $0.7630_{\pm 0.0495}$ |
| | BetaNoised | $0.6110_{\pm 0.0477}$ | $0.6546_{\pm 0.0727}$ | $0.7076_{\pm 0.0983}$ | $0.8495_{\pm 0.0274}$ | $0.6063_{\pm 0.0510}$ | $0.8943_{\pm 0.0173}$ |
| | BetaDebiased | $0.6820_{\pm 0.0510}$ | $0.7173_{\pm 0.0842}$ | $\mathbf{0.8557_{\pm 0.0765}}$ | $\mathbf{0.8729_{\pm 0.0310}}$ | $0.5868_{\pm 0.1005}$ | $0.7632_{\pm 0.0517}$ |
| | DP-MLP | $\mathbf{0.7942_{\pm 0.0404}}$ | $0.5686_{\pm 0.0823}$ | $0.7353_{\pm 0.0887}$ | $0.7697_{\pm 0.0419}$ | $0.5657_{\pm 0.0570}$ | $\mathbf{0.8953_{\pm 0.0299}}$ |
| | Discriminator | $0.6992_{\pm 0.0839}$ | $\mathbf{0.7290_{\pm 0.0720}}$ | - | $0.8695_{\pm 0.0167}$ | $\mathbf{0.7114_{\pm 0.0424}}$ | - |
| | LogReg | $0.6631_{\pm 0.0469}$ | $0.6484_{\pm 0.1081}$ | $0.7618_{\pm 0.1019}$ | $0.8172_{\pm 0.0327}$ | $0.6034_{\pm 0.0534}$ | $0.9102_{\pm 0.0129}$ |
| | MLP | $0.7730_{\pm 0.0412}$ | $0.7358_{\pm 0.1017}$ | $0.7573_{\pm 0.0738}$ | $0.8291_{\pm 0.0333}$ | $0.5974_{\pm 0.0627}$ | $0.8594_{\pm 0.0231}$ |

Table 1: Mean and standard error over 10 runs for ($\epsilon = 1$, $\delta = N_D^{-1} - e^{-6}$) on the Breast and Banknote data. Best score out of the private methods is marked in bold.

data set. Additionally we also consider PrivBayes (Zhang et al., 2017), a DP Bayesian Network, as a potential SDGP.

**Hyperparameter tuning** Hyperparameter tuning is essentially non-private, and has to be accounted for in the privacy budget. Since hyperparemeter tuning in a DP setting is an unresolved problem (Liu and Talwar, 2019; Rosenblatt et al., 2020; Papernot and Steinke, 2021), we follow Jordon et al. (2019) and tune the hyperparameters of the underlying baselines on private validation data sets. However, we propose default parameters for our methods. This leads to an over-optimistic presentation of the baseline performance, and a conservative presentation of our extensions.

**Evaluation metrics** In order to show that IW decreases statistical bias, we train a linear prediction model on the synthetic data and approximate its bias. Since the true DGP is not known, we train the same linear predictor on the test data and report the mean squared error (MSE) between the test parameters and the parameters estimated on the SDGP, as $\beta$ MSE. We further analyse the divergence of the weighted SDGP and the DGP in a similar way by computing the Wassertstein (WST) distance w.r.t the test data. As one exemplary supervised downstream task, we consider a linear downstream classifier or regressor trained on the synthetic data. This downstream predictor is then assessed by the error measured in the parameter vector compared to the parameters learnt using the test set ($beta$ MSE). Finally, we train a one-hidden-layer MLP on the training data, and report the test prediction error as MLP ROC-AUC for classification tasks, and MLP MSE for regression tasks.

**Choice of budget split** We only present results for $\epsilon = 1$ in this section, and refer the reader to Supplement C.7 for further results with $\epsilon = 6$. If the weight computation procedure requires a separate privacy budget (e.g. if the weights are computed by a separate MLP or logistic regression), we spend 10% of the $\epsilon$-budget on fitting the SDGP and 30% of the $\delta$-budget on the weight computation; the complete budget can be spent on fitting the SDGP if no weights, or the weights of the discriminator are used. In Supplement C.3, we evaluate a range of different privacy splits on the Breast and Boston data.

**Results** In Tables 1 and 2, we see that the performance of the models mostly improved when weighted with any type of estimated weights. Although the best inference for each data set is nearly always achieved after importance weighting, we notice that there are some rare cases where no importance weighting performs (insignificantly) better. For instance, we observe that the SDGP obtained with PrivBayes seems to be close to the true DGP of the Boston Housing data, and that importance weighting is no longer helpful. In settings where the SDGP and the DGP are really close, it is possible that the effects of additional variance induced by estimating and privatising the importance weights (where appropriate) cancels out the reduction in bias. This effect might be mitigated with hyperparameter tuning. Further, we note that debiasing the logistic regression weights mainly results in better performance. Even though we experience a slight

| | IW | DPGAN | PrivBayes |
|---|---|---|---|
| WST ↓ | None | $2.2013_{\pm 0.0945}$ | $1.3938_{\pm 0.0231}$ |
| | BetaNoised | $2.0922_{\pm 0.0419}$ | $1.3009_{\pm 0.0338}$ |
| | BetaDebiased | $2.0930_{\pm 0.0393}$ | $1.2705_{\pm 0.0290}$ |
| | DP-MLP | $2.0542_{\pm 0.0184}$ | $\mathbf{1.0265_{\pm 0.0035}}$ |
| | Discriminator | $\mathbf{2.0145_{\pm 0.0141}}$ | - |
| | LogReg | $2.2051_{\pm 0.0819}$ | $1.4078_{\pm 0.0492}$ |
| | MLP | $2.0350_{\pm 0.0158}$ | $1.0072_{\pm 0.0009}$ |
| $\beta$ MSE ↓ | None | $0.1867_{\pm 0.0434}$ | $\mathbf{0.0011_{\pm 0.0002}}$ |
| | BetaNoised | $0.1761_{\pm 0.0948}$ | $0.0088_{\pm 0.0028}$ |
| | BetaDebiased | $\mathbf{0.0667_{\pm 0.0188}}$ | $0.0077_{\pm 0.0022}$ |
| | DP-MLP | $0.1530_{\pm 0.0812}$ | $0.0048_{\pm 0.0024}$ |
| | Discriminator | $0.1567_{\pm 0.1825}$ | - |
| | LogReg | $0.0749_{\pm 0.0279}$ | $0.0037_{\pm 0.0016}$ |
| | MLP | $0.1476_{\pm 0.0804}$ | $0.0008_{\pm 0.0002}$ |
| MLP MSE ↓ | None | $1.8851_{\pm 0.5262}$ | $0.1973_{\pm 0.0108}$ |
| | BetaNoised | $1.0057_{\pm 0.1973}$ | $0.2200_{\pm 0.0154}$ |
| | BetaDebiased | $\mathbf{0.9024_{\pm 0.1244}}$ | $0.2139_{\pm 0.0122}$ |
| | DP-MLP | $0.9462_{\pm 0.1702}$ | $\mathbf{0.1877_{\pm 0.0174}}$ |
| | Discriminator | $1.6256_{\pm 0.2394}$ | - |
| | LogReg | $1.0606_{\pm 0.2648}$ | $0.2515_{\pm 0.0305}$ |
| | MLP | $1.0979_{\pm 0.2225}$ | $0.1697_{\pm 0.0079}$ |

Table 2: Mean and standard error over 10 runs for ($\epsilon = 1$, $\delta = N_D^{-1} - e^{-6}$) on the Boston Housing data. Best score out of the private methods is marked in bold.

| IW | $\beta$ MSE ↓ | MLP ROC-AUC ↑ |
|---|---|---|
| None | $0.6605_{\pm 0.0384}$ | $0.8502_{\pm 0.0386}$ |
| BetaNoised | $0.6247_{\pm 0.0184}$ | $0.8766_{\pm 0.0086}$ |
| BetaDebiased | $0.6240_{\pm 0.0179}$ | $\mathbf{0.8783_{\pm 0.0093}}$ |
| DP-MLP | $\mathbf{0.5813_{\pm 0.0246}}$ | $0.8683_{\pm 0.0055}$ |
| Discriminator | $0.6242_{\pm 0.0140}$ | $0.8631_{\pm 0.0310}$ |
| LogReg | $0.6234_{\pm 0.0183}$ | $0.8770_{\pm 0.0092}$ |
| MLP | $0.5707_{\pm 0.0207}$ | $0.8737_{\pm 0.0058}$ |

Table 3: Mean and standard error over 10 runs with standard errors for ($\epsilon = 9.64, \delta = 60,000^{-1} - e^{-6}$) on MNIST.

drop in performance from BetaNoised to BetaDebiased in some rare cases, this can be explained by randomness in the data set as we show in Supplement Table 3 that the weights estimated by BetaDebiased are significantly closer to the true LogReg weights than the importance weights given by BetaNoised. If a GAN is used as SDGP, and the data curator is hesitant to release additional importance weights, the discriminator weights nearly always lead to an improvement in results without requiring additional computations. To further illustrate the practical meaning of debiasing, we have included an exemplary case study in Supplement C.6.

### 4.3 BAYESIAN UPDATING WITH IW

We investigate the effectiveness of IW in a Bayesian learning setting as per Equation 3. We evaluated and compared the performance of these weighted posteriors alongside the standard non-weighted posterior by applying them to learning the parameters of models for various regression tasks. Figure 2 shows the ROC-AUC scores associated with the Bayesian predictive distribution arising from integration over the posterior of a Bayesian logistic regression model fit on synthesised versions of the Banknote dataset. We observe that the ROC-AUC under PrivBayes' synthetic data is significantly improved upon across all IW methods, with similar gains made to the median performance under CGAN's synthetic data. Additionally, most of the methods help in decreasing variability in the results, especially DP-MLP and MLP. See Supplement C.5 for a full specification of the experimental details and for further results from fitting Bayesian linear regression and multinomial logistic regression models on the TGFB and Iris datasets respectively.

### 4.4 MNIST

Additionally, we assessed how IW performs in a high-dimensional setting such as a classification task on the MNIST dataset. Since PrivBayes does not scale to large data sets, we only evaluate DPCGAN as possible SDGP. For this we follow the setup by Torkzadehmahani et al. (2019) for $\epsilon = 9.64$ and $\delta = 6000^{-1} - 10^{-6}$. We observe in Table 3 that all IW methods improve upon the state of the art.

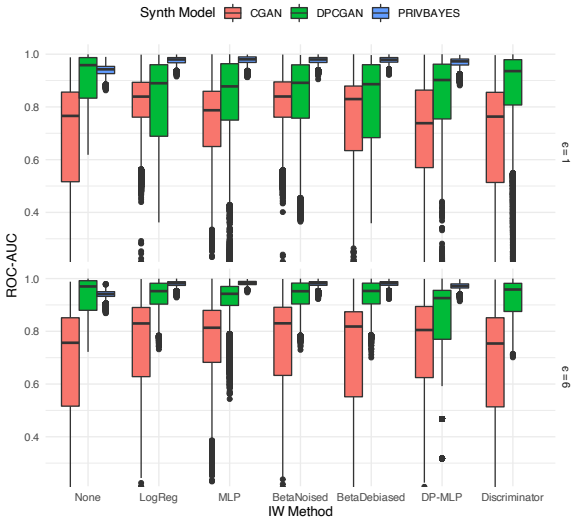



Figure 2: ROC-AUC score box plots calculated via chains of parameters sampled from a Bayesian logistic regression model fit on synthesised Banknote data across 10 seeds.

## 5 DISCUSSION

In this paper, we investigated importance weighting methods to correct for biases in downstream estimation tasks when using differentially private synthetic data. While classification algorithms can be used to estimate the required importance

weights, noise must be added in order to maintain privacy. We presented methods to debias inference based on privatised weights estimated by logistic regression, developed private estimation procedures allowing the complexity of neural networks to be leveraged for weight estimation, and proposed using inbuilt discriminator weights from GAN data generation to avoid increases to the privacy budget.

Following these developments, we advocate that future releases of DP synthetic data are augmented with privatised importance weights to allow researchers to conduct unbiased downstream model estimation. Future work will focus on improved hyperparameter tuning practises to choose the optimal IW approach for the task and dataset at hand. Further improving upon the likelihood ratio estimates in a non-private setting could simplify such a choice.

## Acknowledgements

SG is a student of the EPSRC CDT in Modern Statistics and Statistical Machine Learning (EP/S023151/1) and receives funding from the Oxford Radcliffe Scholarship and Novartis. HW is supported by the Feuer International Scholarship in Artificial Intelligence. JJ was funded by the Ayudas Fundación BBVA a Equipos de Investigación Cientifica 2017 and Government of Spain's Plan Nacional PGC2018-101643-B-I00 grants whilst working on this project. SJV is supported by the University of Warwick, University of Warwick and German Resarch Centre for Aritifical Intelligence. CH is supported by The Alan Turing Institute, Health Data Research UK, the Medical Research Council UK, the EPSRC through the Bayes4Health programme Grant EP/R018561/1, and AI for Science and Government UK Research and Innovation (UKRI).

## References

Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, pages 308–318, 2016.

Nazmiye Ceren Abay, Yan Zhou, Murat Kantarcioglu, Bhavani Thuraisingham, and Latanya Sweeney. Privacy preserving synthetic data release using deep learning. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 510–526. Springer, 2018.

Gergely Acs, Luca Melis, Claude Castelluccia, and Emiliano De Cristofaro. Differentially private mixture of generative neural networks. *IEEE Transactions on Knowledge and Data Engineering*, 31(6):1109–1121, 2018.

Marco Avella-Medina. Privacy-preserving parametric inference: a case for robust statistics. *Journal of the American Statistical Association*, 116(534):969–983, 2021.

Samaneh Azadi, Catherine Olsson, Trevor Darrell, Ian Goodfellow, and Augustus Odena. Discriminator rejection sampling. *arXiv preprint arXiv:1810.06758*, 2018.

Borja Balle and Yu-Xiang Wang. Improving the gaussian mechanism for differential privacy: Analytical calibration and optimal denoising. In *International Conference on Machine Learning*, pages 394–403. PMLR, 2018.

Robert H Berk. Limiting behavior of posterior distributions when the model is incorrect. *The Annals of Mathematical Statistics*, pages 51–58, 1966.

Pier Bissiri, Chris Holmes, and Stephen Walker. A general framework for updating belief distributions. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 2016.

Kamalika Chaudhuri, Claire Monteleoni, and Anand D Sarwate. Differentially private empirical risk minimization. *Journal of Machine Learning Research*, 12(3), 2011.

Qingrong Chen, Chong Xiang, Minhui Xue, Bo Li, Nikita Borisov, Dali Kaarfar, and Haojin Zhu. Differentially private data generative models. *arXiv preprint arXiv:1812.02274*, 2018.

Victor Chernozhukov and Han Hong. An MCMC approach to classical estimation. *Journal of Econometrics*, 115(2): 293–346, 2003.

Christos Dimitrakakis, Blaine Nelson, Aikaterini Mitrokotsa, and Benjamin IP Rubinstein. Robust and private bayesian inference. In *International Conference on Algorithmic Learning Theory*, pages 291–305. Springer, 2014.

Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of cryptography conference*, pages 265–284. Springer, 2006.

Cynthia Dwork, Aaron Roth, et al. The algorithmic foundations of differential privacy. *Foundations and Trends in Theoretical Computer Science*, 9(3-4):211–407, 2014.

Charles Elkan. Preserving privacy in data mining via importance weighting. In *International Workshop on Privacy and Security Issues in Data Mining and Machine Learning*, pages 15–21. Springer, 2010.

Liyue Fan. A survey of differentially private generative adversarial networks. In *The AAAI Workshop on Privacy-Preserving Artificial Intelligence*, 2020.

James Foulds, Joseph Geumlek, Max Welling, and Kamalika Chaudhuri. On the theory and practice of privacy-preserving bayesian data analysis. *arXiv preprint arXiv:1603.07294*, 2016.

Lorenzo Frigerio, Anderson Santana de Oliveira, Laurent Gomez, and Patrick Duverger. Differentially private generative adversarial networks for time series, continuous, and discrete open data. In *IFIP International Conference on ICT Systems Security and Privacy Protection*, pages 151–164. Springer, 2019.

Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *arXiv preprint arXiv:1406.2661*, 2014.

Aditya Grover, Jiaming Song, Ashish Kapoor, Kenneth Tran, Alekh Agarwal, Eric J Horvitz, and Stefano Ermon. Bias correction of learned generative models using likelihood-free importance weighting. In *Advances in Neural Information Processing Systems*, pages 11058–11070, 2019.

Moritz Hardt and Guy N Rothblum. A multiplicative weights mechanism for privacy-preserving data analysis. In *2010 IEEE 51st Annual Symposium on Foundations of Computer Science*, pages 61–70. IEEE, 2010.

Stephanie Hyland, Cristóbal Esteban, and Gunnar Rätsch. Real-valued (medical) time series generation with recurrent conditional gans. *arXiv*, 2018.

Zhanglong Ji and Charles Elkan. Differential privacy based on importance weighting. *Machine Learning*, 93(1):163–183, 2013.

Zhanglong Ji, Xiaoqian Jiang, Shuang Wang, Li Xiong, and Lucila Ohno-Machado. Differentially private distributed logistic regression using private and public data. *BMC medical genomics*, 7(1):1–10, 2014.

James Jordon, Jinsung Yoon, and Mihaela van der Schaar. Pate-gan: Generating synthetic data with differential privacy guarantees. In *International Conference on Learning Representations*, 2019.

Shiva Prasad Kasiviswanathan, Homin K Lee, Kobbi Nissim, Sofya Raskhodnikova, and Adam Smith. What can we learn privately? *SIAM Journal on Computing*, 40(3): 793–826, 2011.

Moshe Lichman. UCI machine learning repository, 2013.

Jingcheng Liu and Kunal Talwar. Private selection from private candidates. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, pages 298–309, 2019.

Terrance Liu, Giuseppe Vietri, Thomas Steinke, Jonathan Ullman, and Steven Wu. Leveraging public data for practical private query release. In *International Conference on Machine Learning*, pages 6968–6977. PMLR, 2021.

Simon P Lyddon, Chris Holmes, and Stephen Walker. General Bayesian updating and the loss-likelihood bootstrap. *Biometrika*, 2018.

Marcel Neunhoeffer, Zhiwei Steven Wu, and Cynthia Dwork. Private post-GAN boosting. *arXiv preprint arXiv:2007.11934*, 2020.

Nicolas Papernot and Thomas Steinke. Hyperparameter tuning with renyi differential privacy. *arXiv preprint arXiv:2110.03620*, 2021.

Lucas Rosenblatt, Xiaoyan Liu, Samira Pouyanfar, Eduardo de Leon, Anuj Desai, and Joshua Allen. Differentially Private Synthetic Data: Applied Evaluations and Enhancements. *arXiv*, Nov 2020. URL `https://arxiv.org/abs/2011.05537v1`.

Masashi Sugiyama, Taiji Suzuki, and Takafumi Kanamori. *Density Ratio Estimation in Machine Learning*. Cambridge University Press, 2012.

Reihaneh Torkzadehmahani, Peter Kairouz, and Benedict Paten. Dp-cgan: Differentially private synthetic data and label generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0, 2019.

Ryan Turner, Jane Hung, Eric Frank, Yunus Saatchi, and Jason Yosinski. Metropolis–Hastings generative adversarial networks. In *International Conference on Machine Learning*, pages 6345–6353. PMLR, 2019.

Aad W Van der Vaart. *Asymptotic Statistics*, volume 3. Cambridge University Press, 2000.

Yu-Xiang Wang, Stephen Fienberg, and Alex Smola. Privacy for free: Posterior sampling and stochastic gradient monte carlo. In *International Conference on Machine Learning*, pages 2493–2502. PMLR, 2015.

Harrison Wilde, Jack Jewson, Sebastian Vollmer, and Chris Holmes. Foundations of Bayesian learning from synthetic data. *arXiv preprint arXiv:2011.08299*, 2020.

Liyang Xie, Kaixiang Lin, Shu Wang, Fei Wang, and Jiayu Zhou. Differentially private generative adversarial network. *arXiv preprint arXiv:1802.06739*, 2018.

Jun Zhang, Graham Cormode, Cecilia M Procopiuc, Divesh Srivastava, and Xiaokui Xiao. PrivBayes: Private data release via Bayesian networks. *ACM Transactions on Database Systems (TODS)*, 42(4):1–41, 2017.

Xinyang Zhang, Shouling Ji, and Ting Wang. Differentially private releasing via deep generative model. *arXiv preprint arXiv:1801.01594*, 2018.

# Mitigating Statistical Bias within Differentially Private Synthetic Data (Supplementary Material)

**Sahra Ghalebikesabi**[1] **Harrison Wilde**[2] **Jack Jewson**[3] **Arnaud Doucet**[1]
**Sebastian Vollmer**[5] **Chris Holmes**[1]

[1]University of Oxford
[2]University of Warwick
[3]Universitat Pompeu Fabra
[5]University of Kaiserslautern, German Research Centre for Artificial Intelligence (DFKI)

## A ADDITIONAL MATERIAL

### A.1 UNBIASED IMPORTANCE WEIGHTING BY OUTPUT PERTURBATION

A simple approach to ensure DP of an algorithm is to add noise (Dwork et al., 2006) to its output, that is the estimated importance weights of the synthetic data. We establish general results under which such a noise perturbation of an unbiased non-private weights algorithm $\widehat{w}(x)$ preserves the unbiasedness of IS estimation.

**Theorem 1.** *Let $\sigma^2(h)/N$ denote the variance of the IS estimate $I_N(h|w)$ defined in Equation* (2)*. Then the IS estimator $I_N(h|w^*)$ using noise perturbed importance weights $w^*(x_i) = \widehat{w}(x_i) + \zeta_i$, where $\zeta_i$ are i.i.d. and $\mathbb{E}[\exp(\zeta_i)] = 1$, is unbiased and has variance $\sigma^{*2}(h)/N$ where*

$$\sigma^{*2}(h) = \sigma^2(h) + \text{Var}\left[\exp(\zeta)\right] \mathbb{E}_{p_G}[(\widehat{w}(x)h(x))^2]. \tag{1}$$

We refer the reader to Supplement B.3 for the proof. In the following we will analyse how the noise $\zeta$ has to be chosen to ensure DP.

**Corollary 1.** *The IS estimator with importance weights defined by*

$$\log w^*(x_i) = \widehat{\beta}^T x_i + \zeta_i \tag{2}$$

$$\textit{for} \quad \zeta_i \sim \textit{Laplace}(\log(1-\rho^2), \rho) \textit{ and } \quad \rho = \frac{2\sqrt{d}}{N_D \lambda \epsilon} < 1$$

*is $(N_S\epsilon, 0)$-differentially private. It is further unbiased and for $\rho < \frac{1}{2}$ has variance as defined in equation 1:*

$$\text{Var}\left[\exp(\zeta)\right] = \exp(2\log(1-\rho^2)) \left( \frac{1}{1-4\lambda^2} - \frac{1}{(1-\lambda^2)^2} \right).$$

Note that privacy budget is additive. If we want to release $N_S$ DP weights, we thus have to scale the noise proportional to $N_S$. Although this approach increases the variance of the estimator, it remains unbiased.

A limitation of this approach is that $\rho < 1/2$. Alternatively, Blum et al. (2005) show that adding Gaussian noise $\zeta' \sim N(0, \frac{2}{\epsilon^2} S(f)^2 \log \frac{2}{\delta})$ to an algorithm $f$ ensures $(\epsilon, \delta)$-DP for $\delta > 0$. From our analysis it follows that we could adjust Corollary 1 as follows.

*Accepted for the 38th Conference on Uncertainty in Artificial Intelligence* (UAI 2022).

**Corollary 2.** *The IS estimator with importance weights defined by*

$$\log w^*(x_i) = \widehat{\beta}^T x_i + \zeta_i'$$
$$\textit{for} \quad \zeta_i' \sim N(-\frac{\gamma^2}{2}, \gamma^2) \quad \textit{and} \quad \gamma = \sqrt{\frac{8d}{(N_D \lambda \epsilon)^2} \log \frac{2}{\delta}}$$

*is* $(N_S\epsilon, \delta)$*-differentially private with* $\delta > 0$ *and* $\epsilon < 1$*. It is further unbiased and has variance as defined in equation 1 with* $\text{Var}\left[\exp(\zeta')\right] = \gamma^2$.

This result trivially extends to the case of $\epsilon \geq 1$ with accordingly adjusted noise scales following results from Balle and Wang (2018).

**Sources of Bias and Variance.** This analysis gives us insights on two sources of bias and variance. The first one is the bias and/or variance introduced by *privatising* the weights. The estimator of Ji and Elkan (2013) is biased but as a result adds noise with a smaller variance, whereas to be unbiased by noising the weights we have to pay a price of increasing the variance, e.g., by adding more noise or by releasing fewer samples. The second source is the bias and variance introduced by *estimating* the weights through the classifier. The importance weighting procedure is only unbiased when we know exactly how to estimate the true weights. Using a logistic regression to estimate these cannot reasonably be considered as unbiased for any complicated data. However, using an arbitrarily complex classifier such as a classification neural network could arguably be considered as less biased at estimating the density ratio if it converges, but possibly increases the variance of the estimators due to the increased number of parameters to learn.

### A.2 POST-PROCESSING OF LIKELIHOOD RATIOS

The performance of importance weighting can suffer from a heavy right tailed distribution of the likelihood ratio estimates which increases the variance of downstream estimators. A simple remedy is tempering: for a $\tau \in [0, 1]$ the weights $\{\widehat{w}(x_i)^\tau\}_{i \in \{1,\dots,N_G\}}$ are less extreme.

Alternatively, Vehtari et al. (2015) propose Pareto smoothed IS (PSIS). This procedure requires to fit a generalised Pareto distribution to the upper tail of the distribution of the simulated importance ratios. Their algorithm does not only post-hoc stabilise IS, but also reports a warning when the estimated shape parameter of the Pareto distribution exceeds a certain threshold. Similarly, Koopman et al. (2009) propose a test to detect whether importance weights have finite variance. In both warnings, there are certain characteristics of the DGP which are not captured by the SDGP and the resulting IS estimates are likely to be unstable. This warning can thus be understood as a general indicator for unsuitable proposal distributions. For large shape parameters the data owner should not release the SDGP. It is also computationally more efficient than comparable distribution divergences such as maximum mean discrepancy or Wasserstein distance. We must also consider that unlike traditional IS where the importance weights are known (at least up to normalisation), here they are being estimated from data, providing further motivation for regularisation.

Aside from unstable likelihood ratios, the computed importance weights can suffer from the inability of the classification method to correctly capture the density ratios. To mitigate this problematic, Turner et al. (2019) propose post-calibration of the likelihood ratios in a non-private setting. If we can assume that the data analyst has access to a small dataset of the DGP, as e.g. in Wilde et al. (2020), we can make use of post-calibration methods, such as beta calibration (Kull et al., 2017).

## B PROOFS

### B.1 PROPOSITION 1: BIAS AND VARIANCE OF ALGORITHM 1 OF JI & ELKAN (2013)

Consider Ji and Elkan (2013) Algorithm 1, where under the assumption that $\frac{p(y=1)}{p(y=0)} \approx \frac{N_D}{N_S} = 1$, the unprivatised importance weights are estimated using logistic regression

$$\widehat{w}(x_i) = \frac{p^*(y=1|x_i)}{p^*(y=0|x_i)} = \exp\left(\widehat{\beta}^T x_i\right),$$

and then the privacy preserving process adds noise to the $\widehat{\beta}$ coefficients of this logistic regression $\beta^* = \widehat{\beta} + \zeta$ with $\zeta \sim \text{Laplace}(2\sqrt{d}/(N_D\lambda\epsilon))$, a vector of length $d$, to generate privatised estimates of the importance weights

$$\overline{w}(x_i) = \exp\left({\beta^*}^T x_i\right) = \exp\left(\widehat{\beta}^T x_i\right) \cdot \exp\left(\zeta x_i\right). \tag{3}$$

The following proposition proves that $\overline{w}(x_i)$ is a *biased* estimate of $\widehat{w}(x_i)$, the consequences being that if the 'true' importance weight really is given by a logistic regression then the procedure of Ji and Elkan (2013) will be biased.

**Proposition 1.** *Let $\overline{w}$ denote the importance weights computed by noise perturbing the regression coefficients as in Equation* (3) *(Ji and Elkan, 2013, Algorithm 1). The importance sampling estimator $I_N(h|\overline{w})$ is biased.*

**Proof.** Firstly, we show that $\overline{w}(x_i)$ is not an unbiased estimate of $\widehat{w}(x_i)$

$$\begin{aligned}
\mathbb{E}_\zeta\left[\overline{w}(x_i)\right] &= \mathbb{E}_\zeta\left[\exp\left(\widehat{\beta}^T x_i\right) \cdot \exp\left(\zeta x_i\right)\right] \\
&= \mathbb{E}_\zeta\left[\widehat{w}(x_i) \cdot \exp\left(\zeta x_i\right)\right] \\
&\neq \widehat{w}(x_i).
\end{aligned}$$

As a consequence, we show that even if the true density ratio can be captured by a logistic regression, i.e. there exists $\beta_0$ such that $\frac{p_D(x)}{p_G(x)} = \exp\left(\beta_0^T x\right)$, then the importance sampling estimator

$$I_N(h|\overline{w}) = \frac{1}{N}\sum_{i=1}^{N} \overline{w}(x_i)h(x_i), \quad x_i \sim p_G(\cdot),$$

with $\overline{w}(\cdot)$ calculated using 'privatised' $\beta^* = \beta_0 + \zeta$, $\zeta$ distributed as above, is a biased estimate of $\mathbb{E}_{p_D}\left[h(x)\right]$. Indeed, we have

$$\begin{aligned}
\mathbb{E}_{x_{1:N}\sim p_G}\left[\frac{1}{N}\sum_{i=1}^{N} \overline{w}(x_i)h(x_i)\right] &= \mathbb{E}_{x_{1:N}\sim p_G}\left[\frac{1}{N}\sum_{i=1}^{N} \exp\left(\beta_0^T x_i\right) \cdot \exp\left(\zeta x_i\right) h(x_i)\right] \\
&= \frac{1}{N}\sum_{i=1}^{N} \mathbb{E}_{x_i\sim p_G}\left[w\left(x_i\right) \cdot \exp\left(\zeta x_i\right) h(x_i)\right] \\
&= \frac{1}{N}\sum_{i=1}^{N} \mathbb{E}_{x_i\sim p_D}\left[\exp\left(\zeta x_i\right) h(x_i)\right] \\
&\neq \mathbb{E}_{x_i\sim p_D}\left[h(x_i)\right].
\end{aligned}$$

The proof of Proposition 1 provides several insights on what is required for an unbiased estimator. The fact that the bias depends explicitly on the observation suggests either 1) asking the data curator to debias the noise given the synthetic data they are about to release or 2) adding noise to the weights themselves rather to the process of how they are calculated.

Ji and Elkan (2013) compute the variance of the estimator $\beta^* = \widehat{\beta} + \zeta$ where $\zeta \sim \text{Laplace}(\frac{4(d+1)d}{(N_D\lambda\epsilon)^2})$ as

$$\text{Var}(\beta^*) = \text{Var}(\widehat{\beta}) + \text{Var}(\zeta) = \text{Var}(\widehat{\beta}) + \frac{4(d+1)d}{(N_D\lambda\epsilon)^2}.$$

They show that the asymptotic variance of importance sampling with the unperturbed weights obtained from the logistic regression $w_{logreg}$ can be upper bounded by

$$\text{Var}(I_N(h, w_{logreg})) = \alpha^T \text{Var}(\widehat{\beta})\alpha = \alpha^T \frac{dI_d}{N_D\lambda^2}\alpha$$

with

$$\alpha = \frac{\sum_{x_i,x_j\in D} e^{\beta_0^T(x_i+x_j)}\left(h\left(x_i\right) - h\left(x_j\right)\right)\left(x_i - x_j\right)}{\sum_{x_i,x_j\in E} e^{\beta_0^T(x_i+x_j)}},$$

where $\beta_0$ optimises the loss function of a logistic regression on fixed $G$ and the true distribution of $D$. The asymptotic variance of the importance sampling estimator with the weights $w^*_{logreg}$ from the logistic regression with parameter $\beta^*$ is then

$$\text{Var}(I_N(h, w^*_{logreg})) = \alpha^T \text{Var}(\beta^*)\alpha = \alpha^T\left(\frac{dI_d}{N_D\lambda^2} + \frac{4(d+1)d}{(N_D\lambda\epsilon)^2}\right)\alpha.$$

### B.2 PROPOSITION 2: DEBIASING OF JI & ELKAN (2013)

As prescribed by Ji and Elkan (2013) Algorithm 1, consider importance weights

$$\overline{w}(x_i) = \exp\left(\beta^{*^T} x_i\right) = \exp\left(\widehat{\beta}^T x_i\right) \cdot \exp\left(\zeta^T x_i\right). \tag{4}$$

for privacy preserved $\widehat{\beta}$ coefficients of this logistic regression $\beta^* = \widehat{\beta} + \zeta$ with $\zeta \sim \text{Laplace}(2\sqrt{d}/(N_D\lambda\epsilon))$, a vector of length $d$. Proposition 1 proved that using $\overline{w}(\cdot)$ resulted in biased expectation estimation. However, Proposition 2 demonstrates that we can debias this in closed form.

**Proposition 2.** *Let $\overline{w}$ denote the importance weights computed by noise perturbing the regression coefficients as in Equation* (4) *(Ji and Elkan, 2013, Algorithm 1) with $\zeta \sim p_\zeta$. Define*

$$b(x_i) := 1/\mathbb{E}_{\zeta\sim p_\zeta}[\exp\left(\zeta^T x_i\right)],$$

*and adjusted importance weight*

$$\overline{w}^*(x_i) = \overline{w}(x_i) \cdot b(x_i) = \widehat{w}(x_i) \cdot \exp\left(\zeta^T x_i\right) \cdot b(x_i).$$

*The importance sampling estimator $I_N(h|\overline{w}^*)$ is unbiased and $(\epsilon, 0)$-differentially private. The variance of estimator $I_N(h|\overline{w}^*)$ has the following decomposition*

$$\text{Var}_{p_G^*}\left[I_N(h|\overline{w}^*)\right] = \frac{\overline{\sigma}^{*^2}(h)}{N} + \left(1 - \frac{1}{N}\right)\overline{c}^*(h).$$

*with*

$$\begin{aligned}
\overline{\sigma}^{*^2}(h) &= \sigma^2(h) + \mathbb{E}_{x\sim p_G}\left[h(x)^2\widehat{w}(x)^2\text{Var}_{\zeta\sim p_\zeta}\left[b(x)\exp(\zeta x)\right]\right], \\
\sigma^2(h) &= \text{Var}_{x\sim p_G}\left[h(x)\widehat{w}(x)\right], \\
\overline{c}^*(h) &= \mathbb{E}_{x,x'\sim p_G}\left[h(x)\widehat{w}(x)h(x')\widehat{w}(x')\left(\frac{b(x)b(x')}{b(x+x')} - 1\right)\right].
\end{aligned} \tag{5}$$

**Proof.** Consider $(x_1, \ldots, x_N, \zeta) \overset{i.i.d}{\sim} p_G^*$, i.e. $x_i \overset{i.i.d}{\sim} p_G$, $i = 1, \ldots, N$ and $\zeta \sim p_\zeta$ and

$$I_N(h|\overline{w}^*) = \frac{1}{N}\sum_{i=1}^{N} h(x_i)\widehat{w}(x_i)\exp\left(\zeta^T x_i\right) b(x_i),$$

then

$$\begin{aligned}
\mathbb{E}_{p_G^*}\left[I_N(h|\overline{w}^*)\right] =& \mathbb{E}_{x\sim p_G(x)}\mathbb{E}_{\zeta\sim p_\zeta}[h(x)\widehat{w}(x)\exp\left(\zeta^T x\right) b(x)] \\
=& \mathbb{E}_{x\sim p_G(x)}[h(x)\widehat{w}(x)b(x)\mathbb{E}_{\zeta\sim p_\zeta}[\exp\left(\zeta^T x\right)]] \\
=& \mathbb{E}_{x\sim p_G(x)}[h(x)\widehat{w}(x)] \\
=& \mathbb{E}_{x\sim p_D(x)}[h(x)]
\end{aligned}$$

and as a result $I_N(h|\overline{w}^*)$ is an unbiased estimator of $\mathbb{E}_{x\sim p_D(x)}[h(x)]$. The variance of estimator $I_N(h|\overline{w}^*)$ is given by

$$\begin{aligned}
\text{Var}_{p_G^*}\left[I_N(h|\overline{w}^*)\right] &= \frac{1}{N^2}\sum_{i=1}^{N}\text{Var}_{p_G^*}\left[h(x_i)\overline{w}^*(x_i)\right] + \frac{2}{N^2}\sum_{i=1}^{N}\sum_{j<i}\text{Cov}_{p_G^*}\left[h(x_i)\overline{w}^*(x_i), h(x_j)\overline{w}^*(x_j)\right] \\
&= \frac{\overline{\sigma}^{*^2}(h)}{N} + \left(1 - \frac{1}{N}\right)\overline{c}^*(h).
\end{aligned} \tag{6}$$

where the weights are dependent under $p_G^*$ because $\zeta$ **is not** sampled independently for each $x_i$, it is only sampled once. The terms making up (6) are

$$\begin{aligned}
\overline{\sigma}^{*^2}(h) =& \text{Var}_{p_G^*}\left[h(x)\widehat{w}(x)\exp(\zeta x)b(x)\right] \\
=& \mathbb{E}_{p_G^*}\left[\left(h(x)\widehat{w}(x)\exp(\zeta x)b(x)\right)^2\right] - \mathbb{E}_{p_G^*}\left[h(x)\widehat{w}(x)\exp(\zeta x)b(x)\right]^2 \\
=& \mathbb{E}_{x\sim p_G}\left[h(x)^2\widehat{w}(x)^2\mathbb{E}_{\zeta\sim p_\zeta}\left[b(x)^2\exp(\zeta x)^2\right]\right] - \mathbb{E}_{p_G}\left[h(x)\widehat{w}(x)\right]^2 \\
=& \mathbb{E}_{x\sim p_G}\left[h(x)^2\widehat{w}(x)^2\left(\text{Var}_{\zeta\sim p_\zeta}\left[b(x)\exp(\zeta x)\right] + 1\right)\right] - \mathbb{E}_{p_G}\left[h(x)\widehat{w}(x)\right]^2 \\
=& \sigma^2(h) + \mathbb{E}_{x\sim p_G}\left[h(x)^2\widehat{w}(x)^2\text{Var}_{\zeta\sim p_\zeta}\left[b(x)\exp(\zeta x)\right]\right],
\end{aligned}$$

with $\mathbb{E}_{\zeta\sim p_\zeta}\left[b(x)\exp(\zeta x)\right]=1$ by construction and $\sigma^2(h)$ defined in (5), and

$$\begin{aligned}\overline{c}^*(h) =&\text{Cov}_{p_G^*}\left[h(x)\widehat{w}(x)\exp\left(\zeta^T x\right)b(x), h(x')\widehat{w}(x')\exp\left(\zeta^T x'\right)b(x')\right]\\ =&\mathbb{E}_{x,x'\sim p_G,\zeta\sim p_\zeta}\left[h(x)\widehat{w}(x)\exp\left(\zeta^T x\right)b(x)\cdot h(x')\widehat{w}(x')\exp\left(\zeta^T x'\right)b(x')\right]\\ &-\mathbb{E}_{x,\zeta\sim p_G^*}\left[h(x)\widehat{w}(x)\exp\left(\zeta^T x\right)b(x)\right]\cdot\mathbb{E}_{x',\zeta\sim p_G^*}\left[h(x')\widehat{w}(x')\exp\left(\zeta^T x'\right)b(x')\right].\end{aligned}$$

By $\mathbb{E}_\zeta\left[\exp\left(\zeta^T x\right)b(x)\right]=1$, and $x,x'\overset{iid}{\sim}p_G$ the second term simplifies to

$$\mathbb{E}_{x\sim p_G^*}\left[h(x)\widehat{w}(x)\exp\left(\zeta^T x\right)b(x)\right]\cdot\mathbb{E}_{x'\sim p_G^*}\left[h(x')\widehat{w}(x')\exp\left(\zeta^T x'\right)b(x')\right]=\mathbb{E}_{x\sim p_G}\left[h(x)\widehat{w}(x)\right]^2.$$

The first term can be simplified as

$$\begin{aligned}&\mathbb{E}_{x,x'\sim p_G,\zeta\sim p_\zeta}\left[h(x)\widehat{w}(x)\exp\left(\zeta^T x\right)b(x)\cdot h(x')\widehat{w}(x')\exp\left(\zeta^T x'\right)b(x')\right]\\ =\;&\mathbb{E}_{x,x'\sim p_G}\left[h(x)\widehat{w}(x)h(x')\widehat{w}(x')b(x)b(x')\mathbb{E}_{\zeta\sim p_\zeta}\left[\exp\left(\zeta^T(x+x')\right)\right]\right]\\ =\;&\mathbb{E}_{x,x'\sim p_G}\left[h(x)\widehat{w}(x)h(x')\widehat{w}(x')\frac{b(x)b(x')}{b(x+x')}\right]\\ =\;&\mathbb{E}_{x,x'\sim p_G}\left[h(x)\widehat{w}(x)h(x')\widehat{w}(x')\left(\frac{b(x)b(x')}{b(x+x')}-1\right)\right]\\ &+\mathbb{E}_{x\sim p_G}\left[h(x)\widehat{w}(x)\right]\mathbb{E}_{x'\sim p_G}\left[h(x')\widehat{w}(x')\right]\quad\text{(indep.)}\\ =\;&\mathbb{E}_{x,x'\sim p_G}\left[h(x)\widehat{w}(x)h(x')\widehat{w}(x')\left(\frac{b(x)b(x')}{b(x+x')}-1\right)\right]\\ &+\mathbb{E}_{x\sim p_G}\left[h(x)\widehat{w}(x)\right]^2.\end{aligned}$$

As a result

$$\overline{c}^*(h)=\mathbb{E}_{x,x'\sim p_G}\left[h(x)\widehat{w}(x)h(x')\widehat{w}(x')\left(\frac{b(x)b(x')}{b(x+x')}-1\right)\right]$$

### B.2.1 Special Case 1: Laplace Noise

Recall that $x_i$ and $\zeta$ are $d$-dimensional vectors with $d\geq 1$. For i.i.d. $\zeta_j$, $j=1,\ldots,d$

$$\begin{aligned}\mathbb{E}\left[\exp\left(\zeta^T x_i\right)\right]&=\mathbb{E}\left[\exp\left(\sum_{j=1}^d\zeta_j x_{ij}\right)\right]\\ &=\mathbb{E}\left[\prod_{j=1}^d\exp\left(\zeta_j x_{ij}\right)\right]\\ &=\prod_{j=1}^d\mathbb{E}\left[\exp\left(\zeta_j x_{ij}\right)\right],\quad\text{(independence)}\end{aligned}$$

which is the moment generating function for random variable $\zeta_j$ evaluated at $t=x_{ij}$. Now for $\zeta_j\overset{\text{iid}}{\sim}\mathcal{L}(\mu,\rho)$

$$\begin{aligned}\prod_{j=1}^d\mathbb{E}\left[\exp\left(\zeta_j x_{ij}\right)\right]&=\prod_{j=1}^d\frac{\exp\left(\mu x_{ij}\right)}{1-\rho^2x_{ij}^2},\ \text{for}\ |x_{ij}|<1/\rho\quad\forall j\\ &=\frac{\exp\left(\mu\sum_{j=1}^d x_{ij}\right)}{\prod_{j=1}^d\left(1-\rho^2x_{ij}^2\right)},\ \text{for}\ |x_{ij}|<1/\rho\quad\forall j.\end{aligned}$$

as a result

$$b(x_i)=\frac{\prod_{j=1}^d\left(1-\rho^2x_{ij}^2\right)}{\exp\left(\mu\sum_{j=1}^d x_{ij}\right)},\ \text{with}\ |x_{ij}|<1/\rho\quad\forall j \tag{7}$$

**The variance** Of interest to the performance of such an approach are the terms

$$\begin{aligned}
\text{Var}_{\zeta\sim p_\zeta}\left[b(x_i)\exp(\zeta^T x_i)\right] &= b(x_i)^2\text{Var}_{\zeta\sim p_\zeta}\left[\exp(\zeta^T x_i)\right] \\
&= b(x_i)^2\left(\mathbb{E}_{\zeta\sim p_\zeta}\left[\exp(\zeta^T x_i)^2\right] - \mathbb{E}_{\zeta\sim p_\zeta}\left[\exp(\zeta^T x_i)\right]^2\right) \\
&= b(x_i)^2\left(\mathbb{E}_{\zeta\sim p_\zeta}\left[\exp(2\zeta^T x_i)\right] - \mathbb{E}_{\zeta\sim gp_\zeta}\left[\exp(\zeta^T x_i)\right]^2\right) \\
&= \frac{\prod_{j=1}^d\left(1-\rho^2x_{ij}^2\right)^2}{\exp\left(2\mu\sum_{j=1}^d x_{ij}\right)}\left(\frac{\exp\left(2\mu\sum_{j=1}^d x_{ij}\right)}{\prod_{j=1}^d\left(1-4b^2x_{ij}^2\right)} - \frac{\exp\left(2\mu\sum_{j=1}^d x_{ij}\right)}{\prod_{j=1}^d\left(1-\rho^2x_{ij}^2\right)^2}\right) \\
&= \prod_{j=1}^d\frac{\left(1-\rho^2x_{ij}^2\right)^2}{\left(1-4b^2x_{ij}^2\right)} - 1
\end{aligned}$$

with $|x_{ij}| < 1/2\rho \quad \forall j$, and

$$\begin{aligned}
\left(\frac{b(x)b(x')}{b(x+x')} - 1\right) &= \frac{\frac{\prod_{j=1}^d\left(1-\rho^2x_j^2\right)}{\exp\left(\mu\sum_{j=1}^d x_j\right)}\frac{\prod_{j=1}^d\left(1-\rho^2x_j'^2\right)}{\exp\left(\mu\sum_{j=1}^d x_j'\right)}}{\frac{\prod_{j=1}^d\left(1-\rho^2(x_j+x_j')^2\right)}{\exp\left(\mu\sum_{j=1}^d(x_j+x_j')\right)}} - 1, \text{ with } |x_j|, |x_j'| \text{ and } |x_j+x_j'| < 1/\rho \quad \forall j \\
&= \frac{\prod_{j=1}^d\left(1-\rho^2x_j^2\right)\left(1-\rho^2x_j'^2\right)}{\prod_{j=1}^d\left(1-\rho^2(x_j+x_j')^2\right)} - 1.
\end{aligned}$$

### B.2.2 Special Case 2: Gaussian Noise

Recall that $x_i$ and $\zeta$ are $d$-dimensional vectors with $d \geq 1$. The reciprocal of the bias correction

$$\frac{1}{b(x_i)} = \mathbb{E}_\zeta[\exp\left(\zeta^T x_i\right)],$$

is the moment generating function of random variable $\zeta^T x_i$ evaluated at $t = 1$. Now if $\zeta_j \overset{\text{iid}}{\sim} \mathcal{N}(\mu, \sigma^2)$, $j = 1, \ldots, d$, then

$$\zeta^T x_i = \sum_{j=1}^d \zeta_j x_{ij} \sim \mathcal{N}(\mu\sum_{j=1}^d x_{ij}, \sigma^2\sum_{j=1}^d x_{ij}^2)$$

and therefore

$$\mathbb{E}_\zeta\left[\exp\left(\zeta^T x_i\right)\right] = \exp\left(\mu\sum_{j=1}^d x_{ij} + \frac{1}{2}\sigma^2\sum_{j=1}^d x_{ij}^2\right).$$

**The variance** Of interest to the performance of such an approach are the terms

$$\begin{aligned}
\text{Var}_{\zeta\sim p_\zeta}\left[b(x_i)\exp(\zeta^T x_i)\right] &= b(x_i)^2\text{Var}_{\zeta\sim p_\zeta}\left[\exp(\zeta^T x_i)\right] \\
&= b(x_i)^2\left(\mathbb{E}_{\zeta\sim p_\zeta}\left[\exp(2\zeta^T x_i)\right] - \mathbb{E}_{\zeta\sim p_\zeta}\left[\exp(\zeta^T x_i)\right]^2\right) \\
&= \exp\left(-2\mu\sum_{j=1}^d x_{ij} - \sigma^2\sum_{j=1}^d x_{ij}^2\right)\left(\exp\left(2\mu\sum_{j=1}^d x_{ij} + 2\sigma^2\sum_{j=1}^d x_{ij}^2\right)\right. \\
&\quad \left. - \exp\left(2\mu\sum_{j=1}^d x_{ij} + \sigma^2\sum_{j=1}^d x_{ij}^2\right)\right) \\
&= \exp\left(\sigma^2\sum_{j=1}^d x_{ij}^2\right) - 1
\end{aligned}$$

and

$$\left(\frac{b(x)b(x')}{b(x+x')}-1\right)=\frac{\exp\left(-\mu\sum_{j=1}^{d}x_j-\frac{1}{2}\sigma^2\sum_{j=1}^{d}x_j^2\right)\exp\left(-\mu\sum_{j=1}^{d}x'_j-\frac{1}{2}\sigma^2\sum_{j=1}^{d}x_j'^2\right)}{\exp\left(-\mu\sum_{j=1}^{d}(x_j+x'_j)-\frac{1}{2}\sigma^2\sum_{j=1}^{d}(x_j+x'_j)^2\right)}-1$$
$$=\exp\left(\frac{1}{2}\sigma^2\sum_{j=1}^{d}\left\{(x_j+x'_j)^2-x_j^2-x_j'^2\right\}\right)-1$$
$$=\exp\left(\sigma^2\sum_{j=1}^{d}x_jx'_j\right)-1$$

### B.2.3 Differential Privacy

The differential privacy of the approach follows from the post-processing theorem: since the synthetic data $x_1,\ldots,x_{N_G}$ is already privatised, the corresponding weights $\bar{w}(x_1),...,\bar{w}(x_{N_G})$ are $(\epsilon,\delta)$ differentially private, and the adversary can be assumed to know which differential privacy mechanism is used (Balle and Wang, 2018), the data curator can debias the weights without any additional privacy budget.

### B.2.4 Variance Comparison of Debiasing Ji & Elkan (2013)

Ji and Elkan (2013) provide bounds for the asymptotic variance of their privatised estimator. Here, we investigate the finite sample variance of their (biased) method and compare it with the finite variance of our unbiased estimator form Proposition 2. Note that we do not consider self-normalised IW while this is an implicit assumption made by Ji and Elkan (2013).

The variance of estimator $I_N(h|\overline{w})$, where $\overline{w}$ is defined in Equation (4), is given by

$$\begin{aligned}\text{Var}_{p_G^*}\left[I_N(h|\overline{w})\right]&=\frac{1}{N^2}\sum_{i=1}^{N}\text{Var}_{p_G^*}\left[h(x_i)\overline{w}(x_i)\right]+\frac{2}{N^2}\sum_{i=1}^{N}\sum_{j<i}\text{Cov}_{p_G^*}\left[h(x_i)\overline{w}(x_i),h(x_j)\overline{w}(x_j)\right]\\&=\frac{\overline{\sigma}^2(h)}{N}+\left(1-\frac{1}{N}\right)\overline{c}(h).\end{aligned}$$

where, $x,x'\sim p_G^*$. The term $\overline{\sigma}^2(h)$ is

$$\begin{aligned}\overline{\sigma}^2(h)=&\text{Var}_{p_G^*}\left[h(x)\widehat{w}(x)\exp(\zeta^Tx)\right]\\=&\mathbb{E}_{p_G^*}\left[\left(h(x)\widehat{w}(x)\exp(\zeta^Tx)\right)^2\right]-\mathbb{E}_{p_G^*}\left[h(x)\widehat{w}(x)\exp(\zeta^Tx)\right]^2\\=&\mathbb{E}_{x\sim p_G}\left[h(x)^2\widehat{w}(x)^2\mathbb{E}_{\zeta\sim p_\zeta}\left[\exp(\zeta^Tx)^2\right]\right]-\mathbb{E}_{x\sim p_G(x)}\left[\frac{h(x)\widehat{w}(x)}{b(x)}\right]^2\\=&\mathbb{E}_{x\sim p_G}\left[h(x)^2\widehat{w}(x)^2\left(\text{Var}_{\zeta\sim p_\zeta}\left[\exp(\zeta^Tx)\right]+\frac{1}{b(x)^2}\right)\right]-\mathbb{E}_{x\sim p_G(x)}\left[\frac{h(x)\widehat{w}(x)}{b(x)}\right]^2\\=&\mathbb{E}_{x\sim p_G}\left[h(x)^2\widehat{w}(x)^2\text{Var}_{\zeta\sim p_\zeta}\left[\exp(\zeta^Tx)\right]\right]+\text{Var}_{x\sim p_G(x)}\left[\frac{h(x)\widehat{w}(x)}{b(x)}\right].\end{aligned}$$

Further, $\overline{c}(h)$ is

$$\begin{aligned}\overline{c}(h)=&\text{Cov}_{p_G^*}\left[h(x)\widehat{w}(x)\exp\left(\zeta^Tx\right),h(x')\widehat{w}(x')\exp\left(\zeta^Tx'\right)\right]\\=&\mathbb{E}_{x,x'\sim p_G^*}\left[h(x)\widehat{w}(x)\exp\left(\zeta^Tx\right)\cdot h(x')\widehat{w}(x')\exp\left(\zeta^Tx'\right)\right]\\&-\mathbb{E}_{x\sim p_G^*}\left[h(x)\widehat{w}(x)\exp\left(\zeta^Tx\right)\right]\cdot\mathbb{E}_{x'\sim p_G^*}\left[h(x')\widehat{w}(x')\exp\left(\zeta^Tx'\right)\right],\end{aligned}$$

where firstly,

$$\mathbb{E}_{x\sim p_G^*}\left[h(x)\widehat{w}(x)\exp\left(\zeta^Tx\right)\right]\cdot\mathbb{E}_{x'\sim p_G^*}\left[h(x')\widehat{w}(x')\exp\left(\zeta^Tx'\right)\right]=\mathbb{E}_{x\sim p_G(x)}\left[\frac{h(x)\widehat{w}(x)}{b(x)}\right]^2,$$

and

$$\begin{aligned}
&\mathbb{E}_{x,x'\sim p_G^*}\left[h(x)\widehat{w}(x)\exp\left(\zeta^T x\right)\cdot h(x')\widehat{w}(x')\exp\left(\zeta^T x'\right)\right]\\
=&\mathbb{E}_{x,x'\sim p_G}\left[h(x)\widehat{w}(x)h(x')\widehat{w}(x')\mathbb{E}_{\zeta\sim p_\zeta}\left[\exp\left(\zeta^T(x+x')\right)\right]\right]\\
=&\mathbb{E}_{x,x'\sim p_G}\left[h(x)\widehat{w}(x)h(x')\widehat{w}(x')\frac{1}{b(x+x')}\right]\\
=&\mathbb{E}_{x,x'\sim p_G}\left[h(x)\widehat{w}(x)h(x')\widehat{w}(x')\left(\frac{1}{b(x+x')}-\frac{1}{b(x)b(x')}\right)\right]\\
&+\mathbb{E}_{x,x'\sim p_G}\left[\frac{h(x)\widehat{w}(x)}{b(x)}\frac{h(x')\widehat{w}(x')}{b(x')}\right]\\
=&\mathbb{E}_{x,x'\sim p_G}\left[h(x)\widehat{w}(x)h(x')\widehat{w}(x')\left(\frac{1}{b(x+x')}-\frac{1}{b(x)b(x')}\right)\right]\\
&+\mathbb{E}_{x\sim p_G}\left[\frac{h(x)\widehat{w}(x)}{b(x)}\right]\mathbb{E}_{x'\sim p_G}\left[\frac{h(x')\widehat{w}(x')}{b(x')}\right]\quad\text{(indep.)}\\
=&\mathbb{E}_{x,x'\sim p_G}\left[h(x)\widehat{w}(x)h(x')\widehat{w}(x')\left(\frac{1}{b(x+x')}-\frac{1}{b(x)b(x')}\right)\right]\\
&+\mathbb{E}_{x\sim p_G}\left[\frac{h(x)\widehat{w}(x)}{b(x)}\right]^2
\end{aligned}$$

as a result

$$\begin{aligned}
\overline{c}(h)&=\mathbb{E}_{x,x'\sim p_G}\left[h(x)\widehat{w}(x)h(x')\widehat{w}(x')\left(\frac{1}{b(x+x')}-\frac{1}{b(x)b(x')}\right)\right]\\
&=\mathbb{E}_{x,x'\sim p_G}\left[\frac{h(x)\widehat{w}(x)}{b(x)}\frac{h(x')\widehat{w}(x')}{b(x')}\left(\frac{b(x)b(x')}{b(x+x')}-1\right)\right].
\end{aligned}$$

**Comparisons after debiasing:** We can compare the variance of $I_N(h|\overline{w})$ with the previously evaluated variance of $I_N(h|\overline{w}^*)$ as follows

$$\begin{aligned}
\text{Var}_{p_G^*}\left[I_N(h|\overline{w}^*)\right]=&\frac{\overline{\sigma}^{*2}(h)}{N}+\left(1-\frac{1}{N}\right)\overline{c}^*(h).\\
\text{Var}_{p_G^*}\left[I_N(h|\overline{w})\right]=&\frac{\overline{\sigma}^{2}(h)}{N}+\left(1-\frac{1}{N}\right)\overline{c}(h).
\end{aligned}$$

with

$$\begin{aligned}
\overline{\sigma}^{*2}(h)=&\mathbb{E}_{x\sim p_G}\left[h(x)^2\widehat{w}(x)^2\text{Var}_{\zeta\sim p_\zeta}\left[b(x)\exp(\zeta x)\right]\right]+\text{Var}_{x\sim p_G(x)}\left[h(x)\widehat{w}(x)\right]\\
\overline{\sigma}^{2}(h)=&\mathbb{E}_{x\sim p_G}\left[h(x)^2\widehat{w}(x)^2\text{Var}_{\zeta\sim p_\zeta}\left[\exp(\zeta^T x)\right]\right]+\text{Var}_{x\sim p_G(x)}\left[\frac{h(x)\widehat{w}(x)}{b(x)}\right]
\end{aligned}$$

and

$$\begin{aligned}
\overline{c}^*(h)&=\mathbb{E}_{x,x'\sim p_G}\left[h(x)\widehat{w}(x)h(x')\widehat{w}(x')\left(\frac{b(x)b(x')}{b(x+x')}-1\right)\right]\\
\overline{c}(h)&=\mathbb{E}_{x,x'\sim p_G}\left[\frac{h(x)\widehat{w}(x)}{b(x)}\frac{h(x')\widehat{w}(x')}{b(x')}\left(\frac{b(x)b(x')}{b(x+x')}-1\right)\right].
\end{aligned}$$

**Comparison for the introduction of Laplace noise:** From Equation (7), under $\zeta_j\sim\mathcal{L}(0,\rho)$ we have that

$$b(x_i)=\prod_{j=1}^{p}\left(1-\rho^2x_{ij}^2\right),\ \text{with}\ |x_{ij}|<1/\rho\quad\forall j.$$

The condition that $|x_{ij}| < 1/\rho$ ensures that

$$\begin{aligned} 0 \leq \left(1 - \rho^2 x_{ij}^2\right) \leq 1, \quad \forall j \\ \Rightarrow 0 \leq b(x) = \prod_{j=1}^{p} \left(1 - \rho^2 x_j^2\right) \leq 1 \end{aligned}$$

As a result,

$$\begin{aligned} \text{Var}_{\zeta \sim g}\left[b(x)\exp(\zeta^T x)\right] \leq \text{Var}_{\zeta \sim g}\left[\exp(\zeta^T x)\right], \quad \forall x \\ \text{and } h(x)\widehat{w}(x) \leq \frac{h(x)\widehat{w}(x)}{b(x)}, \quad \forall x \end{aligned}$$

which provides that

$$\begin{aligned} \overline{\sigma}^{*2}(h) \leq \overline{\sigma}^2(h) \\ \text{and } \overline{c}^*(h) \leq \overline{c}(h) \\ \Rightarrow \text{Var}_{p_G^*}\left[I_N(h|\overline{w}^*)\right] \leq \text{Var}_{p_G^*}\left[I_N(h|\overline{w})\right]. \end{aligned} \tag{8}$$

Not only does debiasing remove bias, it also makes the estimator's variance smaller.

### B.3 THEOREM 1: NOISY IMPORTANCE SAMPLING

For privacy purposes, we want to be able to noise the importance weights as in

$$\log w^*(x) = \log \widehat{w}(x) + \zeta, \quad \text{for } \zeta \sim g \text{ drawn from a noise distribution} \tag{9}$$

but we would like to still preserve the consistency properties of importance sampling estimates.

To achieve this, we expand the original target in importance sampling as follows

$$p_D^*(x, \zeta) = p_D(x)\exp(\zeta)g(\zeta)$$

where $\zeta \in \mathbb{R}$ will correspond to some additive noise on the log weights, and $g(\zeta)$ is a probability density on $\mathbb{R}$ such that by assumption

$$\int \exp(\zeta)g(\zeta)d\zeta = 1,$$

So, in particular, this implies that

$$\int p_D^*(x, \zeta)d\zeta = p_D(x).$$

Now, we can use a proposal density $p_G^*(x, \zeta) = p_G(x)g(\zeta)$ targeting $p_D^*(x, \zeta)$ and the resulting importance weight is indeed

$$w^*(x, \zeta) = \frac{p_D^*(x, \zeta)}{p_G^*(x, \zeta)} = \widehat{w}(x)\exp(\zeta),$$

i.e. the importance weight in this extended space is a noisy version of the original weight $\widehat{w}(x)$. We thus have

$$\begin{aligned} \mathbb{E}_{p_D}[h(x)] &= \mathbb{E}_{p_G}[h(x)\widehat{w}(x)] \\ &= \mathbb{E}_{p_G^*}[h(x)w^*(x, \zeta)] \\ &= \mathbb{E}_{p_G^*}[h(x)\widehat{w}(x)\exp(\zeta)]. \end{aligned}$$

It follows that for i.i.d. $(x_i, \zeta_i) \sim p_G^*$, i.e. $x_i \sim p_G$ and $\zeta_i \sim g$, then

$$I_N(h|w^*) = \frac{1}{N}\sum_{i=1}^{N} h(x_i)\widehat{w}(x_i)\exp(\zeta_i)$$

is an unbiased and consistent estimator of $\mathbb{E}_{p_D}[h(x)]$. Its variance is

$$\text{Var}\left[I_N(h|w^*)\right] = \frac{1}{N}\text{Var}_{p_D^*}\left[h(x)\widehat{w}(x)\exp(\zeta)\right] = \frac{\sigma^{*2}(h)}{N}.$$

By the variance decomposition formula, we have

$$\begin{aligned}
\sigma^{*2}(h) =& \text{Var}_{p_D^*}\left[h(x)\widehat{w}(x)\exp(\zeta)\right] \\
=& \mathbb{E}_g\left[\exp(\zeta)\right]^2 \text{Var}_{p_G}\left[h(x)\widehat{w}(x)\right] \\
&+ \text{Var}_g\left[\exp(\zeta)\right]\mathbb{E}_{p_G}\left[(h(x)\widehat{w}(x))^2\right] \quad \text{(variance decomposition formula)} \\
=& \sigma^2(h) + \text{Var}_g\left[\exp(\zeta)\right]\mathbb{E}_{p_G}[(h(x)\widehat{w}(x))^2],
\end{aligned}$$

as $\mathbb{E}_g\left[\exp(\zeta)\right] = 1$ by assumption and $\text{Var}\left[I_N(h|w)\right] = \frac{1}{N}\text{Var}_{p_G}\left[h(x)\widehat{w}(x)\right]$. The variance of our estimator is inflated as expected by the introduction of noise.

### B.4 COROLLARY 1 AND 2: DIFFERENTIAL PRIVACY OF LOG-LAPLACE NOISED IMPORTANCE WEIGHTS

Following Kozubowski and Podgórski (2003), the (symmetric) log-Laplace distribution is the distribution of random variable $x$ such that $y = \log(x)$ has a Laplace density with location parameter $\mu$ and scale $\lambda$. The density of a log-Laplace$(\mu, \lambda)$ random variable is

$$f_X(x|\mu,\lambda) = \frac{1}{2\lambda}\frac{1}{x}\exp\left(-\frac{1}{\lambda}\left|\log x - \mu\right|\right).$$

Note this is recovered from the asymmetric log-Laplace in Kozubowski and Podgórski (2003) with $\alpha = \beta = \frac{1}{\lambda}$. Kozubowski and Podgórski (2003) further provide forms for the expectation and variance of the log-Laplace distribution as

$$\begin{aligned}
\mathbb{E}\left[X\right] &= \frac{\exp(\mu)}{1-\lambda^2} \text{ for } \lambda < 1, \\
\text{Var}[X] &= \exp(2\mu)\left(\frac{1}{1-4\lambda^2} - \frac{1}{\left(1-\lambda^2\right)^2}\right) \text{ for } \lambda < \frac{1}{2}.
\end{aligned} \tag{10}$$

Next we wish to investigate the differential privacy provided by using the Laplace mechanism (Dwork et al., 2006) to noise importance weights. Adding Laplace noise to the log-weights, as in Equation (9), is equivalent to multiplying the importance weights by log-Laplace noise. In order for the importance sampling to remain unbiased, the log-Laplace noise must have expectation 1. From Equation (10) this will be the case for all $\lambda < 1$ if we set $\mu = \log\left(1-\lambda^2\right)$.

A binary logistic-regression classifier specifies class probabilities

$$\widehat{p}(y=1|x,\widehat{\beta}) = \frac{1}{1+\exp\left(-x\widehat{\beta}\right)}, \quad \widehat{p}(y=0|x,\widehat{\beta}) = \frac{\exp\left(-x\widehat{\beta}\right)}{1+\exp\left(-x\widehat{\beta}\right)}.$$

We denote by $z_{1:N_G}$ the private data sampled from the DGP, and by $x_{1:N_D}$ the synthetic data sampled from the SDGP. Let $z'_{1:N_G}$ be the neighboring data set of $z_{1:N_G}$. The importance weights estimated by such a classifier become

$$\begin{aligned}
\widehat{w}(x_i|x_{1:N_G}, z_{1:N_D}) =& \frac{\tilde{p}(y_i=1|x_i,\hat{\beta}(x_{1:N_G}, z_{1:N_D}))}{\tilde{p}(y_i=0|x_i,\hat{\beta}(x_{1:N_G}, z_{1:N_D}))}\frac{N_D}{N_G} \\
=& \frac{1}{1+\exp\left(-x_i\hat{\beta}(x_{1:N_G}, z_{1:N_D})\right)}\frac{1+\exp\left(-x_i\hat{\beta}(x_{1:N_G}, z_{1:N_D})\right)}{\exp\left(-x_i\hat{\beta}(x_{1:N_G}, z_{1:N_D})\right)}\frac{N_D}{N_G} \\
=& \exp\left(x_i\hat{\beta}(x_{1:N_G}, z_{1:N_D})\right)\frac{N_D}{N_G},
\end{aligned}$$

and as a result

$$
\begin{aligned}
&\left|\log \widehat{w}(x_i|x_{1:N_G}, z_{1:N_D}) - \log \widehat{w}(x_i|x_{1:N_G}, z'_{1:N_D})\right| \\
=&\left|x_i\hat{\beta}(x_{1:N_G}, z_{1:N_D}) + \log\frac{N_D}{N_G} - \left(x_i\hat{\beta}(x_{1:N_G}, z'_{1:N_D}) + \log\frac{N_D}{N_G}\right)\right| \\
=&\left|x_i\hat{\beta}(x_{1:N_G}, z_{1:N_D}) - x_i\hat{\beta}(x_{1:N_G}, z'_{1:N_D})\right| \\
=&\left|\sum_{j=1}^{p} x_{ij}\left(\hat{\beta}(x_{1:N_G}, z_{1:N_D})_j - \hat{\beta}(x_{1:N_G}, z'_{1:N_D})_j\right)\right| \\
\leq&|x_i|\sum_{j=1}^{d}\left|\left(\hat{\beta}(x_{1:N_G}, z_{1:N_D})_j - \hat{\beta}(x_{1:N_G}, z'_{1:N_D})_j\right)\right| \\
\leq&\frac{2\sqrt{d}}{N_D\lambda}
\end{aligned}
$$

if the features are minmax scaled using the sensitivity computed by Chaudhuri et al. (2011).

### B.5 REMARK 1: THE IMPORTANCE-WEIGHTED LIKELIHOOD AND *M*-ESTIMATION

**Remark 1.** *Minimisation of the importance weight adjusted log-likelihood, $-w(x_i)\log f(x_i|\theta)$, can be viewed as an $M$-estimator with clear relations to the standard MLE.*

Remark 1 of the paper points out the the connection between the Minimisation of the importance weight adjusted log-likelihood, $\ell_{IW}(x,\theta) := -w(x_i)\log f(x_i|\theta)$ and the standard maximum likelihood estimator which can be seen through the lens of M-estimation. We exemplify this below.

Following Van der Vaart (2000), the $M$-estimate of parameter

$$\beta_h^* := \arg\max_{\beta} \mathbb{E}_{x\sim p_D}\left[h(\beta, x)\right]$$

is given by

$$\hat{\beta}_h^{(n)} := \arg\max_{\beta} \sum_{i=1}^{n} h(\beta, x_i).$$

The estimator $\hat{\beta}_h^{(n)}$ is consistent and is asymptotically normal, i.e.

$$\sqrt{n}\left(\hat{\beta}_h^{(n)} - \beta_h^*\right) \xrightarrow{D} \mathcal{N}\left(0, \tilde{V}\left(\beta_h^*\right)\right)$$

where

$$\tilde{V}(\beta) := \left(\mathbb{E}\left[\nabla_\beta^2 h(\beta, x)\right]\right)^{-1} \cdot \mathrm{Var}\left[\nabla_\beta h(\beta, x)\right] \cdot \left(\mathbb{E}\left[\nabla_\beta^2 h(\beta, x)\right]\right)^{-1}.$$

M-estimators generalises the case of MLE under model misspecification and the variance calculation collapses to the standard inverse Fisher's information if the likelihood is correctly specified for the DGP.

The minimiser of the importance weight adjusted log-likelihood can be considered an M-estimate with the following form

$$\hat{\theta}_{IW}^{(n)} = \arg\max\left\{-\ell_{IW}(x;\theta)\right\} = \arg\max\left\{w(x)\log f(x;\theta)\right\}.$$

As a result, given $x_{1:n} \sim P_G$ the covariance of the asymptotic Gaussian distribution for $\hat{\theta}_{IW}^{(n)}$ simplifies to,

$$
\begin{aligned}
\tilde{V}_{IW}\left(\theta_{IW}^*\right) &= \left(\mathbb{E}_{p_G}\left[-\nabla_\theta^2 \ell_{IW}(x, \theta_{IW}^*)\right]\right)^{-1} \cdot \mathrm{Var}_{p_G}\left[-\nabla_\theta \ell_{IW}(x, \theta_{IW}^*)\right] \cdot \left(\mathbb{E}_{p_G}\left[-\nabla_\theta^2 \ell_{IW}(x, \theta_{IW}^*)\right]\right)^{-1} \\
&= \left(\mathbb{E}_{p_D}\left[-\nabla_\theta^2 \ell_0(x, \theta_0^*)\right]\right)^{-1} \cdot \mathrm{Var}_{p_G}\left[-\nabla_\theta \ell_{IW}(x, \theta_{IW}^*)\right] \cdot \left(\mathbb{E}_{p_D}\left[-\nabla_\theta^2 \ell_0(x, \theta_0^*)\right]\right)^{-1} \\
&= \left(\mathbb{E}_{p_D}\left[-\nabla_\theta^2 \ell_0(x, \theta_0^*)\right]\right)^{-1} \cdot \mathbb{E}_{p_G}\left[\left(-\nabla_\theta \ell_{IW}(x, \theta_{IW}^*)\right)\left(-\nabla_\theta \ell_{IW}(x, \theta_{IW}^*)\right)^T\right] \cdot \left(\mathbb{E}_{p_D}\left[-\nabla_\theta^2 \ell_0(x, \theta_0^*)\right]\right)^{-1}
\end{aligned}
$$

where $\mathrm{Var}_{p_G}\left[-\nabla_\theta \ell_{IW}(x, \theta_{IW}^*)\right] = \mathbb{E}_{p_G}\left[\left(-\nabla_\theta \ell_{IW}(x, \theta_{IW}^*)\right)\left(-\nabla_\theta \ell_{IW}(x, \theta_{IW}^*)\right)^T\right]$ because at the maximiser $\theta_{IW}^*$ $\mathbb{E}_{p_G}\left[-\nabla_\theta \ell_{IW}(x, \theta_{IW}^*)\right] = 0$

Further we can write the variance of the minimiser of the importance weight adjusted log-likelihood in terms of the variance of the standard MLE given the same number of observations $x_{1:n} \sim P_D$ as follows:

$$\frac{\tilde{V}_{IW}\left(\theta^*_{IW}\right)}{\tilde{V}_0\left(\theta^*_0\right)} = \frac{\mathbb{E}_{p_G}\left[\left(\nabla_\theta \ell_{IW}(x,\theta^*_{IW})\right)\left(\nabla_\theta \ell_{IW}(x,\theta^*_{IW})\right)^T\right]}{\mathbb{E}_{p_D}\left[\left(\nabla_\theta \ell_0(x,\theta^*_0)\right)\left(\nabla_\theta \ell_0(x,\theta^*_0)\right)^T\right]} = \frac{\mathbb{E}_{p_D}\left[w(x)\left(\nabla_\theta \ell_0(x,\theta^*_{IW})\right)\left(\nabla_\theta \ell_0(x,\theta^*_{IW})\right)^T\right]}{\mathbb{E}_{p_D}\left[\left(\nabla_\theta \ell_0(x,\theta^*_0)\right)\left(\nabla_\theta \ell_0(x,\theta^*_0)\right)^T\right]}.$$

We can then use such notions to produce an idea of the effective sample size of synthetic data.

### B.5.1 The Effective Sample Size of Synthetic Data

When constructing traditional Importance Sampling estimates it is typical to talk about the 'effective sample' size of the sample from the proposal density. The effective sample size is the number of independent samples from the true target that gives an unbiased estimator with the same variance as the importance sampling estimator using $N_G$ samples from the proposal density. When using importance weights to adjust the likelihood for Bayesian updating we are not directly seeking to estimate an expectation, but minimize an (expected) loss to produce a parameter estimate.

Analogously, in this scenario we define the effective sample size of the synthetic data as the number of samples, $N_G^{(e)}$, from true DGP $P_D$ that would provide an unbiased maximum likelihood estimate (MLE) with the same variance as the Importance-Weighted MLE (IW-MLE), i.e.

$$N_G^{(e)} := \left\{ n : \left| V\left[\hat{\theta}_{IW}^{(N_G)}\right]\right| = \left| V\left[\hat{\theta}_0^{(n)}\right]\right| \right\},$$

where the function $V$ corresponds to the asymptotic variance of that estimator, and $|\cdot|$ is a norm summary of the matrix values covariance of the estimator. Given the asymptotic analysis presented above for the importance-weighted likelihood we have that

$$N_G^{(e)} = \left( \frac{\sqrt{N_G}\left|\tilde{V}\left(\hat{\theta}_0^{(n)}\right)\right|}{\left|\tilde{V}\left(\hat{\theta}_{IW}^{(N_G)}\right)\right|} \right)^2 \tag{11}$$

where

$$\begin{aligned}\frac{\left|\tilde{V}\left(\hat{\theta}_0^{(n)}\right)\right|}{\left|\tilde{V}\left(\hat{\theta}_{IW}^{(N_G)}\right)\right|} &= \frac{\left|\mathbb{E}_{p_D}\left[\widehat{w}(x)\left(\nabla_\theta \ell_0(x,\theta^*_{IW})\right)\left(\nabla_\theta \ell_0(x,\theta^*_{IW})\right)^T\right]\right|}{\left|\mathbb{E}_{p_D}\left[\left(\nabla_\theta \ell_0(x,\theta^*_0)\right)\left(\nabla_\theta \ell_0(x,\theta^*_0)\right)^T\right]\right|} \\ &= \frac{\left|\mathbb{E}_{p_G}\left[\left(\nabla_\theta \ell_{IW}(x,\theta^*_{IW})\right)\left(\nabla_\theta \ell_{IW}(x,\theta^*_{IW})\right)^T\right]\right|}{\left|\mathbb{E}_{p_G}\left[\widehat{w}(x)\left(\nabla_\theta \ell_0(x,\theta^*_0)\right)\left(\nabla_\theta \ell_0(x,\theta^*_0)\right)^T\right]\right|}.\end{aligned}$$

We note that for multidimensional parameter vectors the $V$'s are covariance matrices and therefore we need to take a scalar summary using the norm $|\cdot|$ of these matrices in order to provide an integer effective sample size $N_G^{(e)}$. Faced with a similar problem Lyddon et al. (2018) consider the matrix trace for example.

Lastly, given a sample $x_{1:N_G} \sim P_G$ the effective sample size can be estimated by using empirical expectations

$$\frac{\left|\tilde{V}\left(\hat{\theta}_0^{(n)}\right)\right|}{\left|\tilde{V}\left(\hat{\theta}_{IW}^{(N_G)}\right)\right|} \approx \frac{\left|\frac{1}{N_G}\sum_{i=1}^{N_G}\left(\nabla_\theta \ell_{IW}(x_i,\hat{\theta}_{IW}^{(n)})\right)\left(\nabla_\theta \ell_{IW}(x_i,\hat{\theta}_{IW}^{(n)})\right)^T\right|}{\left|\frac{1}{N_G}\sum_{i=1}^{N_G}\widehat{w}(x_i)\left(\nabla_\theta \ell_0(x_i,\hat{\theta}_{IW}^{(n)})\right)\left(\nabla_\theta \ell_0(x_i,\hat{\theta}_{IW}^{(n)})\right)^T\right|}.$$

## B.6 THEOREM 1: ASYMPTOTIC POSTERIOR DISTRIBUTION OF IMPORTANCE WEIGHTED BAYESIAN UPDATING

Section 3.1 of the paper considers the importance weighted Bayesian updating as a special case of general Bayesian updating where the loss function is specifically chosen to account for the fact that inference is being done with samples from $p_G$ while trying to approximate $p_D$. We henceforth write

$$\begin{aligned}\pi_{IW}(\theta|\{x_i\}_{i\in\{1,\ldots,N_G\}}) \propto& \pi(\theta)\exp\left(-\sum_{i=1}^{N_G} -\widehat{w}(x_i)\log f(x_i|\theta)\right) \\ =& \pi(\theta)\exp\left(-\sum_{i=1}^{N_G}\ell_{IW}(x_i;\theta)\right),\end{aligned}$$

for $\ell_{IW}(x_i;\theta) := -\widehat{w}(x_i)\log f(x_i|\theta)$ and $\widehat{w}(x_i) = p_D(x_i)/p_G(x_i)$. The next theorem shows that such a posterior given observations from $p_G$ has the same asymptotic distribution as the standard Bayes posterior given samples from $p_D$ would have, and therefore we consider this posterior to be asymptotically calibrated.

We give here the formal statement of Theorem 1. Below $\xrightarrow{D}$ denotes convergence in distribution.

**Theorem 1.** *Let the regular conditions in (Chernozhukov and Hong, 2003; Lyddon et al., 2018) hold. Consider* $\hat{\theta}_{IW}^{(N)} := \arg\min_{\theta\in\Theta}\sum_{i=1}^{N}\ell_{IW}(x_i;\theta)$, $x_i \overset{\text{i.i.d.}}{\sim} p_G$ *and* $\hat{\theta}_0^{(N)} := \arg\min_{\theta\in\Theta}\sum_{i=1}^{N}\ell_0(x_i;\theta)$, $x_i \overset{\text{i.i.d.}}{\sim} p_D$ *where* $\ell_0(x;\theta) := -\log f(x;\theta)$. *Then both* $\hat{\theta}_0^{(N)}$ *and* $\hat{\theta}_{IW}^{(N)}$ *are consistent estimates of* $\theta_0^* := \arg\min_{\theta\in\Theta}\int\ell_0(x;\theta)dP_D(x)$. *Moreover there exists a non-singular matrix* $J^{-1}$ *such that we have under the importance weighted Bayesian posterior* $\pi_{IW}(\theta|x_{1:N})$

$$\sqrt{N}\left(\theta - \hat{\theta}_{IW}^{(N)}\right) \xrightarrow{D} \mathcal{N}\left(0, J^{-1}\right),$$

*almost surely w.r.t.* $x_{1:\infty}$[1] *while under the standard Bayesian posterior* $\pi(\theta|x_{1:N})$

$$\sqrt{N}\left(\theta - \hat{\theta}_0^{(N)}\right) \xrightarrow{D} \mathcal{N}\left(0, J^{-1}\right),$$

*almost surely w.r.t.* $x_{1:\infty}$.

**Proof.** Firstly, define

$$\theta_{IW}^* := \arg\min_{\theta\in\Theta}\int\ell_{IW}(x;\theta)dP_G(x), \quad J_{IW}(\theta) := \int\nabla_\theta^2\ell_{IW}(x;\theta)dP_G(x).$$

Then Chernozhukov and Hong (2003); Lyddon et al. (2018) show that under regularity conditions the following asymptotic result holds

$$\sqrt{N}\left(\theta - \hat{\theta}_{IW}^{(N)}\right) \xrightarrow{D} \mathcal{N}\left(0, J_{IW}\left(\theta_{IW}^*\right)^{-1}\right)$$

as $N\to\infty$ when $\theta$ is distributed according to the general Bayesian posterior almost surely w.r.t. $x_{1:\infty}$. Similarly, if we define

$$J_0(\theta) := \int\nabla_\theta^2\ell_0(x;\theta)dP_D(x),$$

then we have that under the standard Bayesian posterior (Chernozhukov and Hong, 2003; Kleijn et al., 2012; Lyddon et al., 2018)

$$\sqrt{N}\left(\theta - \hat{\theta}_0^{(N)}\right) \xrightarrow{D} \mathcal{N}\left(0, J_0\left(\theta_0^*\right)^{-1}\right)$$

almost surely w.r.t. $x_{1:\infty}$. Now it follows from the importance sampling identity that

$$\theta_{IW}^* = \arg\min_{\theta\in\Theta}\int\ell_{IW}(x;\theta)dP_G(x) = \arg\min_{\theta\in\Theta}\int\ell_0(x;\theta)dP_D(x) = \theta_0^*,$$
$$J_{IW}(\theta) = \int\nabla_\theta^2\ell_{IW}(x;\theta)dP_G(x) = \int\widehat{w}(x)\nabla_\theta^2\ell_0(x;\theta)dP_G(x) = \int\nabla_\theta^2\ell_0(x;\theta)dP_D(x) = J_0(\theta)$$

Moreover $\hat{\theta}_0^{(N)}$ and $\hat{\theta}_{IW}^{(N)}$ are also consistent estimates of $\theta_0^*$ under the same regularity conditions. This establishes the result.

### B.6.1 Finite Sample Importance-Weighted Bayesian posterior

To complement the asymptotic results connecting the importance weighted general Bayesian posterior given data from $p_G$ and the standard Bayesian $p_D$ we can consider the difference between these two for finite $n = m$. This is formulated in the following proposition.

**Proposition 1.** *The expected KLD beween standard Bayesian posterior* $\pi(\theta|x_{1:n})$ *and its importance weighted approximation* $\pi_{IW}(\theta|z_{1:m})$ *in expectation over the generating distributions for* $x_{1:n}\sim P_D$ *and* $z_{1:m}\sim P_G$, *for* $n = m$ *is*

$$\begin{aligned}&\mathbb{E}_{x\sim p_D}\left[\mathbb{E}_{z\sim p_G}\left[KLD(\pi(\theta|x_{1:n})||\pi_{IW}(\theta|z_{1:m})\right]\right]\\ =&n\mathbb{E}_{x\sim p_D}\left[\mathbb{E}_{\theta\sim\pi(\cdot|x_{1:n})}\left[\left(\log f(x;\theta) - \mathbb{E}_{x'\sim p_D}\left[\log f(x';\theta)\right]\right)\right]\right]\end{aligned}$$

[1] $\pi_{IW}(\theta|x_{1:N})$ and $\pi(\theta|x_{1:N})$ are here interpreted as random probability measures, and functions of the random observations $x_{1:N}$.

**Proof.** We have

$$\begin{aligned}
&\mathbb{E}_{x\sim p_D}\left[\mathbb{E}_{z\sim p_G}\left[KLD(\pi(\theta|x_{1:n})||\pi_{IW}(\theta|z_{1:m})\right]\right]\\
=&\mathbb{E}_{x\sim p_D}\left[\mathbb{E}_{z\sim p_G}\left[\int \pi(\theta|x_{1:n})\log\frac{\pi(\theta|x_{1:n})}{\pi_{IW}(\theta|z_{1:m})}d\theta\right]\right]\\
=&\mathbb{E}_{x\sim p_D}\left[\mathbb{E}_{z\sim p_G}\left[\mathbb{E}_{\pi(\theta|x_{1:n})}\left[\sum_{i=1}^{n}\log f(x_i;\theta)-\sum_{j=1}^{m}\widehat{w}(z_i)\log f(z_i;\theta)\right]\right]\right].
\end{aligned}$$

Now by Fubini we can reorder these integrals assuming that they all exist

$$\begin{aligned}
=&\mathbb{E}_{x\sim p_D}\left[\mathbb{E}_{\theta\sim\pi(\cdot|x_{1:n})}\left[\left(\sum_{i=1}^{n}\log f(x_i;\theta)-\sum_{j=1}^{m}\mathbb{E}_{z\sim p_G}\left[\widehat{w}(z_i)\log f(z_i;\theta)\right]\right)\right]\right]\\
=&\mathbb{E}_{x\sim p_D}\left[\mathbb{E}_{\theta\sim\pi(\cdot|x_{1:n})}\left[\left(\sum_{i=1}^{n}\log f(x_i;\theta)-m\mathbb{E}_{x'\sim p_D}\left[\log f(x';\theta)\right]\right)\right]\right].
\end{aligned}$$

Now assuming $n=m$, we have

$$\begin{aligned}
=&\mathbb{E}_{x\sim p_D}\left[\mathbb{E}_{\theta\sim\pi(\cdot|x_{1:n})}\left[\sum_{i=1}^{n}\left(\log f(x_i;\theta)-\mathbb{E}_{x'\sim p_D}\left[\log f(x';\theta)\right]\right)\right]\right]\\
=&n\mathbb{E}_{x\sim p_D}\left[\mathbb{E}_{\theta\sim\pi(\cdot|x_{1:n})}\left[\left(\log f(x;\theta)-\mathbb{E}_{x'\sim p_D}\left[\log f(x';\theta)\right]\right)\right]\right].
\end{aligned}$$

# C EXPERIMENTS

## C.1 EXPERIMENTAL DETAILS

Please refer to Table 1 for an overview of the data sets used. We considered a random $80/20$ train test split for all data sets except for MNIST for which the default split was used.

| Data | # training observations | # features | prediction problem |
|---|---|---|---|
| Iris | 150 | 4 | 3-class classification |
| tgfb | 262 | 7 | regression |
| Boston | 506 | 10 | regression |
| Breast | 569 | 30 | binary classification |
| Banknote | 1372 | 4 | binary classification |
| MNIST | 60000 | 784 | 10-class classification |

Table 1: Characteristics of the analysed data sets

We obtained the code for PrivBayes from `https://github.com/DataResponsibly/DataSynthesizer`, and the code for DPCGAN from `https://github.com/ricardocarvalhods/dpcgan`. This code was used and changed to write the code for DPGAN. For the logistic regression alternatives we use an adaption of the `sklearn` implementation. DPGAN was trained on labelled data by concatenating the features with the one hot encoding of the labels. Our implementation will be made available online. We train different downstream tasks on the synthetic data and test them on test data to ensure their utility for the setting of supervised learning. The downstream algorithms were trained using `sklearn` with default parameters.

Hyperparameter tuning is a non-private operation as it queries private data to evaluate the model at validation time. To ensure that we do not undermine the performance of the baselines we tuned them for $\epsilon = 1.$, and chose default parameters for our method. PrivBayes is trained in correlated attribute mode, and with optimal bandwidth computation. For the GAN alternatives, we tuned the norm clip (1.0, 0.5), the batch size (32, 64), and number of epochs (50, 100) with grid search on a validation set (10% split of training). The noise multiplier was chosen such that the desired privacy budget was reached. The models were then retrained on the full training data set. Note that these hyperparameters are chosen smaller than in a non-private setting as the noise to be added would otherwise explode. The optimal hyperparameters can be found in the GitHub repository. Further we chose learning rate of the discriminator and generator as 0.15, and the

number of hidden dimensions as $d$ following Jordon et al. (2019). For the MNIST experiment, we chose to use the hyperparameters found by Torkzadehmahani et al. (2019). The regularisation parameter of the logistic regression for weight estimation was chosen from $0.1, 1, 2$.

The MLP for likelihood ratio estimation was computed based on the `tensorflow` and `tensorflow_privacy` package. To ensure the privacy of the MLP, we started with a configuration of one epoch, a batch size of 1, an L2 norm clip of 1, a noise multiplier of 5.2, 20 microbatches and a learning rate of 0.1. We computed the $\epsilon$ using built-in functions and increased/decreased the noise multiplier and the number of epochs until the desired privacy level was reached. We chose $N_S = N_D$ unless otherwise mentioned. To compute the output-noised weights we computed the largest $N_S$ such that the scale restriction was satisfied and conducted the downstream analysis on this smaller dataset.

### C.2 COMPUTATIONAL TIME OF IMPORTANCE WEIGHT ESTIMATION

Please refer to Table 2 for an overview of the additional time needed to compute the importance weights. All experimental results were computed by training on a single Tesla V100 GPU. We observe that the estimation of the importance weights comes with negligible computational overhead.

| weighting | Iris | Banknote | Housing | Breast | MNIST |
|---|---|---|---|---|---|
| BetaNoised | $0.0064_{\pm 0.0002}$ | $0.0084_{\pm 0.0002}$ | $0.0133_{\pm 0.0011}$ | $0.0824_{\pm 0.0206}$ | $51.5605_{\pm 9.0042}$ |
| BetaDebiased | $0.0237_{\pm 0.0125}$ | $0.0112_{\pm 0.0003}$ | $0.0742_{\pm 0.0083}$ | $0.1856_{\pm 0.0858}$ | $59.0723_{\pm 10.5120}$ |
| DP-MLP | $0.8338_{\pm 0.0964}$ | $5.4649_{\pm 0.0654}$ | $1.7303_{\pm 0.1104}$ | $2.9363_{\pm 0.1208}$ | $87.2693_{\pm 4.7303}$ |
| Discriminator | $0.0000_{\pm 0.0000}$ | $0.0000_{\pm 0.0000}$ | $0.0000_{\pm 0.0000}$ | $0.0000_{\pm 0.0000}$ | $0.0000_{\pm 0.0001}$ |
| LogReg | $0.0071_{\pm 0.0004}$ | $0.0099_{\pm 0.0003}$ | $0.0143_{\pm 0.0012}$ | $0.0910_{\pm 0.0210}$ | $52.0331_{\pm 9.1285}$ |
| MLP | $0.7741_{\pm 0.1436}$ | $1.5895_{\pm 0.0261}$ | $1.7491_{\pm 0.1414}$ | $1.4480_{\pm 0.1441}$ | $30.1968_{\pm 6.3155}$ |

Table 2: Additional computational time in seconds needed for the computation of importance weights averaged over 10 seeds and SDGP for $\epsilon = 1$.

### C.3 CHOICE OF PRIVACY SPLIT

In Figure 1, we plot the change in evaluation metrics for different values of privacy budget splits. We notice that the impact of the split parameter decreases the larger $\epsilon$ is. Similarly, the variability in the metrics for different $\delta$ splits decreases, the larger $\epsilon_{IW}$ is, where $\epsilon_{IW}$ denotes the privacy budget dedicated to the importance weight estimation. While a larger $\delta$ split of 30-50% seems beneficial for DP-MLP, the fraction of $\epsilon$ dedicated to the importance weighting model should be chosen relatively small, i.e. 10%. Note that we chose these default values based on their performance on the Adult, Credit and Spam data set. Tuning them to the underlying data and task characteristics will be able to improve their results. As hyperparameter tuning is an unsolved problem in DP, we leave the procedure for choosing the optimal privacy split per data set for future work. We note that an additional intricacy appears in DP because of the noise injection which increases the variability of the model's performances.

### C.4 MSE OF IMPORTANCE WEIGHT ESTIMATION

For each of our experiments, we compute the mean squared error between the privatised parameters of the logistic regression for importance weight estimation and the parameters of an unperturbed logistic regression trained on the private data. Please refer to Table 3 for the results. We observe that debiasing almost always decreases the MSE in the low-privacy regimes. For large privacy budgets, the scale of the perturbations can be negligible for low-dimensional data sets which is why both approaches perform similarly on Iris and Banknote, but debiasing still helps with larger data sets such as Breast.

### C.5 BAYESIAN UPDATING EXPERIMENTAL DETAILS

In addition to the logistic regression ROC-AUC score distributions presented in the main body of the paper, we applied importance weighted posteriors to updating and learning the parameters of linear regression and multinomial logistic regression models applied to the TGFB and Iris datasets respectively, see Figures 2a and 2b. It can be seen that in the case of linear regression, the DP-MLP and MLP IW methods are again very effective, with the performance improving across all SDGPs. Other methods again tend to reduce variance in the results whilst not damaging performance and so can be seen to be effective in at least ensuring greater robustness and consistency when learning under synthetic data. In the case of the Iris data, we calculated 1 vs all ROC-AUC scores for each class separately, then averaged these per-class ROC-AUCs to get a single multi-class average ROC-AUC. Again, MLP and DP-MLP are stand-out in their

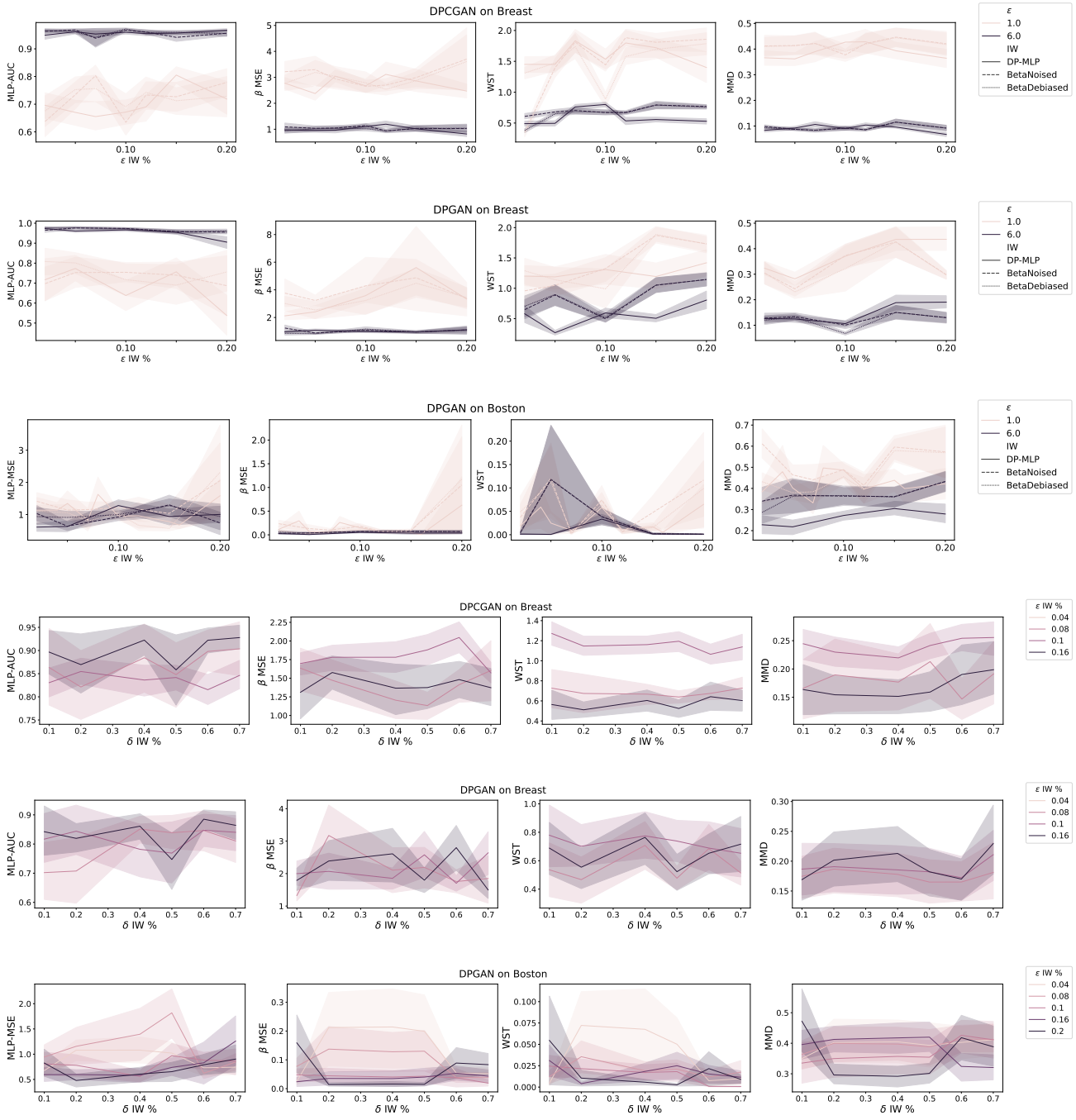



Figure 1: Multiple metrics measured across a range of privacy splits on Breast and Boston averaged over 10 seeds, and displayed with standard errors. The maximum mean discrepancy (MMD) was included as a measure of divergence between the weighted SDGP and the test distribution.

| SDGP | data | $\epsilon = 1$ | | $\epsilon = 6$ | |
|---|---|---|---|---|---|
| | | BetaNoised | BetaDebiased | BetaNoised | BetaDebiased |
| CGAN | Breast | $1.4833_{\pm 0.9603}$ | $\mathbf{0.0775_{\pm 0.0197}}$ | $0.0024_{\pm 0.0006}$ | $\mathbf{0.0020_{\pm 0.0004}}$ |
| | Banknote | $0.0420_{\pm 0.0211}$ | $\mathbf{0.0413_{\pm 0.0196}}$ | $\mathbf{0.0014_{\pm 0.0007}}$ | $\mathbf{0.0014_{\pm 0.0007}}$ |
| | Iris | $8.7522_{\pm 4.9893}$ | $\mathbf{3.4687_{\pm 1.3044}}$ | $\mathbf{0.1160_{\pm 0.0240}}$ | $0.1290_{\pm 0.0311}$ |
| GAN | Housing | $8.2081_{\pm 7.7702}$ | $\mathbf{1.4406_{\pm 0.8314}}$ | $3.7916_{\pm 3.3246}$ | $\mathbf{1.5479_{\pm 1.0430}}$ |
| DPCGAN | Breast | $0.0582_{\pm 0.0165}$ | $\mathbf{0.0445_{\pm 0.0162}}$ | $0.0015_{\pm 0.0003}$ | $\mathbf{0.0014_{\pm 0.0003}}$ |
| | Banknote | $0.0420_{\pm 0.0211}$ | $\mathbf{0.0413_{\pm 0.0196}}$ | $0.0022_{\pm 0.0013}$ | $\mathbf{0.0021_{\pm 0.0012}}$ |
| | Iris | $\mathbf{0.7834_{\pm 0.2341}}$ | $1.2300_{\pm 0.7050}$ | $\mathbf{0.2502_{\pm 0.1627}}$ | $0.2806_{\pm 0.1760}$ |
| DPGAN | Breast | $6.0487_{\pm 3.7927}$ | $\mathbf{3.7629_{\pm 2.2881}}$ | $0.0251_{\pm 0.0245}$ | $\mathbf{0.0238_{\pm 0.0234}}$ |
| | Banknote | $\mathbf{0.0582_{\pm 0.0353}}$ | $0.0610_{\pm 0.0397}$ | $0.0062_{\pm 0.0057}$ | $\mathbf{0.0061_{\pm 0.0056}}$ |
| | Iris | $2.6486_{\pm 1.3518}$ | $\mathbf{1.3698_{\pm 1.1554}}$ | $\mathbf{0.0741_{\pm 0.0228}}$ | $0.0864_{\pm 0.0274}$ |
| | Housing | $5.9175_{\pm 2.8546}$ | $\mathbf{0.8398_{\pm 0.6328}}$ | $\mathbf{1.9044_{\pm 1.1426}}$ | $2.1111_{\pm 1.3450}$ |

Table 3: Mean squared error of the privatised log importance weights $\log \overline{w}$ resp. $\log \overline{w}^*$ averaged over 10 runs with standard errors reported in brackets for $(\epsilon = 1, \delta = 10^{-5})$ and $(\epsilon = 6, \delta = 10^{-5})$ where $\epsilon_{IW} = 0.1\epsilon$.

performance, significantly improving the performance measured by this metric, especially under synthetic data from the CGAN, DPCGAN and PrivBayes generators. Similar gains can be seen across the majority of the methods for the DPCGAN, especially at the higher $\epsilon = 6$.

All of these models were implemented in the `Turing.jl` `PPL` Ge et al. (2018). We then ran an experiment for each model and dataset on a defined grid across all seeds, synthetic generators and $\epsilon$ values. For each combination, we generated 10,000 samples across 4 chains (not counting 1,000 discarded warm-up samples per chain) for each of the importance weighting methods, as well as once for a model fit on the synthetic data with its standard non-weighted posterior, and once for the real data. We used Turing's implementation of the NUTS sampling algorithm with a target acceptance ratio of 0.65 for sampling the linear regression models' parameters, and for the logistic and multinomial logistic regression models we used HMC with a leapfrog step size of 0.05 and 10 leapfrog steps per iteration. The logistic and multinomial logistic regression models' coefficients (including intercepts) were given centred Normal priors with $\sigma = 1$. The linear regression models' coefficient priors were given the same centred Normal priors with $\sigma = 1$; its variance was given a non-informative prior via a truncated Normal distribution ensuring positivity with $\sigma = 10$.

We then took all 10,000 samples and calculated our evaluation metrics on the test set for each sample, storing all of these. We then present the distributions of metric scores that arise in the included box-plot figures.

### C.6 ILLUSTRATIVE EXAMPLE OF THE IMPLICATIONS OF BIAS MITIGATION

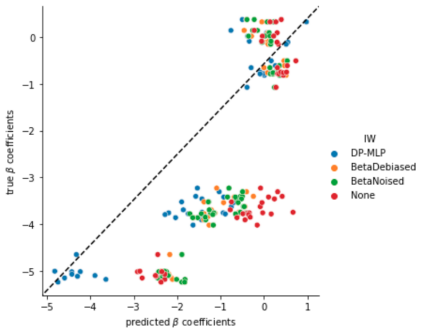
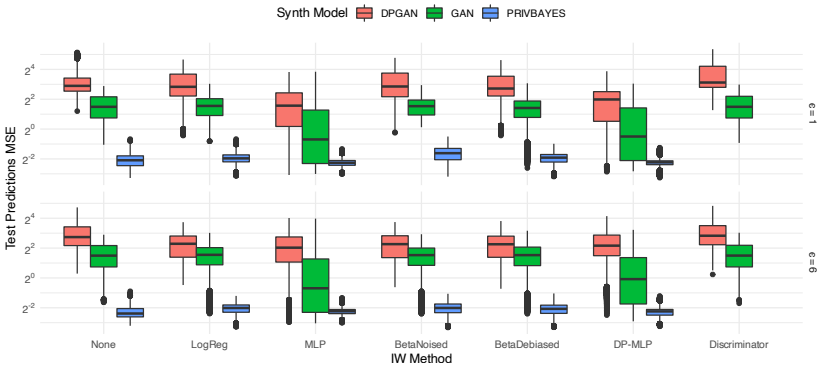



Figure 3: Illustrative example of debiasing with IW on PrivBayes synthesised Banknote data.

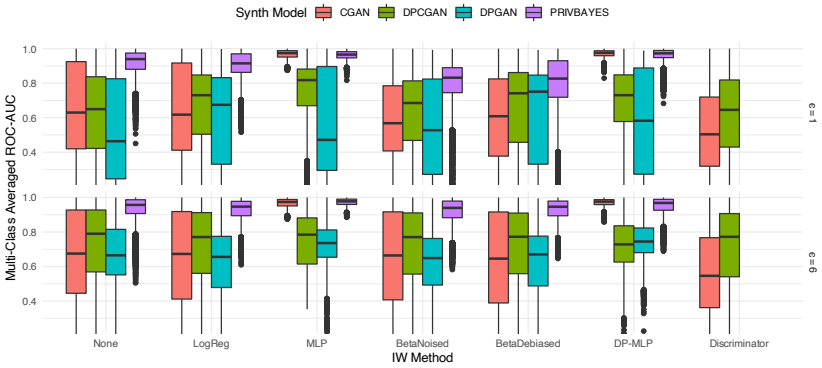
In Figure 3, we visualise the benefit of debiasing: We fitted a logistic regression as a downstream classifier on the private data to get the *true* $\beta$ *coefficients*. The *predicted* $\beta$ *coefficients* are estimated by training the logistic classifier on the importance weighted synthetic data. Each dot in the figure plots one dimension of the predicted $\beta$ coefficients against its true counterpart for one training run (out of ten). An optimal classifier would reconstruct the true coefficients. In this case all lines would be on the diagonal. An *unbiased* estimator would on average reconstruct the true coefficients: For each true $\beta$ coefficient, the predicted coefficients would be centred around the true value. We observe that coefficients learned without importance weighting exhibit the largest distance to the diagonal line, while the importance weighting alternatives push the dots closer to the diagonal line. Our method, DP-MLP, is particularly successful in decreasing the bias in the $\beta$ coefficients.

### C.7 COMPLETE UCI RESULTS

The complete experimental results on the UCI data sets can be found in Tables 4 to 7. Each table displays the performance of the different weight estimators for private and non-private synthetic data generative models for $\epsilon \in \{1, 6\}$, $\epsilon_{IW} = 0.1\epsilon$ and $\delta_{IW} = 0.3\delta$. We observe that importance weighting brings significant gains especially in low privacy regimes. For high privacy regimes this effect is reduced as the SDGP gets closer to the DGP.




(a) Test set prediction MSE distributions calculated via chains of parameters sampled from a Bayesian linear regression model fit on synthesised TGFB data across 10 seeds.




(b) Multi-class averaged ROC-AUC distributions calculated via chains of parameters sampled from a Bayesian multinomial logistic regression model fit on synthesised Iris data across 10 seeds.

| | | SDGP | CGAN | DPCGAN | DPGAN | PrivBayes |
|---|---|---|---|---|---|---|
| $\epsilon = 1$ | MLP-ROC-AUC ↑ | None | $0.4619_{\pm 0.1010}$ | $0.4717_{\pm 0.1103}$ | $0.5357_{\pm 0.0752}$ | $0.5243_{\pm 0.1299}$ |
| | | BetaNoised | $0.5824_{\pm 0.0931}$ | $0.5841_{\pm 0.0831}$ | $0.5487_{\pm 0.0803}$ | $\mathbf{0.6651_{\pm 0.0884}}$ |
| | | BetaDebiased | $0.5669_{\pm 0.1237}$ | $0.5913_{\pm 0.1136}$ | $0.5998_{\pm 0.1141}$ | $0.5005_{\pm 0.0793}$ |
| | | DP-MLP | $\mathbf{0.6299_{\pm 0.0984}}$ | $0.5725_{\pm 0.0859}$ | $0.5448_{\pm 0.0912}$ | $0.6143_{\pm 0.0374}$ |
| | | Discriminator | $0.5809_{\pm 0.0840}$ | $\mathbf{0.5995_{\pm 0.0982}}$ | $\mathbf{0.6475_{\pm 0.0701}}$ | - |
| | | LogReg | $0.4980_{\pm 0.0780}$ | $0.4908_{\pm 0.0950}$ | $0.4806_{\pm 0.0806}$ | $0.6245_{\pm 0.1235}$ |
| | | MLP | $0.7230_{\pm 0.0791}$ | $0.6273_{\pm 0.0988}$ | $0.5770_{\pm 0.1199}$ | $0.6778_{\pm 0.0923}$ |
| | $\beta$ MSE ↓ | None | $1.3594_{\pm 0.3789}$ | $1.0460_{\pm 0.2457}$ | $\mathbf{3.8955_{\pm 0.9764}}$ | $0.3511_{\pm 0.0753}$ |
| | | BetaNoised | $1.4944_{\pm 0.2321}$ | $1.1133_{\pm 0.1911}$ | $4.1565_{\pm 1.0469}$ | $0.4739_{\pm 0.0469}$ |
| | | BetaDebiased | $1.3682_{\pm 0.3080}$ | $1.3347_{\pm 0.2830}$ | $4.1694_{\pm 0.9246}$ | $0.8147_{\pm 0.1690}$ |
| | | DP-MLP | $\mathbf{0.6109_{\pm 0.0481}}$ | $1.0663_{\pm 0.1411}$ | $4.4986_{\pm 1.2881}$ | $\mathbf{0.1962_{\pm 0.0413}}$ |
| | | Discriminator | $1.0454_{\pm 0.3012}$ | $\mathbf{0.9404_{\pm 0.1024}}$ | $3.9049_{\pm 0.6010}$ | - |
| | | LogReg | $1.3345_{\pm 0.2725}$ | $0.9557_{\pm 0.1356}$ | $4.1971_{\pm 1.1035}$ | $0.3659_{\pm 0.0660}$ |
| | | MLP | $0.6091_{\pm 0.0546}$ | $0.8316_{\pm 0.1630}$ | $4.5109_{\pm 1.3057}$ | $0.1551_{\pm 0.0162}$ |
| | WST ↓ | None | $0.7226_{\pm 0.0543}$ | $0.7448_{\pm 0.0423}$ | $0.7919_{\pm 0.0458}$ | $0.5055_{\pm 0.0111}$ |
| | | BetaNoised | $0.2771_{\pm 0.0490}$ | $0.1014_{\pm 0.0519}$ | $0.1893_{\pm 0.0266}$ | $0.1412_{\pm 0.0493}$ |
| | | BetaDebiased | $\mathbf{0.2340_{\pm 0.0210}}$ | $\mathbf{0.0989_{\pm 0.0062}}$ | $0.1457_{\pm 0.0143}$ | $\mathbf{0.1059_{\pm 0.0032}}$ |
| | | DP-MLP | $0.3960_{\pm 0.0561}$ | $0.2376_{\pm 0.0196}$ | $0.2613_{\pm 0.0627}$ | $0.3451_{\pm 0.0253}$ |
| | | Discriminator | $0.2698_{\pm 0.0383}$ | $0.1696_{\pm 0.0371}$ | $\mathbf{0.1003_{\pm 0.0003}}$ | - |
| | | LogReg | $0.2341_{\pm 0.0687}$ | $0.1444_{\pm 0.0406}$ | $0.1611_{\pm 0.0178}$ | $0.3531_{\pm 0.0357}$ |
| | | MLP | $0.2677_{\pm 0.0693}$ | $0.0967_{\pm 0.0287}$ | $0.0752_{\pm 0.0261}$ | $0.1396_{\pm 0.0139}$ |
| $\epsilon = 6$ | MLP-ROC-AUC ↑ | None | $0.4662_{\pm 0.1039}$ | $0.5202_{\pm 0.0928}$ | $0.5252_{\pm 0.0844}$ | $0.4875_{\pm 0.1139}$ |
| | | BetaNoised | $0.5842_{\pm 0.0900}$ | $0.5531_{\pm 0.1093}$ | $0.5603_{\pm 0.0980}$ | $\mathbf{0.6218_{\pm 0.1304}}$ |
| | | BetaDebiased | $\mathbf{0.6029_{\pm 0.1100}}$ | $\mathbf{0.6992_{\pm 0.0801}}$ | $\mathbf{0.6445_{\pm 0.0906}}$ | $0.5388_{\pm 0.1258}$ |
| | | DP-MLP | $0.6007_{\pm 0.1060}$ | $0.6054_{\pm 0.0951}$ | $0.5181_{\pm 0.0957}$ | $0.5639_{\pm 0.0483}$ |
| | | Discriminator | $0.5894_{\pm 0.0829}$ | $0.5806_{\pm 0.1014}$ | $0.5909_{\pm 0.0903}$ | - |
| | | LogReg | $0.5073_{\pm 0.0852}$ | $0.5353_{\pm 0.0793}$ | $0.4934_{\pm 0.1051}$ | $0.7088_{\pm 0.0843}$ |
| | | MLP | $0.7206_{\pm 0.0774}$ | $0.7118_{\pm 0.0774}$ | $0.5923_{\pm 0.1130}$ | $0.6734_{\pm 0.0881}$ |
| | $\beta$ MSE ↓ | None | $1.4111_{\pm 0.3882}$ | $1.0262_{\pm 0.1866}$ | $\mathbf{2.0710_{\pm 0.3284}}$ | $0.2650_{\pm 0.0610}$ |
| | | BetaNoised | $1.2894_{\pm 0.2726}$ | $0.9507_{\pm 0.3017}$ | $2.8284_{\pm 1.0195}$ | $0.3338_{\pm 0.0701}$ |
| | | BetaDebiased | $1.2679_{\pm 0.2854}$ | $0.9511_{\pm 0.3113}$ | $2.8256_{\pm 1.0359}$ | $0.3492_{\pm 0.0719}$ |
| | | DP-MLP | $\mathbf{0.5928_{\pm 0.0682}}$ | $\mathbf{0.7773_{\pm 0.2286}}$ | $4.1112_{\pm 1.1372}$ | $\mathbf{0.2559_{\pm 0.0527}}$ |
| | | Discriminator | $1.0434_{\pm 0.3014}$ | $0.9449_{\pm 0.2838}$ | $2.1203_{\pm 0.5427}$ | - |
| | | LogReg | $1.2606_{\pm 0.2771}$ | $0.9604_{\pm 0.3155}$ | $2.8409_{\pm 1.0311}$ | $0.3603_{\pm 0.0806}$ |
| | | MLP | $0.6174_{\pm 0.0523}$ | $0.5102_{\pm 0.1630}$ | $3.9403_{\pm 1.1462}$ | $0.1283_{\pm 0.0252}$ |
| | WST ↓ | None | $0.7399_{\pm 0.0445}$ | $0.6598_{\pm 0.1077}$ | $0.6770_{\pm 0.0379}$ | $0.4255_{\pm 0.0208}$ |
| | | BetaNoised | $0.2703_{\pm 0.0492}$ | $0.3032_{\pm 0.0697}$ | $0.2622_{\pm 0.0229}$ | $0.4467_{\pm 0.0200}$ |
| | | BetaDebiased | $0.3035_{\pm 0.0601}$ | $0.3171_{\pm 0.0746}$ | $0.2770_{\pm 0.0332}$ | $\mathbf{0.3383_{\pm 0.0070}}$ |
| | | DP-MLP | $0.4507_{\pm 0.0722}$ | $0.5374_{\pm 0.0654}$ | $0.4445_{\pm 0.0635}$ | $0.4850_{\pm 0.0160}$ |
| | | Discriminator | $\mathbf{0.2134_{\pm 0.0419}}$ | $\mathbf{0.2168_{\pm 0.0032}}$ | $\mathbf{0.2178_{\pm 0.0037}}$ | - |
| | | LogReg | $0.3090_{\pm 0.0612}$ | $0.2836_{\pm 0.0742}$ | $0.2601_{\pm 0.0262}$ | $0.4591_{\pm 0.0121}$ |
| | | MLP | $0.2064_{\pm 0.0819}$ | $0.1343_{\pm 0.0299}$ | $0.2711_{\pm 0.0235}$ | $0.1981_{\pm 0.0192}$ |

Table 4: Results on Iris averaged over 10 seeds.

| | | SDGP | CGAN | DPCGAN | DPGAN | PrivBayes |
|---|---|---|---|---|---|---|
| $\epsilon = 1$ | MLP-ROC-AUC ↑ | None | $0.7408_{\pm 0.0522}$ | $0.8546_{\pm 0.0213}$ | $0.6863_{\pm 0.0436}$ | $0.7630_{\pm 0.0495}$ |
| | | BetaNoised | $0.7469_{\pm 0.0522}$ | $0.8495_{\pm 0.0274}$ | $0.6063_{\pm 0.0510}$ | $0.8943_{\pm 0.0173}$ |
| | | BetaDebiased | $\mathbf{0.7864_{\pm 0.0888}}$ | $\mathbf{0.8729_{\pm 0.0310}}$ | $0.5868_{\pm 0.1005}$ | $0.7632_{\pm 0.0517}$ |
| | | DP-MLP | $0.7313_{\pm 0.0613}$ | $0.7697_{\pm 0.0419}$ | $0.5657_{\pm 0.0570}$ | $\mathbf{0.8953_{\pm 0.0299}}$ |
| | | Discriminator | $0.7511_{\pm 0.0523}$ | $0.8695_{\pm 0.0167}$ | $\mathbf{0.7114_{\pm 0.0424}}$ | - |
| | | LogReg | $0.7986_{\pm 0.0391}$ | $0.8172_{\pm 0.0327}$ | $0.6034_{\pm 0.0534}$ | $0.9102_{\pm 0.0129}$ |
| | | MLP | $0.7253_{\pm 0.0521}$ | $0.8291_{\pm 0.0333}$ | $0.5974_{\pm 0.0627}$ | $0.8594_{\pm 0.0231}$ |
| | $\beta$ MSE ↓ | None | $15.3278_{\pm 2.5238}$ | $11.0215_{\pm 1.8377}$ | $39.3243_{\pm 3.7708}$ | $8.1724_{\pm 0.3987}$ |
| | | BetaNoised | $11.7636_{\pm 2.1960}$ | $8.4298_{\pm 1.0383}$ | $35.2862_{\pm 4.0365}$ | $5.7001_{\pm 0.1885}$ |
| | | BetaDebiased | $\mathbf{8.4946_{\pm 1.7858}}$ | $\mathbf{8.3508_{\pm 2.3127}}$ | $32.9909_{\pm 5.9024}$ | $6.6862_{\pm 0.1458}$ |
| | | DP-MLP | $14.6644_{\pm 2.9599}$ | $17.1597_{\pm 2.5448}$ | $36.4618_{\pm 4.1011}$ | $\mathbf{3.5519_{\pm 0.2895}}$ |
| | | Discriminator | $14.9537_{\pm 2.5553}$ | $12.5471_{\pm 2.3124}$ | $\mathbf{30.9282_{\pm 5.4283}}$ | - |
| | | LogReg | $11.7777_{\pm 2.2000}$ | $8.4760_{\pm 1.0406}$ | $35.2964_{\pm 4.0396}$ | $5.6751_{\pm 0.1785}$ |
| | | MLP | $15.4584_{\pm 3.0826}$ | $17.9390_{\pm 2.4926}$ | $35.5211_{\pm 4.2147}$ | $2.6286_{\pm 0.3761}$ |
| | WST ↓ | None | $0.6702_{\pm 0.0282}$ | $0.4746_{\pm 0.0214}$ | $0.7442_{\pm 0.0333}$ | $0.3237_{\pm 0.0162}$ |
| | | BetaNoised | $0.3106_{\pm 0.0475}$ | $0.2509_{\pm 0.0436}$ | $0.4355_{\pm 0.0456}$ | $0.2318_{\pm 0.0035}$ |
| | | BetaDebiased | $0.3837_{\pm 0.0990}$ | $0.4015_{\pm 0.0766}$ | $0.4618_{\pm 0.0832}$ | $0.2369_{\pm 0.0061}$ |
| | | DP-MLP | $\mathbf{0.1418_{\pm 0.0283}}$ | $\mathbf{0.2035_{\pm 0.0427}}$ | $0.4298_{\pm 0.0433}$ | $\mathbf{0.0456_{\pm 0.0061}}$ |
| | | Discriminator | $0.6366_{\pm 0.0273}$ | $0.3382_{\pm 0.0399}$ | $\mathbf{0.1087_{\pm 0.0415}}$ | - |
| | | LogReg | $0.3092_{\pm 0.0470}$ | $0.2508_{\pm 0.0432}$ | $0.4348_{\pm 0.0460}$ | $0.2348_{\pm 0.0034}$ |
| | | MLP | $0.0494_{\pm 0.0141}$ | $0.0913_{\pm 0.0259}$ | $0.3860_{\pm 0.0452}$ | $0.0021_{\pm 0.0004}$ |
| $\epsilon = 6$ | MLP-ROC-AUC ↑ | None | $0.7212_{\pm 0.0491}$ | $0.8958_{\pm 0.0179}$ | $\mathbf{0.8323_{\pm 0.0301}}$ | $0.8357_{\pm 0.0354}$ |
| | | BetaNoised | $\mathbf{0.7811_{\pm 0.0423}}$ | $0.8771_{\pm 0.0227}$ | $0.8216_{\pm 0.0320}$ | $0.8588_{\pm 0.0295}$ |
| | | BetaDebiased | $0.6951_{\pm 0.0958}$ | $\mathbf{0.8992_{\pm 0.0334}}$ | $0.7061_{\pm 0.1083}$ | $0.8136_{\pm 0.0648}$ |
| | | DP-MLP | $0.6879_{\pm 0.0547}$ | $0.8582_{\pm 0.0330}$ | $0.7445_{\pm 0.0511}$ | $\mathbf{0.8899_{\pm 0.0148}}$ |
| | | Discriminator | $0.7332_{\pm 0.0529}$ | $0.8976_{\pm 0.0148}$ | $0.8071_{\pm 0.0362}$ | - |
| | | LogReg | $0.7953_{\pm 0.0421}$ | $0.8867_{\pm 0.0207}$ | $0.7871_{\pm 0.0351}$ | $0.8668_{\pm 0.0336}$ |
| | | MLP | $0.6960_{\pm 0.0456}$ | $0.8599_{\pm 0.0291}$ | $0.8025_{\pm 0.0212}$ | $0.8404_{\pm 0.0400}$ |
| | $\beta$ MSE ↓ | None | $19.2959_{\pm 4.0480}$ | $8.3074_{\pm 1.6718}$ | $18.0835_{\pm 2.5051}$ | $7.9052_{\pm 0.3837}$ |
| | | BetaNoised | $14.4350_{\pm 2.3116}$ | $6.4683_{\pm 0.9572}$ | $23.0590_{\pm 3.2307}$ | $5.4736_{\pm 0.1792}$ |
| | | BetaDebiased | $\mathbf{13.1578_{\pm 2.9727}}$ | $\mathbf{5.6890_{\pm 1.0695}}$ | $19.1627_{\pm 6.1430}$ | $6.4776_{\pm 0.1134}$ |
| | | DP-MLP | $18.7059_{\pm 3.0658}$ | $8.8820_{\pm 1.4421}$ | $24.0433_{\pm 3.4451}$ | $\mathbf{3.0883_{\pm 0.2703}}$ |
| | | Discriminator | $18.9194_{\pm 4.0483}$ | $8.0682_{\pm 1.5928}$ | $\mathbf{13.6267_{\pm 1.9313}}$ | - |
| | | LogReg | $14.4464_{\pm 2.3126}$ | $6.4701_{\pm 0.9581}$ | $23.0696_{\pm 3.2327}$ | $5.4706_{\pm 0.1781}$ |
| | | MLP | $18.2400_{\pm 3.1143}$ | $9.7111_{\pm 1.4901}$ | $23.0268_{\pm 3.2550}$ | $2.4589_{\pm 0.3184}$ |
| | WST ↓ | None | $0.6642_{\pm 0.0270}$ | $0.4723_{\pm 0.0294}$ | $0.5645_{\pm 0.0219}$ | $0.2928_{\pm 0.0118}$ |
| | | BetaNoised | $0.2507_{\pm 0.0384}$ | $0.3078_{\pm 0.0231}$ | $0.2608_{\pm 0.0370}$ | $0.2269_{\pm 0.0036}$ |
| | | BetaDebiased | $0.2316_{\pm 0.0670}$ | $0.2892_{\pm 0.0442}$ | $0.3029_{\pm 0.0883}$ | $0.2176_{\pm 0.0076}$ |
| | | DP-MLP | $\mathbf{0.1395_{\pm 0.0262}}$ | $\mathbf{0.0957_{\pm 0.0183}}$ | $0.1730_{\pm 0.0413}$ | $\mathbf{0.1142_{\pm 0.0017}}$ |
| | | Discriminator | $0.6303_{\pm 0.0278}$ | $0.3596_{\pm 0.0470}$ | $\mathbf{0.0436_{\pm 0.0100}}$ | - |
| | | LogReg | $0.2504_{\pm 0.0384}$ | $0.3083_{\pm 0.0231}$ | $0.2607_{\pm 0.0370}$ | $0.2272_{\pm 0.0035}$ |
| | | MLP | $0.0658_{\pm 0.0208}$ | $0.0409_{\pm 0.0104}$ | $0.0787_{\pm 0.0325}$ | $0.2025_{\pm 0.0004}$ |

Table 5: Results on Banknote averaged over 10 seeds.

| | | SDGP | GAN | DPGAN | PrivBayes |
|---|---|---|---|---|---|
| $\epsilon = 1$ | MLP MSE ↓ | None | $1.4464_{\pm 0.1591}$ | $1.8851_{\pm 0.5262}$ | $0.1973_{\pm 0.0108}$ |
| | | BetaNoised | $0.6455_{\pm 0.0942}$ | $1.0057_{\pm 0.1973}$ | $0.2200_{\pm 0.0154}$ |
| | | BetaDebiased | $\mathbf{0.6421_{\pm 0.1290}}$ | $\mathbf{0.9024_{\pm 0.1244}}$ | $0.2139_{\pm 0.0122}$ |
| | | DP-MLP | $0.8279_{\pm 0.0974}$ | $0.9462_{\pm 0.1702}$ | $\mathbf{0.1877_{\pm 0.0174}}$ |
| | | Discriminator | $1.5126_{\pm 0.1639}$ | $1.6256_{\pm 0.2394}$ | - |
| | | LogReg | $0.6292_{\pm 0.0909}$ | $1.0606_{\pm 0.2648}$ | $0.2515_{\pm 0.0305}$ |
| | | MLP | $0.6266_{\pm 0.1273}$ | $1.0979_{\pm 0.2225}$ | $0.1697_{\pm 0.0079}$ |
| | $\beta$ MSE ↓ | None | $0.1017_{\pm 0.0118}$ | $0.1867_{\pm 0.0434}$ | $\mathbf{0.0011_{\pm 0.0002}}$ |
| | | BetaNoised | $0.0601_{\pm 0.0172}$ | $0.1761_{\pm 0.0948}$ | $0.0088_{\pm 0.0028}$ |
| | | BetaDebiased | $0.0608_{\pm 0.0190}$ | $\mathbf{0.0667_{\pm 0.0188}}$ | $0.0077_{\pm 0.0022}$ |
| | | DP-MLP | $\mathbf{0.0363_{\pm 0.0192}}$ | $0.1530_{\pm 0.0812}$ | $0.0048_{\pm 0.0024}$ |
| | | Discriminator | $0.0940_{\pm 0.0100}$ | $0.1567_{\pm 0.1825}$ | - |
| | | LogReg | $0.0707_{\pm 0.0194}$ | $0.0749_{\pm 0.0279}$ | $0.0037_{\pm 0.0016}$ |
| | | MLP | $0.0058_{\pm 0.0007}$ | $0.1476_{\pm 0.0804}$ | $0.0008_{\pm 0.0002}$ |
| | WST ↓ | None | $1.3060_{\pm 0.0319}$ | $2.2013_{\pm 0.0945}$ | $1.3938_{\pm 0.0231}$ |
| | | BetaNoised | $1.0060_{\pm 0.0023}$ | $2.0922_{\pm 0.0419}$ | $1.3009_{\pm 0.0338}$ |
| | | BetaDebiased | $1.0023_{\pm 0.0009}$ | $2.0930_{\pm 0.0393}$ | $1.2705_{\pm 0.0290}$ |
| | | DP-MLP | $1.0036_{\pm 0.0015}$ | $2.0542_{\pm 0.0184}$ | $\mathbf{1.0265_{\pm 0.0035}}$ |
| | | Discriminator | $\mathbf{0.9472_{\pm 0.0764}}$ | $\mathbf{2.0145_{\pm 0.0141}}$ | - |
| | | LogReg | $1.0070_{\pm 0.0042}$ | $2.2051_{\pm 0.0819}$ | $1.4078_{\pm 0.0492}$ |
| | | MLP | $1.0001_{\pm 0.0001}$ | $2.0350_{\pm 0.0158}$ | $1.0072_{\pm 0.0009}$ |
| $\epsilon = 6$ | MLP MSE ↓ | None | $1.8218_{\pm 0.1514}$ | $1.8016_{\pm 0.1771}$ | $0.1633_{\pm 0.0074}$ |
| | | BetaNoised | $\mathbf{0.5318_{\pm 0.0806}}$ | $\mathbf{0.6529_{\pm 0.0814}}$ | $0.1940_{\pm 0.0156}$ |
| | | BetaDebiased | $0.5647_{\pm 0.1065}$ | $0.9025_{\pm 0.1462}$ | $0.1810_{\pm 0.0131}$ |
| | | DP-MLP | $0.9737_{\pm 0.1178}$ | $1.0902_{\pm 0.1486}$ | $\mathbf{0.1428_{\pm 0.0068}}$ |
| | | Discriminator | $1.8398_{\pm 0.1446}$ | $1.8631_{\pm 0.1986}$ | - |
| | | LogReg | $0.5501_{\pm 0.0540}$ | $0.9050_{\pm 0.1553}$ | $0.1934_{\pm 0.0224}$ |
| | | MLP | $0.4725_{\pm 0.0736}$ | $0.7464_{\pm 0.1185}$ | $0.1581_{\pm 0.0076}$ |
| | $\beta$ MSE ↓ | None | $0.1230_{\pm 0.0110}$ | $0.1450_{\pm 0.0174}$ | $0.0009_{\pm 0.0002}$ |
| | | BetaNoised | $0.0695_{\pm 0.0203}$ | $0.0608_{\pm 0.0231}$ | $0.0022_{\pm 0.0006}$ |
| | | BetaDebiased | $0.0693_{\pm 0.0207}$ | $0.0613_{\pm 0.0240}$ | $0.0018_{\pm 0.0004}$ |
| | | DP-MLP | $\mathbf{0.0030_{\pm 0.0006}}$ | $\mathbf{0.0354_{\pm 0.0112}}$ | $\mathbf{0.0008_{\pm 0.0002}}$ |
| | | Discriminator | $0.1135_{\pm 0.0098}$ | $0.2274_{\pm 0.0375}$ | - |
| | | LogReg | $0.0697_{\pm 0.0207}$ | $0.0606_{\pm 0.0237}$ | $0.0018_{\pm 0.0004}$ |
| | | MLP | $0.0063_{\pm 0.0011}$ | $0.0212_{\pm 0.0060}$ | $0.0008_{\pm 0.0001}$ |
| | WST ↓ | None | $1.3727_{\pm 0.0249}$ | $1.5681_{\pm 0.0368}$ | $1.3306_{\pm 0.0271}$ |
| | | BetaNoised | $1.0031_{\pm 0.0012}$ | $1.0615_{\pm 0.0304}$ | $1.3906_{\pm 0.0410}$ |
| | | BetaDebiased | $\mathbf{1.0031_{\pm 0.0012}}$ | $1.0598_{\pm 0.0286}$ | $1.4106_{\pm 0.0432}$ |
| | | DP-MLP | $1.0140_{\pm 0.0032}$ | $\mathbf{1.0338_{\pm 0.0126}}$ | $\mathbf{1.2405_{\pm 0.0133}}$ |
| | | Discriminator | $1.0481_{\pm 0.0752}$ | $1.3844_{\pm 0.0654}$ | - |
| | | LogReg | $1.0031_{\pm 0.0012}$ | $1.0623_{\pm 0.0298}$ | $1.4033_{\pm 0.0406}$ |
| | | MLP | $1.0001_{\pm 0.0000}$ | $1.0081_{\pm 0.0045}$ | $1.0097_{\pm 0.0010}$ |

Table 6: Results on Boston averaged over 10 seeds.

| | | SDGP | CGAN | DPCGAN | DPGAN | PrivBayes |
|---|---|---|---|---|---|---|
| $\epsilon = 1$ | MLP-ROC-AUC ↑ | None | $0.6801_{\pm 0.0655}$ | $0.6374_{\pm 0.0421}$ | $0.6791_{\pm 0.0966}$ | $0.8366_{\pm 0.0579}$ |
| | | BetaNoised | $0.7732_{\pm 0.0589}$ | $0.6110_{\pm 0.0477}$ | $0.6546_{\pm 0.0727}$ | $0.7076_{\pm 0.0983}$ |
| | | BetaDebiased | $0.7151_{\pm 0.1146}$ | $0.6820_{\pm 0.0510}$ | $0.7173_{\pm 0.0842}$ | $\mathbf{0.8557_{\pm 0.0765}}$ |
| | | DP-MLP | $0.7166_{\pm 0.1038}$ | $\mathbf{0.7942_{\pm 0.0404}}$ | $0.5686_{\pm 0.0823}$ | $0.7353_{\pm 0.0887}$ |
| | | Discriminator | $\mathbf{0.8607_{\pm 0.0485}}$ | $0.6992_{\pm 0.0839}$ | $\mathbf{0.7290_{\pm 0.0720}}$ | - |
| | | LogReg | $0.7141_{\pm 0.0755}$ | $0.6631_{\pm 0.0469}$ | $0.6484_{\pm 0.1081}$ | $0.7618_{\pm 0.1019}$ |
| | | MLP | $0.6942_{\pm 0.1262}$ | $0.7730_{\pm 0.0412}$ | $0.7358_{\pm 0.1017}$ | $0.7573_{\pm 0.0738}$ |
| | $\beta$ MSE ↓ | None | $2.3646_{\pm 0.2983}$ | $2.0643_{\pm 0.2012}$ | $4.9828_{\pm 1.5701}$ | $2.3904_{\pm 0.1050}$ |
| | | BetaNoised | $1.4900_{\pm 0.1807}$ | $2.7532_{\pm 0.2650}$ | $2.5025_{\pm 0.3763}$ | $2.1144_{\pm 0.2400}$ |
| | | BetaDebiased | $1.5413_{\pm 0.2378}$ | $2.8337_{\pm 0.3842}$ | $\mathbf{2.2324_{\pm 1.0446}}$ | $\mathbf{1.8266_{\pm 0.2392}}$ |
| | | DP-MLP | $\mathbf{0.9977_{\pm 0.1617}}$ | $2.3965_{\pm 0.2083}$ | $3.8865_{\pm 0.6043}$ | $2.3130_{\pm 0.2195}$ |
| | | Discriminator | $1.8554_{\pm 0.3263}$ | $\mathbf{1.4591_{\pm 0.1837}}$ | $4.0612_{\pm 0.9523}$ | - |
| | | LogReg | $1.1940_{\pm 0.1610}$ | $2.6934_{\pm 0.2667}$ | $2.2156_{\pm 0.3366}$ | $1.5333_{\pm 0.2138}$ |
| | | MLP | $1.0120_{\pm 0.1383}$ | $2.3999_{\pm 0.2040}$ | $3.8343_{\pm 0.7032}$ | $1.6581_{\pm 0.2020}$ |
| | WST ↓ | None | $1.8426_{\pm 0.1329}$ | $2.3665_{\pm 0.0982}$ | $1.5853_{\pm 0.1333}$ | $2.1117_{\pm 0.1740}$ |
| | | BetaNoised | $1.3109_{\pm 0.0507}$ | $1.4337_{\pm 0.1114}$ | $2.2232_{\pm 0.2325}$ | $1.2322_{\pm 0.0823}$ |
| | | BetaDebiased | $\mathbf{1.0649_{\pm 0.0120}}$ | $1.8922_{\pm 0.1237}$ | $1.9913_{\pm 0.3507}$ | $\mathbf{1.1825_{\pm 0.0933}}$ |
| | | DP-MLP | $1.4737_{\pm 0.1027}$ | $1.4570_{\pm 0.1492}$ | $1.0315_{\pm 0.1415}$ | $1.2190_{\pm 0.0795}$ |
| | | Discriminator | $1.8814_{\pm 0.1682}$ | $\mathbf{1.0007_{\pm 0.0004}}$ | $\mathbf{1.0001_{\pm 0.0001}}$ | - |
| | | LogReg | $1.4374_{\pm 0.0467}$ | $1.6451_{\pm 0.1168}$ | $2.2953_{\pm 0.2121}$ | $1.4663_{\pm 0.1152}$ |
| | | MLP | $1.3056_{\pm 0.0524}$ | $1.6129_{\pm 0.1404}$ | $1.0709_{\pm 0.1579}$ | $1.4141_{\pm 0.1216}$ |
| $\epsilon = 6$ | MLP-ROC-AUC ↑ | None | $0.6177_{\pm 0.0737}$ | $\mathbf{0.9790_{\pm 0.0058}}$ | $\mathbf{0.9756_{\pm 0.0042}}$ | $0.9435_{\pm 0.0152}$ |
| | | BetaNoised | $0.7185_{\pm 0.0898}$ | $0.9715_{\pm 0.0031}$ | $0.9710_{\pm 0.0065}$ | $0.9699_{\pm 0.0121}$ |
| | | BetaDebiased | $\mathbf{0.9070_{\pm 0.0434}}$ | $0.9723_{\pm 0.0033}$ | $0.9724_{\pm 0.0066}$ | $\mathbf{0.9820_{\pm 0.0064}}$ |
| | | DP-MLP | $0.7203_{\pm 0.1028}$ | $0.9703_{\pm 0.0040}$ | $0.9728_{\pm 0.0059}$ | $0.9754_{\pm 0.0063}$ |
| | | Discriminator | $0.8712_{\pm 0.0471}$ | $0.9763_{\pm 0.0071}$ | $0.9737_{\pm 0.0065}$ | - |
| | | LogReg | $0.6869_{\pm 0.0760}$ | $0.9706_{\pm 0.0033}$ | $0.9719_{\pm 0.0049}$ | $0.9825_{\pm 0.0061}$ |
| | | MLP | $0.6899_{\pm 0.1290}$ | $0.9584_{\pm 0.0080}$ | $0.9767_{\pm 0.0043}$ | $0.9506_{\pm 0.0250}$ |
| | $\beta$ MSE ↓ | None | $2.3602_{\pm 0.4035}$ | $0.9886_{\pm 0.2287}$ | $1.0653_{\pm 0.1229}$ | $\mathbf{0.9142_{\pm 0.1575}}$ |
| | | BetaNoised | $1.2400_{\pm 0.1637}$ | $1.0329_{\pm 0.0732}$ | $1.1586_{\pm 0.1312}$ | $1.0465_{\pm 0.1358}$ |
| | | BetaDebiased | $\mathbf{0.9388_{\pm 0.0802}}$ | $1.0150_{\pm 0.0783}$ | $1.1617_{\pm 0.1936}$ | $0.9843_{\pm 0.1766}$ |
| | | DP-MLP | $0.9949_{\pm 0.1486}$ | $1.0119_{\pm 0.0698}$ | $0.8969_{\pm 0.0837}$ | $1.3442_{\pm 0.0900}$ |
| | | Discriminator | $1.7588_{\pm 0.3421}$ | $\mathbf{0.8539_{\pm 0.2323}}$ | $\mathbf{0.5423_{\pm 0.0457}}$ | - |
| | | LogReg | $1.2221_{\pm 0.1598}$ | $1.0310_{\pm 0.0719}$ | $1.1484_{\pm 0.1276}$ | $1.0234_{\pm 0.1274}$ |
| | | MLP | $1.0845_{\pm 0.1210}$ | $1.0953_{\pm 0.0844}$ | $0.9275_{\pm 0.0938}$ | $1.5354_{\pm 0.1343}$ |
| | WST ↓ | None | $1.8436_{\pm 0.1257}$ | $1.3378_{\pm 0.0282}$ | $1.6449_{\pm 0.0849}$ | $2.0437_{\pm 0.2188}$ |
| | | BetaNoised | $1.4164_{\pm 0.0483}$ | $0.6526_{\pm 0.0463}$ | $1.5485_{\pm 0.0635}$ | $1.4808_{\pm 0.0943}$ |
| | | BetaDebiased | $\mathbf{1.3314_{\pm 0.0459}}$ | $0.6641_{\pm 0.0482}$ | $1.5156_{\pm 0.0935}$ | $\mathbf{1.4133_{\pm 0.1346}}$ |
| | | DP-MLP | $1.7176_{\pm 0.1206}$ | $0.7931_{\pm 0.0380}$ | $1.5551_{\pm 0.0826}$ | $1.4923_{\pm 0.0685}$ |
| | | Discriminator | $1.8523_{\pm 0.1553}$ | $\mathbf{0.2363_{\pm 0.0425}}$ | $\mathbf{1.1020_{\pm 0.0158}}$ | - |
| | | LogReg | $1.4140_{\pm 0.0493}$ | $0.6597_{\pm 0.0470}$ | $1.5281_{\pm 0.0622}$ | $1.4824_{\pm 0.0952}$ |
| | | MLP | $1.3487_{\pm 0.0591}$ | $0.3762_{\pm 0.0383}$ | $1.2309_{\pm 0.0387}$ | $1.3406_{\pm 0.0792}$ |

Table 7: Results on Breast averaged over 10 seeds.

### C.8 COMPARISON TO EXPERIMENTAL RESULTS REPORTED BY RELATED WORK

We compare our results to PATE-GAN and DPGAN as DP synthetic data generators (Jordon et al., 2019; Xie et al., 2018). The PATEGAN implementation is taken from `https://github.com/vanderschaarlab/mlforhealthlabpub`. For DPGAN we chose the code from the DataSynthesizer package. In the implementation of the PATE-GAN method, Jordon et al. (2019) generate 50 independent synthetic data sets for each function call, returning the best synthetic data set as defined by a comparison with non-private validation data. The relative level of privacy violation in these situations is unknown, making interpretation of results and comparison between methods in tables and figures challenging. On re-implementing the methods to generate DP synthetic data, we find a substantial and significant drop in performance, which nonetheless is improved through bias mitigation. Please see the GitHub repository for further results and an illustration why PATE GAN underperforms.

## References


Borja Balle and Yu-Xiang Wang. Improving the gaussian mechanism for differential privacy: Analytical calibration and optimal denoising. In *International Conference on Machine Learning*, pages 394–403. PMLR, 2018.

Avrim Blum, Cynthia Dwork, Frank McSherry, and Kobbi Nissim. Practical privacy: the sulq framework. In *Proceedings of the twenty-fourth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 128–138, 2005.

Kamalika Chaudhuri, Claire Monteleoni, and Anand D Sarwate. Differentially private empirical risk minimization. *Journal of Machine Learning Research*, 12(3), 2011.

Victor Chernozhukov and Han Hong. An MCMC approach to classical estimation. *Journal of Econometrics*, 115(2):293–346, 2003.

Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of cryptography conference*, pages 265–284. Springer, 2006.

Hong Ge, Kai Xu, and Zoubin Ghahramani. Turing: a language for flexible probabilistic inference. In *International Conference on Artificial Intelligence and Statistics, AISTATS 2018, 9-11 April 2018, Playa Blanca, Lanzarote, Canary Islands, Spain*, pages 1682–1690, 2018. URL `http://proceedings.mlr.press/v84/ge18b.html`.

Zhanglong Ji and Charles Elkan. Differential privacy based on importance weighting. *Machine Learning*, 93(1):163–183, 2013.

James Jordon, Jinsung Yoon, and Mihaela van der Schaar. Pate-gan: Generating synthetic data with differential privacy guarantees. In *International Conference on Learning Representations*, 2019.

BJK Kleijn, AW Van der Vaart, et al. The Bernstein-von-Mises theorem under misspecification. *Electronic Journal of Statistics*, 6: 354–381, 2012.

Siem Jan Koopman, Neil Shephard, and Drew Creal. Testing the assumptions behind importance sampling. *Journal of Econometrics*, 149 (1):2–11, 2009.

Tomasz J Kozubowski and Krzysztof Podgórski. Log-Laplace distributions. *International Mathematical Journal*, 3(4):467–495, 2003.

Meelis Kull, Telmo Silva Filho, and Peter Flach. Beta calibration: a well-founded and easily implemented improvement on logistic calibration for binary classifiers. In *Artificial Intelligence and Statistics*, pages 623–631. PMLR, 2017.

Simon P Lyddon, Chris Holmes, and Stephen Walker. General Bayesian updating and the loss-likelihood bootstrap. *Biometrika*, 2018.

Reihaneh Torkzadehmahani, Peter Kairouz, and Benedict Paten. Dp-cgan: Differentially private synthetic data and label generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0, 2019.

Ryan Turner, Jane Hung, Eric Frank, Yunus Saatchi, and Jason Yosinski. Metropolis–Hastings generative adversarial networks. In *International Conference on Machine Learning*, pages 6345–6353. PMLR, 2019.

Aad W Van der Vaart. *Asymptotic Statistics*, volume 3. Cambridge University Press, 2000.

Aki Vehtari, Daniel Simpson, Andrew Gelman, Yuling Yao, and Jonah Gabry. Pareto smoothed importance sampling. *arXiv preprint arXiv:1507.02646*, 2015.

Harrison Wilde, Jack Jewson, Sebastian Vollmer, and Chris Holmes. Foundations of Bayesian learning from synthetic data. *arXiv preprint arXiv:2011.08299*, 2020.

Liyang Xie, Kaixiang Lin, Shu Wang, Fei Wang, and Jiayu Zhou. Differentially private generative adversarial network. *arXiv preprint arXiv:1802.06739*, 2018.

## Statement of Authorship for joint/multi-authored papers for PGR thesis

To appear at the end of each thesis chapter submitted as an article/paper

The statement shall describe the candidate's and co-authors' independent research contributions in the thesis publications. For each publication there should exist a complete statement that is to be filled out and signed by the candidate and supervisor **(only required where there isn't already a statement of contribution within the paper itself).**

| | |
|---|---|
| Title of Paper | Mitigating statistical bias within differentially private synthetic data |
| Publication Status | xPublished □ Accepted for Publication<br>□Submitted for Publication □Unpublished and unsubmitted work written in a manuscript style |
| Publication Details | Ghalebikesabi, S., Wilde, H., Jewson, J., Doucet, A., Vollmer, S., & Holmes, C. (2022, August). Mitigating statistical bias within differentially private synthetic data. In Uncertainty in Artificial Intelligence (pp. 696-705). PMLR. |

### Student Confirmation

| | | | |
|---|---|---|---|
| Student Name: | Sahra Ghalebikesabi | | |
| Contribution to the Paper | I conceived part of the methodology, implementation and experiments. The idea was conceived by Chris Holmes. During frequent meetings, my collaborators helped out with helpful suggestions on methodology and feedback. They also contributed to proofs, checking the results, and proof-reading. | | |
| Signature |  | Date | 28.10.2023 |

### Supervisor Confirmation

By signing the Statement of Authorship, you are certifying that the candidate made a substantial contribution to the publication, and that the description described above is accurate.

| | | |
|---|---|---|
| Supervisor name and title: Chris Holmes | | |
| Supervisor comments<br><br>I am in agreement with the description of the contributions. | | |
| Signature | Date | 28.10.2023 |

This completed form should be included in the thesis, at the end of the relevant chapter.

# 6

# Differentially Private Diffusion Models Generate Useful Synthetic Images

# Differentially Private Diffusion Models Generate Useful Synthetic Images

**Sahra Ghalebikesabi**[1,+], **Leonard Berrada**[2], **Sven Gowal**[2], **Ira Ktena**[2], **Robert Stanforth**[2], **Jamie Hayes**[2], **Soham De**[2], **Olivia Wiles**[2], **Borja Balle**[2],

[1]University of Oxford
[2]DeepMind
[+]Work done at DeepMind.
sahra.ghalebikesabi@univ.ox.ac.uk, {lberrada — bballe}@deepmind.com

## Abstract

The ability to generate privacy-preserving synthetic versions of sensitive image datasets could unlock numerous ML applications currently constrained by data availability. Due to their astonishing image generation quality, diffusion models are a prime candidate for generating high-quality synthetic data. However, recent studies have found that, by default, the outputs of some diffusion models do not preserve training data privacy.

By privately fine-tuning ImageNet pre-trained diffusion models with more than 80M parameters, we obtain SOTA results on CIFAR-10 and Camelyon17 in terms of both FID and the accuracy of downstream classifiers trained on synthetic data. We decrease the SOTA FID on CIFAR-10 from 26.8 to 9.8, and increase the accuracy from 51.0% to 88.0%. On synthetic data from Camelyon17, we achieve a downstream accuracy of 91.1% which is close to the SOTA of 96.5% when training on the real data. We use generative models to create infinite amounts of data to maximise the downstream prediction performance, and further show how to use synthetic data for hyperparameter tuning. Our results show that diffusion models fine-tuned with differential privacy can produce useful and provably private synthetic data, even in applications with significant distribution shift between pre-training and fine-tuning distributions.

## 1 Introduction

Delivering impactful ML-based solutions for real-world applications in domains like health care and recommendation systems requires access to sensitive personal data that cannot be readily used or shared without risk of introducing ethical and legal implications. Replacing real sensitive data with private synthetic data following the same distribution is a clear pathway to mitigating these concerns. However, despite their theoretical appeal, general-purpose methods for generating useful and provably private synthetic data remain a subject of active research (Dockhorn *et al.*, 2022). The central challenge in this line of work is how to obtain truly privacy-preserving synthetic data free of the common pitfalls faced by classical anonymization approaches (Stadler *et al.*, 2021), while at the same time ensuring the resulting datasets remain useful for a wide variety of downstream tasks, including statistical and exploratory analysis as well as machine learning model selection, training and testing.

It is tempting to obtain synthetic data by training and then sampling from well-known generative models like variational auto-encoders (Kingma and Welling, 2013), generative adversarial nets (Goodfellow *et al.*, 2020), and denoising diffusion probabilistic models (Ho *et al.*, 2020; Song and Ermon, 2019). Unfortunately, it is well-known that out-of-the-box generative models can potentially memorise and regenerate their training data points[1] and, thus, reveal private information. This holds for variational autoencoders (Hilprecht *et al.*, 2019), generative adversarial nets (Hayes *et al.*, 2017), and also diffusion models (Carlini *et al.*, 2023; Duan *et al.*, 2023; Hu and Pang, 2023; Somepalli *et al.*, 2022). In particular, diffusion models have recently gained a lot of attention, with pre-trained models made available online (e.g. Dhariwal and Nichol, 2021), and being fine-tuned on potentially sensitive data such as chest X-rays (e.g. Chambon *et al.*, 2022) and brain MRIs (e.g. Rouzrokh *et al.*, 2022).

Mitigating the privacy loss from sharing synthetic data produced by generative models trained on sensitive data is not straightforward. Differential privacy (DP) (Dwork *et al.*, 2006) has emerged as the gold standard privacy mitigation when training ML models, and its application to generative models would provide guarantees on the information the model (and synthetic data sampled from it) can leak about individual training examples. Yet, scaling up DP training methods to modern large-scale models remains a significant challenge due to the slow down incurred by DP-SGD (the standard workhorse of DP for deep learning) (Wang *et al.*, 2017) and the stark utility degradation often observed when training large models from scratch with DP (Stadler and Troncoso, 2022; Stadler *et al.*, 2021; Zhang *et al.*, 2022). Most previous works on DP generative models worked around these issues by focusing on small models, low-complexity data (Harder *et al.*, 2021; Torkzadehmahani *et al.*, 2019; Xie *et al.*, 2018) or using non-standard models (Harder *et al.*, 2022). However,

[1]A model does not contain its training data, but rather has "memorised" training data when the model is able to use the rules and attributes it has learned about the training data to generate elements of that training data.


Figure 1: DP diffusion models are capable of producing high-quality images. More images can be found in Figures 4, 5, 6.

for DP applications to image classification it is known that using models pre-trained on public data is a method for attaining good utility which is compatible with large-scale models and complex datasets (e.g. De *et al.*, 2022).

**Contributions.** In this paper we demonstrate how to accurately train standard diffusion models with differential privacy. Despite the inherent difficulty of this task, we propose a simple methodology that allows us to generate high-quality synthetic image datasets that are useful for a variety of important downstream tasks. In particular, we privately train denoising diffusion probabilistic models (Ho *et al.*, 2020) with more than 80M parameters on CIFAR-10 and Camelyon17 (Koh *et al.*, 2021), and evaluate the usefulness of synthetic images for downstream model training and evaluation. Crucially, we show that by pre-training on publicly available data (i.e. ImageNet), it is possible to considerably outperform the results of extremely recent work on a similar topic (Dockhorn *et al.*, 2022). With this method, we are able to accurately train models 45x larger than Dockhorn *et al.* (2022) and to achieve a high utility on datasets that are significantly more challenging (e.g. CIFAR-10 and a medical dataset -– instead of MNIST). Further, we demonstrate that the accuracy of downstream classifiers can be improved even more with no additional privacy cost by leveraging larger synthetic datasets and ensembling. Finally, we show that hyperparameter tuning downstream classifiers on the synthetic data reveals trends that are also reflected on the private data set. The long version of our work including the supplementary material can be found on https://arxiv.org/pdf/2302.13861.pdf.

## 2 Related Work

**DP synthetic image generation.** DP image generation is an active area of research. Most efforts have focused on applying a DP stochastic gradient procedure on popular generative models, i.e. generative adversarial networks (Chen *et al.*, 2021; Yoon *et al.*, 2020), or variational autoencoders (Jiang *et al.*, 2022; Pfitzner and Arnrich, 2022). Only one other work has so far analysed the application of DP gradient descent on diffusion models (e.g. Dockhorn *et al.*, 2022) which we contrast against in Table 2. Others have instead proposed custom architectures. Harder *et al.* (2022), e.g., pre-train a perceptual feature extractor using public data, then privatize the mean of the feature embeddings of the sensitive data records, and use the privatised mean to train a generative model.

**Limitations of previous work.** DP image generation based on custom training pipelines and architectures that are not used outside of the DP literature do not profit from the constant research progress on public image generation. Other works that instead build upon popular public generative approaches have been shown to not be DP despite such claims. This could be either due to faulty implementations or proofs. See Stadler *et al.* (2021) for successful privacy attacks on DP GANs, or Appendix B of Dockhorn *et al.* (2022) for an illustration on why DPGEN (Chen *et al.*, 2022) does not actually satisfy DP guarantees.

**Limited success on natural images.** In the space of image generation, positive results of DP synthesizers have only been reported on MNIST, FashionMNIST and CelebA (downsampled to $32 \times 32$) (e.g. Harder *et al.*, 2021). These datasets are relatively easy to learn thanks to plain backgrounds, class separability, and repetitive image features within and even across classes. Meanwhile, CIFAR-10 has been established as a considerably harder generation task than MNIST, FashionMNIST or CelebA (Radiuk, 2017). The images are not only higher dimensional than MNIST and FashionMNIST ($32 \times 32 \times 3$ compared to $28 \times 28$ feature dimensions), but the dataset has wider diversity and complexity. This is reflected by more complex features and textures, such as lightning conditions, object orientations, and complex backgrounds (Radiuk, 2017). Moreover, MNIST and FashionMNIST are considerably lower dimensional than CIFAR-10 and Camelyon ($28 \times 28$ vs $32 \times 32 \times 3$ features), and CelebA is downsampled to the same feature dimensionality as CIFAR-10 but has more than 3 times as many samples as CIFAR-10 (50k vs 162k) which considerably reduces the information loss introduced by DP training. As far as we know, only two other concurrent works have attempted DP image generation on CIFAR-10. While Dockhorn *et al.* (2022) achieve a FID of only 97.7 by training a DP diffusion model from scratch, Harder *et al.* (2022) used pre-training on ImageNet and achieved the SOTA with a FID of 26.8, and a downstream accuracy of only 50%.

**Limited targeted evaluation.** The evaluation carried out on DP synthetic datasets is often not sufficiently targeted towards their utility in practice. The performance of DP image synthesizers is commonly evaluated on two types of metrics: 1) perceptual difference measures between the synthetic and real data distribution, such as FID, and 2) predictive performance of classifiers that are trained on a synthetic dataset of the size of the original training dataset and tested on the real test data. The former metric is known to be easy to manipulate with factors not related to the image quality, such as the number of samples, or the version number of the inception network (Kynkäänniemi *et al.*, 2022). At the same time, it is not obvious how to jointly incorporate the information from both metrics given that they may individually imply different conclusions. Dockhorn *et al.* (2022), for instance, identify different diffusion model samplers to minimise either the FID *or* the downstream test loss. In this paper, we set out to provide examples of additional downstream evaluations.

# 3 Background

## 3.1 Denoising Diffusion Probabilistic Models

Denoising diffusion models (Ho *et al.*, 2020; Sohl-Dickstein *et al.*, 2015; Song *et al.*, 2020) are a class of likelihood-based generative models that have recently established new SOTA results across diverse computer vision problems (Dhariwal and Nichol, 2021; Lugmayr *et al.*, 2022; Rombach *et al.*, 2022). Given $x_0 \sim q(x_0)$, one defines a forward process that generates gradually noisier samples $x_1, ..., x_T$ using a transition kernel $q(x_t|x_{t-1})$ typically chosen as a Gaussian perturbation. At inference time, $x_T$, an observation sampled from a noise distribution, is then gradually denoised $x_{T-1}, x_{T-2}, ...$ until the final sample $x_0$ is reached. Ho *et al.* (2020) parameterize this process by a function $\epsilon_\theta(x_t, t)$ which predicts the noise component $\epsilon$ of a noisy sample $x_t$ given timestep $t$. They then propose a simplified training objective to learn $\theta$:

$$L(\theta) = \mathbb{E}_{t,x_0,\epsilon}[\|\epsilon - \epsilon_\theta(x_t, t)\|^2], \tag{1}$$

with $t \sim \mathcal{U}\{0, T\}$, where $T$ is the pre-specified maximum timestep, and $\mathcal{U}\{a, b\}$ is the discrete uniform distribution bounded by $a$ and $b$. The noisy sample $x_t$ is defined by $x_t = \sqrt{\overline{\alpha}_t}x_0 + \sqrt{1 - \overline{\alpha}_t}\epsilon$ where $\epsilon \sim \mathcal{N}(0, I)$ is a noise sample of the same dimensionality as the data sample $x_0$, and $\overline{\alpha}_t$ is defined such that $x_t$ follows the pre-specified forward process. Most importantly, $\overline{\alpha}_t = \prod_{s=1}^{T}(1 - \beta_s)$ is a decreasing function of timestep $t$ where the form of $\beta_s$ is a hyperparameter. Thus, the larger $t$ is, the noisier $x_t$ will be.

## 3.2 Differential Privacy

Differential Privacy (DP) is a formal privacy notion that, in informal terms, bounds how much a single observation can change the output distribution of a randomised algorithm:

**Definition 1** (Differential Privacy (Dwork *et al.*, 2006))**.** *Let $A$ be a randomized algorithm, and let $\varepsilon > 0$, $\delta \in [0, 1]$. We say that $A$ is $(\varepsilon, \delta)$-DP if for any two neighboring datasets $D, D'$ differing by a single element and $\mathcal{S}$ denoting the support of $A$, we have that*

$$\forall\, S \subset \mathcal{S},\ \mathbb{P}[A(D) \in S] \leq \exp(\varepsilon)\mathbb{P}[A(D') \in S] + \delta.$$

The privacy guarantee is thus controlled by two parameters, $\varepsilon$ and $\delta$. While $\varepsilon$ bounds the log-likelihood ratio of any particular output that can be obtained when running the algorithm on two datasets differing in a single data point, $\delta$ is a small probability which bounds the occurrence of infrequent outputs that violate this bound (typically $1/n$, where $n$ is the number of training examples). The smaller these two parameters get, the higher is the privacy guarantee. We therefore refer to the tuple $(\varepsilon, \delta)$ as privacy budget.

## 3.3 DP Stochastic Gradient Descent

Neural networks are typically privatised with Differentially Private Stochastic Gradient Descent (DP-SGD) (Abadi *et al.*, 2016), or alternatively a different DP optimizer like DP-Adam (McMahan *et al.*, 2018). At each training iteration, the mini-batch gradient is clipped per example, and Gaussian noise is added to it. More formally, let $l_i(\theta) := \mathcal{L}(\theta, x_i, y_i)$ denote the learning objective given model parameters $\theta \in \mathbb{R}^p$, input features $x_i$ and label $y_i$ of observation $i$. When training diffusion models, we estimate the loss function by $\mathcal{L}(\theta, x_i, y_i, t_i, \epsilon_i) = \|\epsilon_i - \epsilon_\theta(\sqrt{\overline{\alpha}_t}x_i + \sqrt{1 - \overline{\alpha}_t}\epsilon_i, t)\|^2]$ where $x_i \in \mathbb{R}_{w\times h\times c}, t_i \sim \mathcal{U}\{1, T\}$ and $\epsilon_i \sim \mathcal{N}(0, 1)$. Let $\texttt{clip}_C(v) : v \in \mathbb{R}^p \mapsto \min\left\{1, \frac{C}{\|v\|_2}\right\} \cdot v \in \mathbb{R}^p$ denote the clipping function which re-scales its input to have a maximal $\ell_2$ norm of $C$. For a minibatch $\mathcal{B}$ with $|\mathcal{B}| = B$ samples, the "privatised" minibatch gradient $\widehat{g}$ takes on the form

$$\widehat{g} = \frac{1}{B}\sum_{i\in\mathcal{B}} \texttt{clip}_C\left(\nabla l_i(w)\right) + \frac{\sigma C}{B}\xi,$$

with $\xi \sim \mathcal{N}(0, I_p)$ and $I_p \in \mathbb{R}^{p\times p}$ being the identity matrix. In practice, the choice of noise variance $\sigma > 0$, batch-size $B$ and maximum number of training iterations are constrained by the predetermined privacy budget $(\varepsilon, \delta)$. Crucially, the choice of hyper-parameters can have a large impact on the accuracy of the resulting model, and overall DP-SGD makes it challenging to accurately train deep neural networks. On CIFAR-10 for example, the highest reported test accuracy for a DP model trained with $\epsilon = 8$ was 63.4% in 2021 (Yu *et al.*, 2021). De *et al.* (2022) improved performance on image classification tasks and in particular obtained nearly 95% test accuracy for $\epsilon = 1$ on CIFAR-10, using notably 1) pre-training, 2) large batch sizes, and 3) augmentation multiplicity.

Here, we analyze to what extent these performance gains transfer from DP image classification to DP image generation by training diffusion models with DP-Adam. Diffusion models are inherently different model architectures that exhibit different training dynamics than standard classifiers which introduces additional difficulties in adapting DP training.

# 4 Improvements towards Fine-Tuning Diffusion Models with Differential Privacy

Similarly to Dockhorn *et al.* (2022), we propose to train diffusion models with a DP optimizer. We will now elaborate on how to improve upon the naive application of DP-SGD.

**Recommendations from previous work.** De *et al.* (2022) identify pre-training, large batch sizes, and augmentation

multiplicity as effective strategies in DP image classification. We adopted their recommendations in the training of DP diffusion models, and confirmed the effectiveness of their strategies to the task of DP image generation. In contrast to the work of Dockhorn *et al.* (2022), where the batch-size is only scaled up to 2,048 samples, we implemented virtual batching which helps us to scale to up to 32,768 samples per batch.

**Pre-training.** Pre-training is especially integral to generating realistic image datasets, even if there is a considerable distribution shift between the pre-training and fine-tuning distributions. Unless otherwise specified, we thus pre-train all of our models on ImageNet32 (Chrabaszcz *et al.*, 2017).

**Augmentation multiplicity with timesteps.** De *et al.* (2022) observe that data augmentation, as commonly implemented in public training, has a detrimental effect on the accuracy in DP classification. Instead they propose the use of augmentation multiplicity (Fort *et al.*, 2021). In more detail, they augment each unique training observation within a batch, e.g. with horizontal flips or random crops, and average the gradients of the augmentations before clipping them. Similarly to Dockhorn *et al.* (2022), we extend augmentation multiplicity to also sample multiple timesteps $t$ for each mini-batch observation, and average the corresponding gradients before clipping. In contrast to Dockhorn *et al.* (2022) where only timestep multiplicity is considered, we combine it with traditional augmentation methods, namely random crops and flipping. As a result, while Dockhorn *et al.* (2022) find that the FID plateaus for around 32 augmentations per image, we see increasing benefits in the number augmentation samples (see Figure 8). For computational reasons, we limit augmentation multiplicity to 128 samples.

**Modified timestep sampling.** The training objective for diffusion models in Equation 1 samples the timestep $t$ uniformly from $\mathcal{U}\{0, T\}$ because the model must learn to denoise images at every noise level. However, it is not straightforward that uniform sampling is the best strategy, especially in the DP setting where the number of model updates is limited by the privacy budget. In particular, in the fine-tuning scenario, a pre-trained model has already learned that at small timesteps the task is to remove small amounts of Gaussian noise from a natural-looking image, and at large timesteps the task is to project a completely noisy sample closer to the manifold of natural-looking images. The model behavior at small and large timesteps is thus more likely to transfer to different image distributions without further tuning. In contrast, for medium timesteps the model must be aware of the data distribution at hand in order to compose a natural-looking image. A similar observation has been recently made for membership inference attacks (Carlini *et al.*, 2023; Hu and Pang, 2023): the adversary has been shown to more likely succeed in membership inference when it uses a diffusion model to denoise images with medium amounts of noise compared to high- or low-variance noised images. This motivates us to explore modifications of the training objective where the timestep sampling distribution is not uniform, and instead focuses on training the regimes that contribute more to modelling the key content of an image.

Motivated by this reasoning, we considered replacing the uniform timestep distribution with a mixture of uniform distributions $t \sim \sum_{i=1}^{K} w_i \mathcal{U}\{l_i, u_i\}$ where $\sum_{i=1}^{K} w_i = 1, 0 \leq l_0, u_K \leq T$ and $u_{k-1} \leq l_k$ for $k \in \{2, ..., K\}$. On CIFAR-10, we found the best performance for a distribution with probability mass focused within $\{30, ..., 600\}$ for $T = 1000$ where timesteps outside this interval are assigned a lower probability than timesteps within this interval. We assume this is due to ImageNet-pre-trained diffusion models being able to denoise other (potentially unseen) natural images if only a small amount of noise is added. Training with privacy for small timesteps can then decrease the performance because more of the training budget is allocated to the timesteps that are harder to learn and because of the noisy optimization procedure. Even when training from scratch on MNIST, we observe that focusing the limited training capacity on the harder-to-learn moderate time steps increases performance.

# 5 Empirical Results

## 5.1 Current Evaluation of DP Image Generators

Even though metrics like the FID or the Inception Score have been designed to correlate with the visual quality of the images, they can be misleading since they highly vary with image quality unrelated factors such as the number of observations, or the version number of the inception network (Kynkäänniemi *et al.*, 2022). They also reduce complex image data distributions to single values that might not capture what practitioners are interested in when dealing with DP synthetic data. Most importantly, they may not effectively capture the nuances in image quality the further apart the observed data distribution is from ImageNet.

An alternative way of comparing DP image generation models is by looking at the test performance of a downstream classifier trained on a synthetic dataset of the same size as the real dataset (Dockhorn *et al.*, 2022; Xie *et al.*, 2018), and tested on the real dataset. We argue that the way DP generative models are currently evaluated downstream, i.e. by evaluating a single model or metric on a limited data set, needs to be revisited.

This line of thinking motivates the proposal of an evaluation framework that focuses on *how* DP image generative models are used by practitioners. As these works typically consider marginal distributions, correlative structures, or prediction tasks on tabular data, it is not obvious how to translate their suggestions to the image domain. As such, our experiments focus on two specific use cases for private synthetic image data: *downstream prediction tasks*, and *model selection*. After presenting our results within the evaluation framework that is commonly used in the current literature, we investigate the utility of the DP image diffusion model trained on CIFAR-10 for the aforementioned tasks in section 6.

## 5.2 Experimental Setup

We now evaluate the empirical efficiency of our proposed methods on three image datasets: MNIST, CIFAR-10 and Camelyon17. Please refer to Table 1 for an overview on our main results. For CIFAR-10, we provide additional experiments to prove the utility of the synthetic data for model selection in section 6.

Table 1: A summary of the best results provided in this paper when training diffusion models with DP-SGD. We report the test accuracy of classifiers trained on different data sets. The highest reported current *SOTA* corresponds to classifiers trained on DP synthetic data, as reported in the literature. Here, [Do22] refers to Dockhorn *et al.* (2022), and [Ha22] to Harder *et al.* (2022). Our test accuracy (*Ours*) denotes the accuracy of a classifier trained on a synthetic dataset that was generated by a DP diffusion model and is of the same size as the original training data. Note that we also report the model size of our generative models (*Diffusion M. Size*). The *Non-synth* test accuracy corresponds to the test accuracy of a DP classifier trained on the real dataset, using the techniques introduced by De *et al.* (2022). *[De22] This number is taken from De *et al.* (2022) for $\epsilon = 8$.

| Dataset | Image Resolution | Diffusion M. Size | Pre-Training Data | Test Accuracy (%) | | | Privacy |
|---|---|---|---|---|---|---|---|
| | | | | SOTA | Ours | Non-synth | $(\epsilon, \delta)$ |
| MNIST | $28 \times 28$ | 4.2M | – | 98.1 [Do22] | 98.6 | 99.1 | $(10, 10^{-5})$ |
| CIFAR-10 | $32 \times 32 \times 3$ | 80.4M | ImageNet32 | 51.0 [Ha22] | 88.0 | 96.6 *[De22] | $(10, 10^{-5})$ |
| Camelyon17 | $32 \times 32 \times 3$ | 80.4M | ImageNet32 | - | 91.1 | 90.5 | $(10, 3 \cdot 10^{-6})$ |

We train diffusion models with a U-Net architecture as used by Ho *et al.* (2020) and Dhariwal and Nichol (2021). In contrast to Dockhorn *et al.* (2022), we found that classifier guidance led to a drop in performance. Unless otherwise specified, all diffusion models are trained with a privatized version of Adam (Kingma and Ba, 2014; McMahan *et al.*, 2018). The clipping norm is set to $10^{-3}$. The privacy budget is set to $\epsilon = 10$, as commonly considered in the DP image generation literature. The same architecture is used for the diffusion model across all datasets. More specifically, the diffusion is performed over 1000 timesteps with a linear noise schedule, and the convolutional architecture employs the following details: a variable width with 2 residual blocks per resolution, 192 base channels, 1 attention head with 16 channels per head, attention at the 16x16 resolution, channel multipliers of (1,2,2,2), and adaptive group normalization layers for injecting timestep and class embeddings into residual blocks as introduced by Dhariwal and Nichol (2021). When fine-tuning, the model is pre-trained on ImageNet32 (Chrabaszcz *et al.*, 2017) for 200,000 iterations. All hyperparameters can be found in Table 3.

As a baseline, we also train DP classifiers directly on the sensitive data, using the training pipeline introduced by De *et al.* (2022), and additionally hyperparameter tuning the learning rate. Please refer to Table 5 for more details. It is not surprising that these results partly outperform the image generators as the training of the DP classifiers is targeted towards maximising downstream performance.

### 5.3 Training from Scratch ($\emptyset \rightarrow$ MNIST)

The MNIST dataset (LeCun, 1998) consists of 60,000 $28 \times 28$ training images depicting the 10 digit classes in grayscale. It is the most commonly used dataset in the DP image generation literature, and included here for completeness.

**Method.** We use a DP diffusion model of 4.2M parameters without any pre-training, with in particular 64 channels, and channel multipliers (1,2,2). The diffusion model is trained for 4,000 iterations at a constant learning-rate of $5 \cdot 10^{-4}$ at batch-size 4,096, with 128 augmentation samples, and a noise multiplier of 2.852. The timesteps are sampled uniformly within $\{0, \ldots, 200\}$ with probability 0.05, within $\{200, \ldots, 800\}$ with probability 0.9, and within $\{800, \ldots, 1000\}$ with probability 0.05. To evaluate the quality of the images, we generate 50,000 samples from the diffusion model. Then we train a WRN-40-4 on these synthetic images (hyperparameters given in Table 4), and evaluate it on the MNIST test set.

**Results.** As reported in Table 1, this yields a state-of-the-art top-1 accuracy of 98.6%, to be compared to the 98.1% accuracy obtained by Dockhorn *et al.* (2022). Crucially, we find that to obtain this SOTA result, it is important to bias the timestep sampling of the diffusion model at training time: this allows the model to get more signal from the challenging phases of the diffusion generation process. Without this biasing, we obtain a classification accuracy of only 98.2%.

### 5.4 Fine-tuning on a Medical Application (ImageNet32 $\rightarrow$ Camelyon17)

To show that fine-tuning works even in settings characterised by dataset shift from the pre-training distribution, we fine-tune a DP diffusion model on a medical dataset. Camelyon17 (Bandi *et al.*, 2018; Koh *et al.*, 2021) comprises 455,954 histopathological $96 \times 96$ image patches of lymph node tissue from five different hospitals. The label signifies whether at least one pixel in the center $32 \times 32$ pixels has been identified as a tumor cell. Camelyon17 is also part of the WILDS (Koh *et al.*, 2021) leaderboard as a domain generalization task: The training dataset contains 302,436 images from three different hospitals whereas the validation and test set contain respectively 34,904 and 85,054 images from a fourth and fifth hospital. Since every hospital uses a different staining technique, it is easy to overfit to the hues of the training data with empirical risk minimisation. At the time of writing, the highest accuracy reported in the leaderboard of official submissions is 92.1% (Gao *et al.*, 2022). Among the unofficial submissions, the SOTA achieves 96.5% by pre-training on a large web image data set (Kumar *et al.*, 2022).

**Method.** First we pre-train an image diffusion model on ImageNet32, before finetuning it with $(10, 3 \cdot 10^{-6})$-DP on the training data, downsampled to $32 \times 32$, with a batch size of 16,384 for 200 steps. We tuned the hyperparameters to achieve the lowest FID on the training data, and used the out-of-distribution validation data to tune the downstream classifiers. The timestep is sampled with 0.015 probability from $\{0, \ldots, 30\}$, with a probability of 0.785 in $\{30, \ldots, 600\}$,

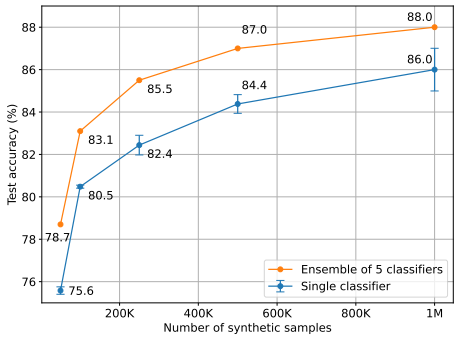



Figure 2: Top-1 accuracy of a CIFAR-10 WRN-40-4 as function of the number of synthetic data samples used to train it.

and with 0.2 in $\{600, \ldots, 1000\}$. Since the diffusion model is pre-trained on ImageNet, we assume that the data is also available for pre-training the classifier. The pre-trained classifier is fine-tuned on a synthetic dataset of the same size as the original training dataset. The classifiers were trained on augmented data where the augmentations include flipping, and color-jittering.

**Results.** We achieve close to state-of-the-art classification performance with 91.1% by training only on the synthetic data whereas the best DP classifier we trained on the real dataset achieved only 90.5%. So while the synthetic dataset is not only useful for in-distribution classification, training on synthetic data is also an effective strategy to combat overfitting to domain artifacts and generalise in out-of-distribution classification tasks, as noted elsewhere in the literature (Gheorghiță *et al.*, 2022; Zhou *et al.*, 2020).

### 5.5 Fine-tuning on Natural Images (ImageNet32 $\rightarrow$ CIFAR-10)

CIFAR-10 (Krizhevsky *et al.*, 2009) is a natural image dataset of 10 different classes with 50,000 RGB images of size $32 \times 32$ during training time.

**Method.** We use the same pre-trained model as before, and tune the remaining hyperparameters by splitting the training data into a set of 45,000 images for training, and 5,000 images for validation based on FID. As for Camelyon17, sampling the timestep with probability 0.15 in $\{0, \ldots, 30\}$, with 0.785 in $\{30, \ldots 60\}$ and 0.2 in $\{600, \ldots, 1000\}$ gives us the lowest FID (see Supplement Table 3).

**Results.** We improve the SOTA FID with ImageNet pre-training (Harder *et al.*, 2022) from 26.8 to 9.8, which is a drop of more than 50%, and increase the downstream accuracy from 51.0% to 75.6% without pre-training the classifier. With pre-training the classifier on ImageNet32, we can achieve a classification accuracy of 86.6% with a single WRN. Modifying the timestep distribution led to a reduction in the FID from 11.6 to 9.8. Note that we achieved the results for different privacy levels by linearly scaling the number of iterations proportionally with $\epsilon$, and adjusting the noise level to the given privacy budget, keeping all parameters the same.

As detailed in Supplement **??**, we obtain state-of-the-art accuracy for a variety of privacy levels. Even for a budget as small as $\epsilon = 1$, the FID obtained with our method is smaller than the current SOTA for $\epsilon = 10$. These results can also be compared with the very recent work by Dockhorn *et al.* (2022), who report an FID of 97.7 when training diffusion models with differential privacy on CIFAR-10 without any pre-training. Due to the difficulty of the task of learning the diffusion model with differential privacy from scratch, the model did not learn to generate CIFAR-10 like samples, and the generated images do not seem to display any clear CIFAR-10 class instances at all. We believe that such mixed results are a clear motivation for our proposed method of pre-training on public data, which make the learning problem significantly more tractable and realistic, and allows to obtain useful image generation.

To further confirm that the diffusion model has correctly learned the distribution shift, we trained a ResNet18 model to discriminate images from CIFAR-10 and ImageNet32 achieving a test accuracy of 98.0% on that task. We then evaluated it on 50,000 synthetically generated images out of which 92% were classified as CIFAR-10 images. This supports our hypothesis that the fine-tuned diffusion model does generate images that are more similar to CIFAR-10 than to the pre-training data of ImageNet.

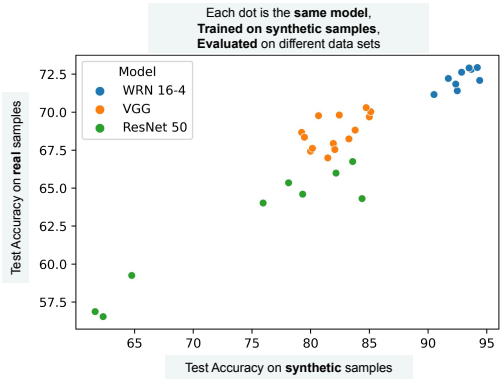
### 5.6 Maximizing Downstream Prediction Performance by Sampling Arbitrary Many Data Records (ImageNet32 $\rightarrow$ CIFAR-10)

**Dataset sample size.** One benefit of synthetic data generators is their ability to render infinitely many synthetic images. As such, there is no reason why the comparison of real and synthetic samples should be limited to predictive models trained on the same number of training samples. We, therefore, investigate whether the performance of a downstream predictor increases with more training images. In Figure 2, we observe that the downstream classification accuracy constantly increases the more synthetic training observations are generated. In particular, we increase the downstream classification accuracy from 72.9% to 86.0% by sampling 1M instead of 50K images – without pretraining the classifier. We note that this difference is much more significant on the more challenging dataset of CIFAR-10 than e.g. MNIST, where we find that increasing the number of samples offers virtually no benefit in terms of downstream accuracy.

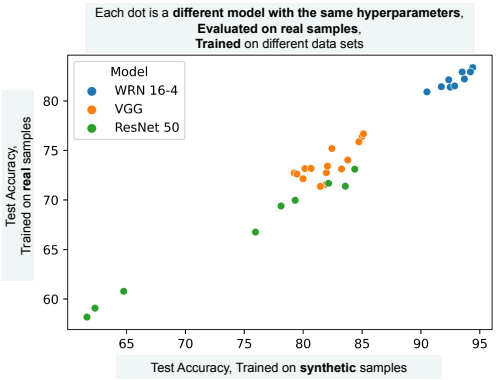
**Ensembling.** We observe that we can further improve the classification accuracy given by a single WRN classifier by instead ensembling five different networks that differ only in the subsampling of the minibatches. As reported in Table 1, we can achieve a test accuracy of 88% on CIFAR-10.

## 6 Model Selection (ImageNet32 $\rightarrow$ CIFAR-10)

One important benefit of training a DP image generator over a DP classifier is the potential to use the generated data repeatedly for training a range of different prediction models and choosing the best one across them. Each experiment training a model on the data comes with a privacy cost, thus tuning a large number of DP classifiers increases the required privacy budget (Papernot and Steinke, 2021). In this section we consider how synthetic data can be used to gain initial insights




(a) Test accuracy on synthetic data vs real data of models trained on synthetic data




(b) Test accuracy of models trained and evaluated on either synthetic or real data

Figure 3: Models rank similarly when evaluated on synthetic and real data. This suggests that findings on hyperparameter selection made on synthetic data transfer to the private data.

on the choice of model, and to reduce the number of experiments run on the sensitive data records. The goal here is to identify the model that performs best on real data, while only having access to synthetic data. This becomes particularly relevant and useful when synthetic data needs to be released for research purposes, or data challenges.

For this purpose, we train 3 different model architectures that are commonly employed on CIFAR-10 (Bu *et al.*, 2022; De *et al.*, 2022): a WRN-28-8 (Zagoruyko and Komodakis, 2016), a ResNet50 (He *et al.*, 2016), and a VGG (Simonyan and Zisserman, 2014). For each architecture, we sweep over combinations of three to five different learning rates and three different values of weight decay. Please refer to Table 6 for more details. We now assess the utility of synthetic data for model selection in two stages of increasing difficulty. First, we check whether models – trained on the synthetic data – rank similarly on the real and synthetic test data. This corresponds to the application setting where a third party tunes a model on the synthetic data, and releases a model that is trained on the same data.

Once we have established that the test performance on real and synthetic data is sufficiently correlated to ensure that a model ranks similarly no matter which data set it was evaluated on, we train each model separately on 50K real and on the same number of synthetic samples. In both cases, models are tested on the same source of data they have been learned on (with sources being real or synthetic here). We then assess whether the test performance on the real and synthetic data is still correlated between models of the same architecture and hyperparameter constellation. This corresponds to the setting where a third party is responsible for finding the best model pipeline, but the data curator trains the final model with the optimal hyperparameters for their own internal use.

In Figure 3, we report our findings. We see that we can select the best or second best hyperparameter constellation in both application settings. More generally, we find that models rank similarly on both synthetic and real data, suggesting that findings with respect to relative model performance might transfer from the DP data to the original real data. However, we also notice that models overfit to the synthetic data distribution and that within one model group it is not obvious which hyperparameter constellation is the best. We therefore advice that synthetic data is used for a high-level orientation of the research direction.

## 7 Conclusion

DP image generation has long attracted interest as a way of sharing synthetic data sets in sensitive application domains. Because of the degradation in performance introduced by DP-SGD, successful results on DP image generation have been limited to small and low-complexity data sets, like MNIST. In this paper, we set out to scale DP image generation to $32 \times 32 \times 3$ RGB image datasets. We proposed a methodology for DP diffusion models based on pre-training, timestep augmentation multiplicity, and a modified timestep sampling scheme. We are the first to train a DP image generator on a medical dataset where we achieved a downstream classification accuracy of 91.1% that is close to the SOTA of 96.5% with training on the real data. What is more, we also increased the SOTA downstream classification accuracy on CIFAR-10 from 51.0% to 88.0%. Recently proposed methods like latent diffusion models (Rombach *et al.*, 2022) constitute a promising model class for DP fine-tuning on higher dimensional datasets, and we hope that our findings can contribute to future research exploring this direction.

Finally, we questioned how DP synthetic image data has been currently evaluated in the computer vision literature, and proposed an evaluation framework that is more suited to the needs of practitioners who would use the DP synthetic data as a replacement of the private dataset. For this purpose, we first considered maximising the downstream prediction performance by generating up to 1M data samples, and training ensembles. Second, we showed that findings from hyperparameter tuning on synthetic data translate to the corresponding findings on the real data.

While the proposed methodology achieves SOTA results on multiple benchmarks, it comes with a computational overhead. The large batch sizes, augmentation multiplicity, the increased model sizes, on top of the overhead introduced by DP-SGD, lead to longer training runs. Even though the run time is reduced by employing pretrained models (which leads to fewer training iterations), the models might not be be deployable in practise if not enough compute is available. We leave the challenge of reducing the computational requirements to future work.

## References


Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, pages 308–318, 2016.

Peter Bandi, Oscar Geessink, Quirine Manson, Marcory Van Dijk, Maschenka Balkenhol, Meyke Hermsen, Babak Ehteshami Bejnordi, Byungjae Lee, Kyunghyun Paeng, Aoxiao Zhong, et al. From detection of individual metastases to classification of lymph node status at the patient level: the camelyon17 challenge. *IEEE Transactions on Medical Imaging*, 2018.

Zhiqi Bu, Jialin Mao, and Shiyun Xu. Scalable and efficient training of large convolutional neural networks with differential privacy. *arXiv preprint arXiv:2205.10683*, 2022.

Nicholas Carlini, Jamie Hayes, Milad Nasr, Matthew Jagielski, Vikash Sehwag, Florian Tramèr, Borja Balle, Daphne Ippolito, and Eric Wallace. Extracting training data from diffusion models, 2023.

Pierre Chambon, Christian Bluethgen, Curtis P Langlotz, and Akshay Chaudhari. Adapting pretrained vision-language foundational models to medical imaging domains. *arXiv preprint arXiv:2210.04133*, 2022.

Dongjie Chen, Sen-ching Samson Cheung, Chen-Nee Chuah, and Sally Ozonoff. Differentially private generative adversarial networks with model inversion. In *2021 IEEE International Workshop on Information Forensics and Security (WIFS)*, pages 1–6. IEEE, 2021.

Jia-Wei Chen, Chia-Mu Yu, Ching-Chia Kao, Tzai-Wei Pang, and Chun-Shien Lu. Dpgen: Differentially private generative energy-guided network for natural image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8387–8396, 2022.

Patryk Chrabaszcz, Ilya Loshchilov, and Frank Hutter. A downsampled variant of imagenet as an alternative to the cifar datasets. *arXiv preprint arXiv:1707.08819*, 2017.

Soham De, Leonard Berrada, Jamie Hayes, Samuel L Smith, and Borja Balle. Unlocking high-accuracy differentially private image classification through scale. *arXiv preprint arXiv:2204.13650*, 2022.

Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in Neural Information Processing Systems*, 34:8780–8794, 2021.

Tim Dockhorn, Tianshi Cao, Arash Vahdat, and Karsten Kreis. Differentially private diffusion models. *arXiv preprint arXiv:2210.09929*, 2022.

Jinhao Duan, Fei Kong, Shiqi Wang, Xiaoshuang Shi, and Kaidi Xu. Are diffusion models vulnerable to membership inference attacks?, 2023.

Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of cryptography conference*, pages 265–284. Springer, 2006.

Stanislav Fort, Andrew Brock, Razvan Pascanu, Soham De, and Samuel L Smith. Drawing multiple augmentation samples per image during training efficiently decreases test error. *arXiv preprint arXiv:2105.13343*, 2021.

Irena Gao, Shiori Sagawa, Pang Wei Koh, Tatsunori Hashimoto, and Percy Liang. Out-of-distribution robustness via targeted augmentations. In *NeurIPS 2022 Workshop on Distribution Shifts: Connecting Methods and Applications*, 2022.

Bogdan A Gheorghiță, Lucian M Itu, Puneet Sharma, Constantin Suciu, Jens Wetzl, Christian Geppert, Mohamed Ali Asik Ali, Aaron M Lee, Stefan K Piechnik, Stefan Neubauer, et al. Improving robustness of automatic cardiac function quantification from cine magnetic resonance imaging using synthetic image data. *Scientific reports*, 12(1):1–12, 2022.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020.

Frederik Harder, Kamil Adamczewski, and Mijung Park. Dpmerf: Differentially private mean embeddings with randomfeatures for practical privacy-preserving data generation. In *International conference on artificial intelligence and statistics*, pages 1819–1827. PMLR, 2021.

Fredrik Harder, Milad Jalali Asadabadi, Danica J Sutherland, and Mijung Park. Differentially private data generation needs better features. *arXiv preprint arXiv:2205.12900*, 2022.

Jamie Hayes, Luca Melis, George Danezis, and Emiliano De Cristofaro. Logan: evaluating privacy leakage of generative models using generative adversarial networks. *arXiv preprint arXiv:1705.07663*, pages 506–519, 2017.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

Benjamin Hilprecht, Martin Härterich, and Daniel Bernau. Monte carlo and reconstruction membership inference attacks against generative models. *Proc. Priv. Enhancing Technol.*, 2019(4):232–249, 2019.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.

Hailong Hu and Jun Pang. Membership inference of diffusion models, 2023.

Dihong Jiang, Guojun Zhang, Mahdi Karami, Xi Chen, Yunfeng Shao, and Yaoliang Yu. Dp $^2$-vae: Differentially private pre-trained variational autoencoders. *arXiv preprint arXiv:2208.03409*, 2022.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Pang Wei Koh, Shiori Sagawa, Henrik Marklund, Sang Michael Xie, Marvin Zhang, Akshay Balsubramani, Weihua Hu, Michihiro Yasunaga, Richard Lanas Phillips, Irena Gao, Tony Lee, Etienne David, Ian Stavness, Wei Guo, Berton A. Earnshaw, Imran S. Haque, Sara Beery, Jure Leskovec, Anshul Kundaje, Emma Pierson, Sergey Levine, Chelsea Finn, and Percy Liang. WILDS: A benchmark of in-the-wild distribution shifts. In *International Conference on Machine Learning*, 2021.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. *Pre-print*, 2009.

Ananya Kumar, Ruoqi Shen, Sébastien Bubeck, and Suriya Gunasekar. How to fine-tune vision models with sgd. *arXiv preprint arXiv:2211.09359*, 2022.

Tuomas Kynkäänniemi, Tero Karras, Miika Aittala, Timo Aila, and Jaakko Lehtinen. The role of imagenet classes in fréchet inception distance. *arXiv preprint arXiv:2203.06026*, 2022.

Yann LeCun. The mnist database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*, 1998.

Andreas Lugmayr, Martin Danelljan, Radu Timofte, Kangwook Kim, Younggeun Kim, Jae-young Lee, Zechao Li, Jinshan Pan, Dongseok Shim, Ki-Ung Song, et al. Ntire 2022 challenge on learning the super-resolution space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 786–797, 2022.

H Brendan McMahan, Galen Andrew, Ulfar Erlingsson, Steve Chien, Ilya Mironov, Nicolas Papernot, and Peter Kairouz. A general approach to adding differential privacy to iterative training procedures. *arXiv preprint arXiv:1812.06210*, 2018.

Nicolas Papernot and Thomas Steinke. Hyperparameter tuning with renyi differential privacy. *arXiv preprint arXiv:2110.03620*, 2021.

Bjarne Pfitzner and Bert Arnrich. Dpd-fvae: Synthetic data generation using federated variational autoencoders with differentially-private decoder. *arXiv preprint arXiv:2211.11591*, 2022.

Pavlo M Radiuk. Impact of training set batch size on the performance of convolutional neural networks for diverse datasets. *Information Technology and Management Science*, 20(1):20–24, 2017.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10684–10695, 2022.

Pouria Rouzrokh, Bardia Khosravi, Shahriar Faghani, Mana Moassefi, Sanaz Vahdati, and Bradley J Erickson. Multitask brain tumor inpainting with diffusion models: A methodological report. *arXiv preprint arXiv:2210.12113*, 2022.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, pages 2256–2265. PMLR, 2015.

Gowthami Somepalli, Vasu Singla, Micah Goldblum, Jonas Geiping, and Tom Goldstein. Diffusion art or digital forgery? investigating data replication in diffusion models. *arXiv preprint arXiv:2212.03860*, 2022.

Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in Neural Information Processing Systems*, 32, 2019.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020.

Theresa Stadler and Carmela Troncoso. Why the search for a privacy-preserving data sharing mechanism is failing. *Nature Computational Science*, 2(4):208–210, 2022.

Theresa Stadler, Bristena Oprisanu, and Carmela Troncoso. Synthetic data–anonymisation groundhog day. *arXiv preprint arXiv:2011.07018*, 2021.

Reihaneh Torkzadehmahani, Peter Kairouz, and Benedict Paten. Dp-cgan: Differentially private synthetic data and label generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0, 2019.

Di Wang, Minwei Ye, and Jinhui Xu. Differentially private empirical risk minimization revisited: Faster and more general. *Advances in Neural Information Processing Systems*, 30, 2017.

Liyang Xie, Kaixiang Lin, Shu Wang, Fei Wang, and Jiayu Zhou. Differentially private generative adversarial network. *arXiv preprint arXiv:1802.06739*, 2018.

Jinsung Yoon, Lydia N Drumright, and Mihaela Van Der Schaar. Anonymization through data synthesis using generative adversarial networks (ads-gan). *IEEE journal of biomedical and health informatics*, 24(8):2378–2388, 2020.

Da Yu, Huishuai Zhang, Wei Chen, Jian Yin, and Tie-Yan Liu. Large scale private learning via low-rank

reparametrization. In *International Conference on Machine Learning*, pages 12208–12218. PMLR, 2021.

Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016.

Xiaojin Zhang, Hanlin Gu, Lixin Fan, Kai Chen, and Qiang Yang. No free lunch theorem for security and utility in federated learning. *arXiv preprint arXiv:2203.05816*, 2022.

Kaiyang Zhou, Yongxin Yang, Timothy Hospedales, and Tao Xiang. Deep domain-adversarial image generation for domain generalisation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 13025–13032, 2020.

## Supplementary Materials

## A Comparison to Related Work

Please refer to Table 2 for a summary of details that differentiate our work from that of Dockhorn *et al.* (2022).

Table 2: Comparison of Our Work and Dockhorn *et al.* (2022).

| | **Dockhorn *et al.* (2022)** | **Ours** |
|---|---|---|
| Analysed for finetuning | False | True |
| Maximal parameter count | 1.8M | 80.4M |
| Time step scale | continuous | discrete |
| Classifier guidance | True | False |
| Modifications to time step sampling | False | True |
| Augmentation strategies | time step | time step, random crop, flipping |
| Number of samples for augmentation multiplicity | 32 | 128 |
| Maximal batch size | 2,048 | 16,384 |
| Evaluation | FID, downstream accuracy | FID, downstream accuracy, maximal downstream performance hyperparameter tuning |

## B Additional Experimental Details and Results

The code for the DP accounting and the classification models can be found on https://github.com/deepmind/jax_privacy. The implementation of the diffusion models follows https://github.com/openai/guided-diffusion. We present our hyperparameters in Tables 3, 4, 5 and 6.

Example DP synthetic images can be found in Figures 4, 5 and 6. As a baseline, we also added samples from Dockhorn *et al.* (2022) when training from scratch on CIFAR-10 in Figure 7. This illustrates the importance of pre-training for large scale DP image generation. Please refer to Figure 8 for a sensitivity study of augmentation multiplicity. Finally, we show with class-wise metrics in Figure 10 that our model is not overfitting to a single class.

**Details on evaluation metrics.** To evaluate the generative performance of DP diffusion models, we estimate the FID and the downstream accuracy. For the former, we follow Dhariwal and Nichol (2021). For the latter, we generate as many samples from the diffusion model as the original private training data has, unless otherwise specified. We then train a classifier on the synthetic data set and report its test performance on the real test data. For the model selection experiments in section 6, we follow a similar procedure. Note that we here train and evaluate the downstream classifiers on the real or the synthetic data, as specified.

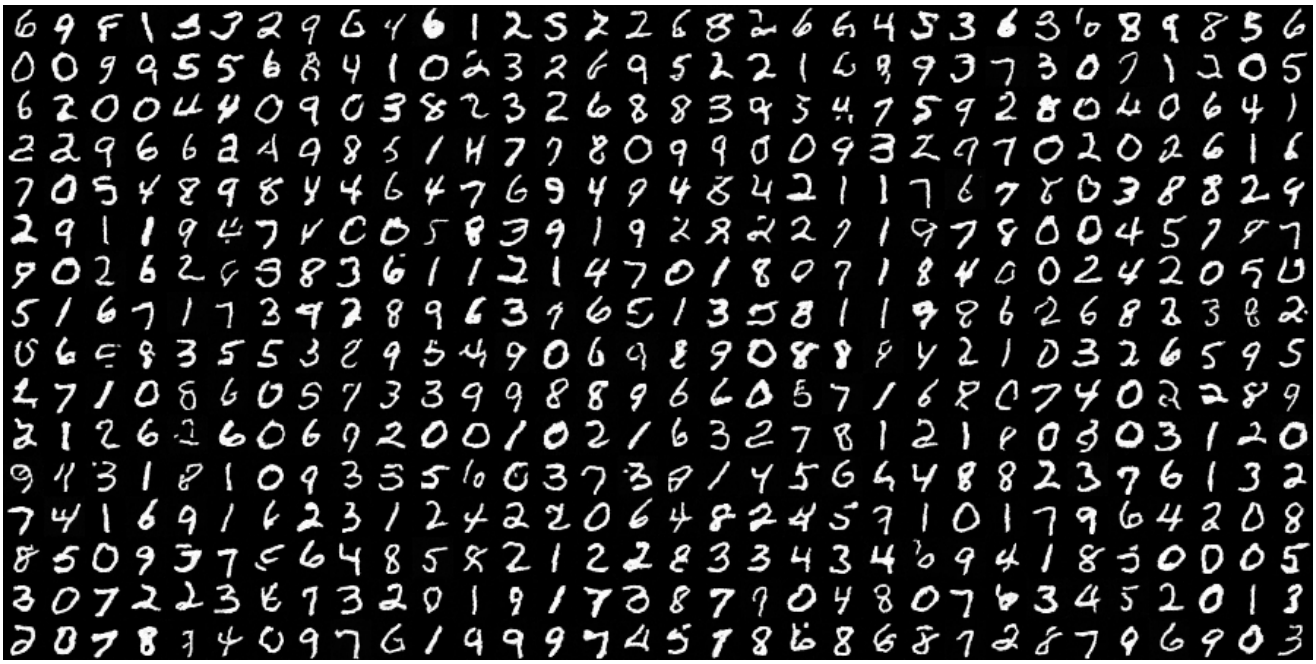


Figure 4: Random samples drawn from a DP image diffusion model trained on MNIST.

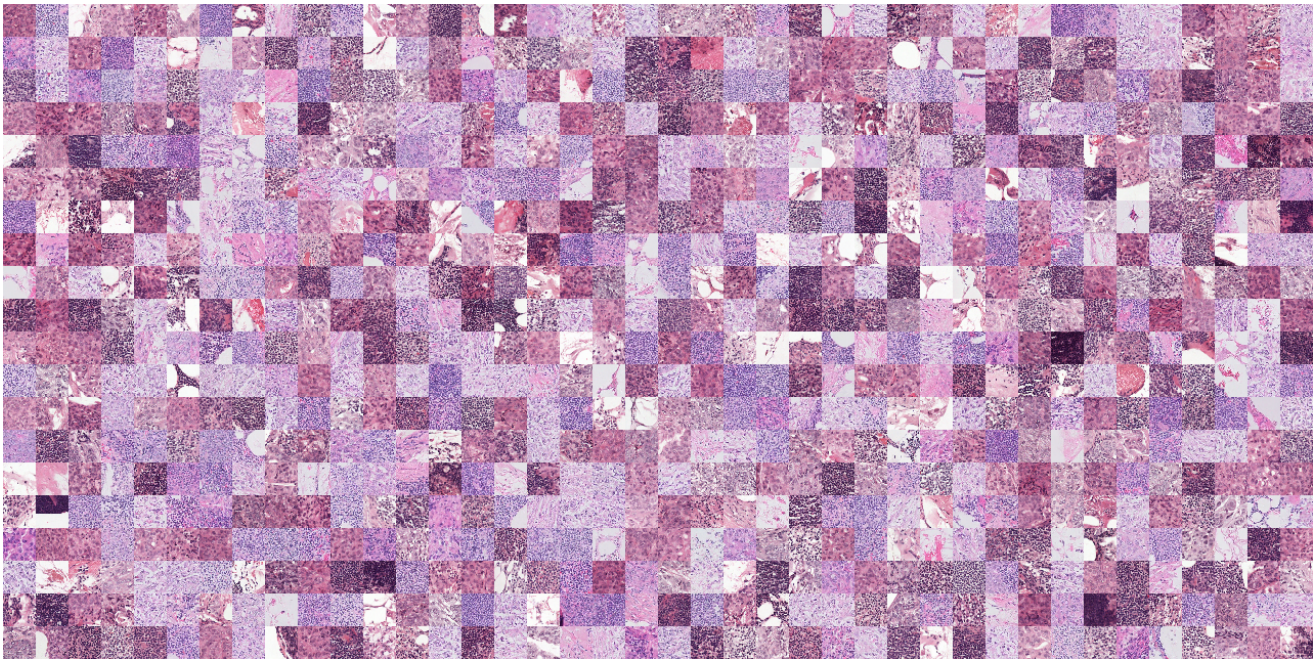


Figure 5: Random samples drawn from a DP image diffusion model trained on Camelyon17.

Figure 6: Random samples drawn from a DP image diffusion model trained on CIFAR-10.

Figure 7: Samples from CIFAR-10 as provided by (Dockhorn *et al.*, 2022) in Appendix F Rebuttal Discussions.

Table 3: Hyperparameters for Diffusion Models.

| | ImageNet32 | MNIST | CIFAR-10 | Camelyon17 |
|---|---|---|---|---|
| Pre-training data set | - | - | ImageNet32 | ImageNet32 |
| Privacy budget $(\epsilon, \delta)$ | $\infty$ | (10, $10^{-5}$) | (*varies, $10^{-5}$) | (10, $10^{-5}$) |
| Iterations | 200k | 4,000 | 200 | 200 |
| Clipping norm | - | $10^{-3}$ | $10^{-3}$ | $10^{-3}$ |
| Noise schedule | linear | linear | linear | linear |
| Model size | 80.4M | 4.2M | 80.4M | 80.4M |
| Channels | 192 | 64 | 192 | 192 |
| Depth | 2 | 1 | 2 | 2 |
| Channels multiple | 1,2,2,2 | 1,2,2 | 1,2,2,2 | 1,2,2,2 |
| Heads channels | 64 | 64 | 64 | 64 |
| Attention resolution | 16 | 16 | 16 | 16 |
| Batch size | 1,024 | 4,096 | 16,384 | 16,384 |
| Learning rate | - | $5\times10^{-4}$ | $10^{-3}$ | $10^{-3}$ |
| Optimizer | Adam | Adam | Adam | Adam |
| Scheduler | linear$_{\text{(from 0 to LR in 5K steps)}}$ | constant | constant | constant |
| $w_1, w_2, w_3$ | 0.03, 0.77, 0.2 | 0.05, 0.9, 0.05 | 0.015, 0.785, 0.2 | 0.015, 0.785, 0.2 |
| $l_1, u_1 = l_2, u_2 = l_3, u_3$ | 0, 30, 800, 1000 | 0, 200, 800, 1000 | 0, 30, 600, 1000 | 0, 30, 600, 1000 |
| # Augmentation samples | 0 | 128 (timestep) | 128 (timestep, flip) | 128 (timestep, flip) |
| Exponential moving average | 0.9999 | 0.9999 | 0.9999 | 0.9999 |

The scale of the gradient noise is adjusted to ensure the desired privacy budget. *We report results for $\epsilon \in \{1, 5, 10, 32\}$. The implementation follows https://github.com/openai/guided-diffusion

Table 4: Hyperparameters for Downstream Classification WRNs Trained on Synthetic Data.

| | MNIST | CIFAR-10 | Camelyon17 |
|---|---|---|---|
| Pre-training data set | - | - | ImageNet32 |
| Iterations | 10,000 | 20,000 | 4,000 |
| Depth | 40 | 40 | 40 |
| Width | 4 | 4 | 4 |
| Dropout | 0.5 | 0.0 | 0.0 |
| Weight decay | $5 \cdot 10^{-4}$ | $5 \cdot 10^{-4}$ | $1 \cdot 10^{-5}$ |
| Label smoothing | 0.05 | 0.0 | 0.0 |
| Batch size | 64 | 4,096 | 512 |
| Learning rate | 0.1 | cosine decay$_{\text{(start at 0.1 with } \alpha = 0\text{, 1 decay step)}}$ | $10^{-4}$ |
| Optimizer | SGD | SGD | SGD |
| Nesterov's momentum | 0.9 | 0.9 | 0.9 |
| Samples augmentation multiplicity | 0 | 0 | 16 |
| Exponential moving average | 0.9999 | 0.9999 | 0.9999 |
| # Augmentation samples | 1 (crop) | 16 (crop, flip, color) | 16 (color, flip, crop) |

Table 5: Hyperparameters for DP WRN Classifiers Trained on the Sensitive Data Records. Training was Stopped Once $\epsilon = 10$ Was Reached.

| | MNIST | Camelyon17 |
|---|---|---|
| Privacy budget $(\epsilon, \delta)$ | (10, $10^{-5}$) | (10, $10^{-5}$) |
| Pre-training data set | - | ImageNet32 |
| Clipping norm | 1 | 1 |
| Noise standard deviation | 3.0 | 4.0 |
| Depth | 16 | 40 |
| Width | 4 | 4 |
| Batch size | 16,384 | 4,096 |
| Learning rate | 4.0 | 0.5 |
| Optimizer | SGD | SGD |
| Samples augmentation multiplicity | 0 | 16 (color, flip, crop) |
| Exponential moving average | 0.9999 | 0.9999 |

Table 6: Hyperparameters for Classifiers for the Model Selection Experiment.

| | ResNet50 | VGG | WRN-16-4 |
|---|---|---|---|
| Learning rate | $5 \cdot 10^{-4}, 2 \cdot 10^{-3}, 4 \cdot 10^{-3}$ | $0.07, 0.04, 0.025, 0.01, 5 \cdot 10^{-3}$ | $0.01, 0.02, 0.03$ |
| Weight decay | $0, 0.1, 0.01$ | $0.0, 10^{-3}, 5 \cdot 10^{-3}$ | $0.0, 0.1, 0.01$ |
| Iterations | 15,000 | 15,000 | 15,000 |
| Label smoothing | 0.05 | 0.0 | 0.0 |
| Batch size | 128 | 128 | 128 |
| Optimizer | SGD | SGD | SGD |
| Momentum | 0.0 | 0.9 | 0.9 |
| Scheduler | cosine decay* | constant | cosine decay* |
| Samples augmentation multiplicity | 16 (flip, crop) | 0 | 16 (flip, crop) |
| Exponential moving average | 0.9999 | 0.9999 | 0.9999 |

All combinations of different values displayed for learning rate and weight decay were trained. *Cosine decay schedule with initial learning rate swept over, 1 decay step, and $\alpha = 0.0$.

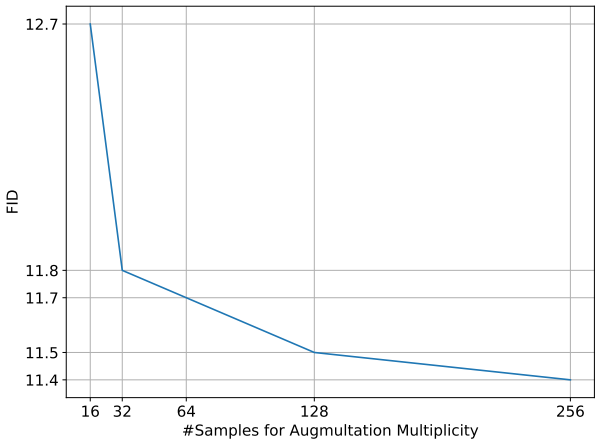



Figure 8: FID on CIFAR-10 for different values of augmentation multiplicity samples. We see that the FID is decreasing for values up to 256.

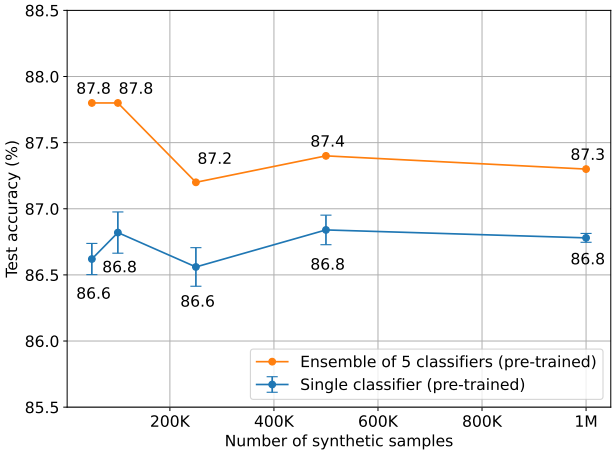
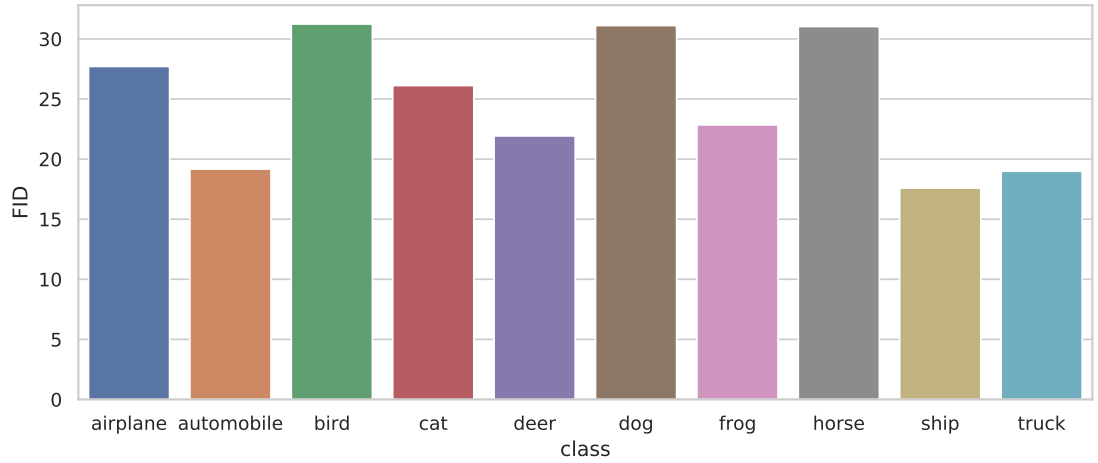
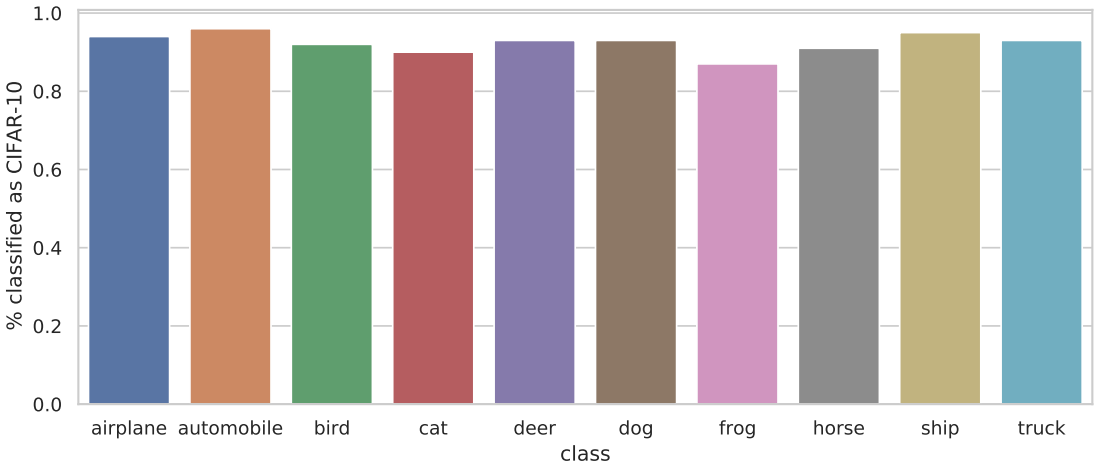



Figure 9: Downstream accuracy of an ImageNet32 pre-trained CIFAR-10 WRN-40-4 as function of the number of synthetic data samples used to train it.




(a) Percentage of samples per class classified as CIFAR-10 images by ResNet18 model trained to discriminate CIFAR-10 and ImageNet32 images.




(b) FID per class.

Figure 10: Class-wise metrics on CIFAR-10.

## Statement of Authorship for joint/multi-authored papers for PGR thesis

To appear at the end of each thesis chapter submitted as an article/paper

The statement shall describe the candidate's and co-authors' independent research contributions in the thesis publications. For each publication there should exist a complete statement that is to be filled out and signed by the candidate and supervisor **(only required where there isn't already a statement of contribution within the paper itself).**

| | |
|---|---|
| Title of Paper | Differentially private diffusion models generate useful synthetic images |
| Publication Status | □Published □ Accepted for Publication<br>xSubmitted for Publication □Unpublished and unsubmitted work written in a manuscript style |
| Publication Details | Ghalebikesabi, S., Berrada, L., Gowal, S., Ktena, I., Stanforth, R., Hayes, J., ... & Balle, B. (2023). Differentially private diffusion models generate useful synthetic images. arXiv preprint arXiv:2302.13861. |

### Student Confirmation

| | | | |
|---|---|---|---|
| Student Name: | Sahra Ghalebikesabi | | |
| Contribution to the Paper | I conceived part of the methodology, part of the implementation and experiments. The idea was conceived by collaborators. During frequent meetings, my collaborators helped out a lot with helpful suggestions on methodology and feedback. They also contributed to writing, checking the results, and proof-reading. | | |
| Signature |  | Date | 28.10.2023 |

### Supervisor Confirmation

By signing the Statement of Authorship, you are certifying that the candidate made a substantial contribution to the publication, and that the description described above is accurate.

| | | |
|---|---|---|
| Supervisor name and title: Chris Holmes | | |
| Supervisor comments<br><br>I am in agreement with the description of the contributions. | | |
| Signature | Date | 28.10.2023 |

This completed form should be included in the thesis, at the end of the relevant chapter.

# 7 Conclusion and Discussion

## Contents



## 7.1 Discussion

Synthetic data generators impact society's perception of artificial intelligence, for instance in the form of chatbots or image generators. It is thus important to make sure that these models are robust to misspecification or can provide privacy when handling sensitive datasets. Throughout the different chapters of this work, we have explored different ways to improve synthetic data generation under model misspecification and DP. We have described contributions that have potential to be relevant across a wide range of misspecified or DP generative model classes. In particular, we have shown the efficacy of the following techniques:

- In Chapter 3, we have shown that relaxing modelling assumptions in nonparametric Bayesian models helps improve the performance of density estimators fitted on finite datasets.

- Similarly, in Chapter 4, we have provided another example of improving the model fit of synthetic data generators under the presence of missing data by redefining the SDGP to explicitly model the missingness mechanism.

- Starting from Chapter 5, we have focused on improving synthetic data generation under DP. We have analysed the performance of DP importance weighting for improving downstream tasks performed on DP synthetic data. We have shown that importance weighting can help under the presence of DP, but that its performance is limited by the quality of the importance weights and generative models.

- Finally, in Chapter 6, we have identified training techniques that help improve DP image generation. With the proposed approaches, we can decrease the state-of-the-art Fréchet inception distance for DP image generation on CIFAR-10 from 26.8 to 9.8.

## 7.2 Limitations

In this section, we illustrate the limitations of the work presented in this thesis.

**Limitations related to generalisability** First and foremost, the contributions of neither Chapters 3 nor 4 guarantee the DGP to be learned correctly in finite datasets. While we generalise the nonparametric form of the DPMM specification by Fong et al. [2021] in Chapter 3, the resulting density estimator still struggles on highly complex data distributions. Similarly, the missingness models specified in Chapter 4 exhibit a fixed parametric form and will not generalise to arbitrary missingness mechanisms.

**Limitations related to scalability** Second, many interesting applications in health care or image generation are characterised by high-dimensional feature spaces. As we only consider images up to 96×96 pixels (or even smaller tabular data samples), it is not obvious how and if the proposed approaches will scale

to high-dimensional datasets. For instance, the large batch sizes proposed in Chapter 6 increase the computational complexity during training considerably, particularly as the feature space gets larger. Additionally, neural autoregressive density estimators still outperform AR-BP on high-dimensional feature spaces or datasets with large sample sizes. Similarly, importance weighting, as studied in Chapter 5, is known to not scale well to high-dimensional feature spaces [Tokdar and Kass, 2010]. While AR-BP, as presented in Chapter 3, does not increase the computational complexity of R-BP [Fong et al., 2021], R-BP already scales poorly in the number of training observations.

**Increased variance and privacy budget due to importance weighting** Third, importance weighting also increases the variance of the importance weighted estimators [Tokdar and Kass, 2010]. Even if importance weighting can decrease the bias in downstream estimators, the inflated variance and the partial use of the privacy budget introduce additional complexity that the data holder has to carefully consider when deciding whether to use importance weighting or not.

## 7.3 Directions for Future Work

In this thesis, we have shown how synthetic data generation can be improved in small to medium-scale applications.

**Improving the scalability of AR-BP** For small-dimensional tabular datasets, we recommend using AR-BP (Chapter 3). While we have proposed an initial approach of incorporating neural networks to augment the modelling capabilities of AR-BP, this method remains computationally demanding and falls short of the performance of AR neural networks, paving the way for further refinements.

**Improving DP importance weighting with techniques from Chapter 6** Another potential contribution includes combining the results of Chapters 5 and 6 for DP data generators trained on low-dimensional datasets. We have seen in

Chapter 5 that the performance of importance weighting improves as the fit of the SDGP gets closer to the DGP, and the better the importance weights estimation performs. We thus believe that applying the techniques from Chapter 6 to the DP estimation of importance weights and to the DP data generation can improve the overall performance of the approach.

Note that importance weighting has been shown to not scale well to high-dimensional feature spaces [Tokdar and Kass, 2010]. Thus, alternative methods are necessary to improve the SDGP performance in this setting.

**Increasing the dimensionality of data generation** Scaling DP synthetic data generation up to high-dimensional feature spaces would enable the development of even more impactful applications. While we demonstrated the viability of DP synthetic data generation within medical applications involving lymph node tissue, other potentially transformative data domains include X-rays such as chest radiographs (e.g., CheXpert Irvin et al. [2019]) or face images such as CelebA-HQ [Karras et al., 2017]). Datasets typical for these applications usually exhibit a considerably higher image resolution than considered in this thesis, motivating future work on DP image generation in high dimensions. Methods to scale up DP generative modelling can include using pre-trained upscalers [Freeman et al., 2002, Yang et al., 2019] or parameter-efficient generative models, such as latent diffusion models [Lyu et al., 2023, Rombach et al., 2022]. Compared to denoising diffusion models, latent diffusion models are more parameter-efficient as they map high-resolution images to a small latent space with pre-trained GANs or VAEs. Following, they only train the diffusion model in the latent space.

# Bibliography


M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, pages 308–318, 2016.

H. Alqahtani, M. Kavakli-Thorne, G. Kumar, and F. SBSSTC. An analysis of evaluation metrics of gans. In *International Conference on Information Technology and Applications (ICITA)*, volume 7, 2019.

K. Amin, A. Kulesza, A. Munoz, and S. Vassilvtiskii. Bounding user contributions: A bias-variance trade-off in differential privacy. In *International Conference on Machine Learning*, pages 263–271. PMLR, 2019.

S. Amiri, E. Nalisnick, A. Belloum, S. Klous, and L. Gommans. Differential privacy vs detecting copyright infringement: A case study with normalizing flows. *1st Workshop on Generative AI and Law*, 2023.

S. Azizi, S. Kornblith, C. Saharia, M. Norouzi, and D. J. Fleet. Synthetic data from diffusion models improves imagenet classification. *arXiv preprint arXiv:2304.08466*, 2023.

D. Bank, N. Koenigstein, and R. Giryes. Autoencoders. *Machine Learning for Data Science Handbook: Data Mining and Knowledge Discovery Handbook*, pages 353–374, 2023.

S. S. Baraheem, T.-N. Le, and T. V. Nguyen. Image synthesis: a review of methods, datasets, evaluation metrics, and future outlook. *Artificial Intelligence Review*, pages 1–53, 2023.

R. F. Barber and J. C. Duchi. Privacy and statistical risk: Formalisms and minimax bounds. *arXiv preprint arXiv:1412.4451*, 2014.

A. Bayestehtashk and I. Shafran. Parsimonious multivariate copula model for density estimation. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 5750–5754. IEEE, 2013.

M. D. Begg and S. Lagakos. On the consequences of model misspecification in logistic regression. *Environmental health perspectives*, 87:69–75, 1990.

S. M. Bellovin, P. K. Dutta, and N. Reitinger. Privacy and synthetic datasets. *Stan. Tech. L. Rev.*, 22:1, 2019.

R. H. Berk. Limiting behavior of posterior distributions when the model is incorrect. *The Annals of Mathematical Statistics*, pages 51–58, 1966.

R. H. Berk. Consistency a posteriori. *The Annals of Mathematical Statistics*, 41(3): 894–906, 1970.

P. Berti, L. Pratelli, and P. Rigo. Limit theorems for a class of identically distributed random variables. *The Annals of Probability*, 32(3):2029–2052, 2004.

C. M. Bishop and N. M. Nasrabadi. *Pattern recognition and machine learning*. Number 4 in 1. Springer, 2006.

P. Bissiri, C. Holmes, and S. Walker. A general framework for updating belief distributions. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 2016.

S. Bond-Taylor, A. Leach, Y. Long, and C. G. Willcocks. Deep generative modelling: A comparative review of vaes, gans, normalizing flows, energy-based and autoregressive models. *IEEE transactions on pattern analysis and machine intelligence*, 2021.

A. Borji. Pros and cons of gan evaluation measures. *Computer vision and image understanding*, 179:41–65, 2019.

J.-P. Briot, G. Hadjeres, and F.-D. Pachet. Deep learning techniques for music generation–a survey. *arXiv preprint arXiv:1709.01620*, 2017.

A. Brock, J. Donahue, and K. Simonyan. Large scale gan training for high fidelity natural image synthesis. *arXiv preprint arXiv:1809.11096*, 2018.

M. Bun, J. Ullman, and S. Vadhan. Fingerprinting codes and the price of approximate differential privacy. In *Proceedings of the forty-sixth annual ACM symposium on Theory of computing*, pages 1–10, 2014.

N. Carlini, J. Hayes, M. Nasr, M. Jagielski, V. Sehwag, F. Tramèr, B. Balle, D. Ippolito, and E. Wallace. Extracting training data from diffusion models, 2023. URL `https://arxiv.org/abs/2301.13188`.

A. Charpentier, J.-D. Fermanian, and O. Scaillet. The estimation of copulas: Theory and practice. *Copulas: From theory to application in finance*, pages 35–64, 2007.

K. Chaudhuri, C. Monteleoni, and A. D. Sarwate. Differentially private empirical risk minimization. *Journal of Machine Learning Research*, 12(3), 2011.

Y.-C. Chen. A tutorial on kernel density estimation and recent advances. *Biostatistics & Epidemiology*, 1(1):161–187, 2017.

M. Connor, G. Canal, and C. Rozell. Variational autoencoder with learned latent structure. In *International Conference on Artificial Intelligence and Statistics*, pages 2359–2367. PMLR, 2021.

I. C. Covert, S. Lundberg, and S.-I. Lee. Explaining by removing: A unified framework for model explanation. *The Journal of Machine Learning Research*, 22(1):9477–9566, 2021.

L. Davies and U. Gather. *Robust statistics*. Springer, 2012.

S. De, L. Berrada, J. Hayes, S. L. Smith, and B. Balle. Unlocking high-accuracy differentially private image classification through scale. *arXiv preprint arXiv:2204.13650*, 2022.

P. Dhariwal and A. Nichol. Diffusion models beat gans on image synthesis. *Advances in Neural Information Processing Systems*, 34:8780–8794, 2021.

T. Dockhorn, T. Cao, A. Vahdat, and K. Kreis. Differentially private diffusion models. *arXiv preprint arXiv:2210.09929*, 2022. URL `https://openreview.net/pdf?id=pX21pH4CsNB`.

J. C. Duchi, M. I. Jordan, and M. J. Wainwright. Local privacy and statistical minimax rates. In *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*, pages 429–438. IEEE, 2013.

C. Dwork. Differential privacy. In *International colloquium on automata, languages, and programming*, pages 1–12. Springer, 2006.

C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of cryptography conference*, pages 265–284. Springer, 2006.

C. Dwork, A. Smith, T. Steinke, J. Ullman, and S. Vadhan. Robust traceability from trace amounts. In *2015 IEEE 56th Annual Symposium on Foundations of Computer Science*, pages 650–669. IEEE, 2015.

M. D. Escobar. *Estimating the means of several normal populations by nonparametric estimation of the distribution of the means*. PhD thesis, Yale University, 1988.

M. D. Escobar and M. West. Bayesian density estimation and inference using mixtures. *Journal of the american statistical association*, 90(430):577–588, 1995.

T. S. Ferguson. A Bayesian analysis of some nonparametric problems. *The annals of statistics*, pages 209–230, 1973.

R. J. Florax and P. Nijkamp. Misspecification in linear spatial regression models. *Tinbergen Institute Discussion Papers*, 1(2003-081):3, 2003.

E. Fong, C. Holmes, and S. G. Walker. Martingale posterior distributions. *To appear at the Journal of the Royal Statistical Society: Series B (with discussion)*, 2021.

S. Fortini and S. Petrone. Quasi-bayes properties of a procedure for sequential learning in mixture models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 82(4):1087–1114, 2020.

J. Foulds, J. Geumlek, M. Welling, and K. Chaudhuri. On the theory and practice of privacy-preserving bayesian data analysis. *arXiv preprint arXiv:1603.07294*, 2016.

W. T. Freeman, T. R. Jones, and E. C. Pasztor. Example-based super-resolution. *IEEE Computer graphics and Applications*, 22(2):56–65, 2002.

S. Ghalebikesabi, R. Cornish, C. Holmes, and L. Kelly. Deep generative missingness pattern-set mixture models. In *International Conference on Artificial Intelligence and Statistics*, pages 3727–3735. PMLR, 2021a.

S. Ghalebikesabi, L. Ter-Minassian, K. DiazOrdaz, and C. C. Holmes. On locality of local explanation models. *Advances in neural information processing systems*, 34:18395–18407, 2021b.

S. Ghalebikesabi, H. Wilde, J. Jewson, A. Doucet, S. Vollmer, and C. Holmes. Mitigating statistical bias within differentially private synthetic data. In *Uncertainty in Artificial Intelligence*, pages 696–705. PMLR, 2022.

S. Ghalebikesabi, L. Berrada, S. Gowal, I. Ktena, R. Stanforth, J. Hayes, S. De, S. L. Smith, O. Wiles, and B. Balle. Differentially private diffusion models generate useful synthetic images. *arXiv preprint arXiv:2302.13861*, 2023a.

S. Ghalebikesabi, C. C. Holmes, E. Fong, and B. Lehmann. Quasi-bayesian nonparametric density estimation via autoregressive predictive updates. In *Uncertainty in Artificial Intelligence*, pages 658–668. PMLR, 2023b.

I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

P. R. Hahn, R. Martin, and S. G. Walker. On recursive Bayesian predictive distributions. *Journal of the American Statistical Association*, 113(523):1085–1093, 2018.

J. M. Hammersley and K. W. Morton. Poor man's monte carlo. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 16(1):23–38, 1954.

F. R. Hampel, E. M. Ronchetti, P. J. Rousseeuw, and W. A. Stahel. *Robust statistics: the approach based on influence functions*. John Wiley & Sons, 2011.

G. Harshvardhan, M. K. Gourisaria, M. Pandey, and S. S. Rautaray. A comprehensive survey and analysis of generative models in machine learning. *Computer Science Review*, 38:100285, 2020.

J. Hayes, L. Melis, G. Danezis, and E. De Cristofaro. Logan: Membership inference attacks against generative models. *Proceedings on Privacy Enhancing Technologies*, 2019(1):133–152, 2019.

J. Hayes, S. Mahloujifar, and B. Balle. Bounding training data reconstruction in dp-sgd. *arXiv preprint arXiv:2302.07225*, 2023.

M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017.

J. Ho, A. Jain, and P. Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.

Z. Hu, Z. Yang, R. Salakhutdinov, and E. P. Xing. On unifying deep generative models. *arXiv preprint arXiv:1706.00550*, 2017.

P. J. Huber. Robust estimation of a location parameter. In *Breakthroughs in statistics: Methodology and distribution; published 1992*, pages 492–518. Springer, 1963.

P. J. Huber. John w. tukey's contributions to robust statistics. *Annals of statistics*, pages 1640–1648, 2002.

P. J. Huber. *Robust statistics*, volume 523. John Wiley & Sons, 2004.

P. J. Huber et al. The behavior of maximum likelihood estimates under nonstandard conditions. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, pages 221–233. Berkeley, CA: University of California Press, 1967.

N. B. Ipsen, P.-A. Mattei, and J. Frellsen. not-miwae: Deep generative modelling with missing not at random data. *arXiv preprint arXiv:2006.12871*, 2020.

J. Irvin, P. Rajpurkar, M. Ko, Y. Yu, S. Ciurea-Ilcus, C. Chute, H. Marklund, B. Haghgoo, R. Ball, K. Shpanskaya, et al. Chexpert: A large chest radiograph dataset with uncertainty labels and expert comparison. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 590–597, 2019.

J. Jewson, S. Ghalebikesabi, and C. Holmes. Differentially private statistical inference through $\beta$-divergence one posterior sampling. *arXiv preprint arXiv:2307.05194*, 2023.

W. Jiang and M. A. Tanner. Gibbs posterior for variable selection in high-dimensional classification and data mining. *Ann. Statist.*, pages 2207—-2231, 2008.

J. Jordon, L. Szpruch, F. Houssiau, M. Bottarelli, G. Cherubin, C. Maple, S. N. Cohen, and A. Weller. Synthetic data–what, why and how? *arXiv preprint arXiv:2205.03257*, 2022.

T. Joy, S. M. Schmon, P. H. Torr, N. Siddharth, and T. Rainforth. Capturing label characteristics in vaes. *arXiv preprint arXiv:2006.10102*, 2020.

G. Kamath, V. Singhal, and J. Ullman. Private mean estimation of heavy-tailed distributions. In *Conference on Learning Theory*, pages 2204–2235. PMLR, 2020.

G. Kamath, A. Mouzakis, M. Regehr, V. Singhal, T. Steinke, and J. Ullman. A bias-variance-privacy trilemma for statistical estimation. *arXiv preprint arXiv:2301.13334*, 2023.

T. Karras, T. Aila, S. Laine, and J. Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196*, 2017.

V. Karwa and S. Vadhan. Finite sample differentially private confidence intervals. *arXiv preprint arXiv:1711.03908*, 2017.

D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

D. P. Kingma, T. Salimans, R. Jozefowicz, X. Chen, I. Sutskever, and M. Welling. Improved variational inference with inverse autoregressive flow. *Advances in neural information processing systems*, 29, 2016.

I. Kobyzev, S. J. Prince, and M. A. Brubaker. Normalizing flows: An introduction and review of current methods. *IEEE transactions on pattern analysis and machine intelligence*, 43(11):3964–3979, 2020.

D. Kondratyuk, L. Yu, X. Gu, J. Lezama, J. Huang, R. Hornung, H. Adam, H. Akbari, Y. Alon, V. Birodkar, et al. Videopoet: A large language model for zero-shot video generation. *arXiv preprint arXiv:2312.14125*, 2023.

Z. Kong, W. Ping, J. Huang, K. Zhao, and B. Catanzaro. Diffwave: A versatile diffusion model for audio synthesis. *arXiv preprint arXiv:2009.09761*, 2020.

H. Larochelle and I. Murray. The neural autoregressive distribution estimator. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 29–37. JMLR Workshop and Conference Proceedings, 2011.

D. Lenis, B. Ackerman, and E. A. Stuart. Measuring model misspecification: Application to propensity score methods with complex survey data. *Computational statistics & data analysis*, 128:48–57, 2018.

A. Lugmayr, M. Danelljan, A. Romero, F. Yu, R. Timofte, and L. Van Gool. Repaint: Inpainting using denoising diffusion probabilistic models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11461–11471, 2022a.

A. Lugmayr, M. Danelljan, R. Timofte, K.-w. Kim, Y. Kim, J.-y. Lee, Z. Li, J. Pan, D. Shim, K.-U. Song, et al. Ntire 2022 challenge on learning the super-resolution space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 786–797, 2022b.

S. M. Lundberg and S.-I. Lee. A unified approach to interpreting model predictions. *Advances in neural information processing systems*, 30, 2017.

C. Luo. Understanding diffusion models: A unified perspective. *arXiv preprint arXiv:2208.11970*, 2022.

S. Lyu, M. Vinaroz, M. F. Liu, and M. Park. Differentially private latent diffusion models. *arXiv preprint arXiv:2305.15759*, 2023.

H. B. McMahan, G. Andrew, U. Erlingsson, S. Chien, I. Mironov, N. Papernot, and P. Kairouz. A general approach to adding differential privacy to iterative training procedures. *arXiv preprint arXiv:1812.06210*, 2018.

F. McSherry and K. Talwar. Mechanism design via differential privacy. In *48th Annual IEEE Symposium on Foundations of Computer Science (FOCS'07)*, pages 94–103. IEEE, 2007.

M. A. Medina, J. L. M. Olea, C. Rush, and A. Velez. On the robustness to misspecification of $\alpha$-posteriors and their variational approximations. *The Journal of Machine Learning Research*, 23(1):6579–6629, 2022.

K. Minami, H. Arai, I. Sato, and H. Nakagawa. Differential privacy without sensitivity. *Advances in Neural Information Processing Systems*, 29, 2016.

Muse AI. Midjourney: AI Image Generator, 2023. URL `https://www.midjourneyai.ai/`. Accessed: 2023-11-11.

T. Nagler and C. Czado. Evading the curse of dimensionality in nonparametric density estimation with simplified vine copulas. *Journal of Multivariate Analysis*, 151:69–89, 2016.

A. Nazabal, P. M. Olmos, Z. Ghahramani, and I. Valera. Handling incomplete heterogeneous data using vaes. *Pattern Recognition*, 107:107501, 2020.

S. I. Nikolenko. *Synthetic data for deep learning*, volume 174. Springer, 2021.

D. J. Nott, C. Drovandi, and D. T. Frazier. Bayesian inference for misspecified generative models. *Annual Review of Statistics and Its Application*, 11, 2023.

OpenAI. DALL · E 2: Creating Images from Text, 2022. URL `https://openai.com/dall-e-2`. Accessed: 2023-11-11.

OpenAI. Introducing ChatGPT Plus, 2023. URL `https://openai.com/blog/chatgpt-plus`. Accessed: 2023-11-11.

E. S. Pearson. The analysis of variance in cases of non-normal variation. *Biometrika*, pages 114–133, 1931.

N. Ponomareva, H. Hazimeh, A. Kurakin, Z. Xu, C. Denison, H. B. McMahan, S. Vassilvitskii, S. Chien, and A. G. Thakurta. How to dp-fy ml: A practical guide to machine learning with differential privacy. *Journal of Artificial Intelligence Research*, 77:1113–1201, 2023.

M. F. Pradier, N. Prasad, P. Chapfuwa, S. Ghalebikesabi, M. Ilse, S. Woodhouse, R. Elyanow, J. Zazo, J. Gonzalez, J. Greissl, et al. Airiva: A deep generative model of adaptive immune repertoires. *Machine Learning for Healthcare 2023*, 2023.

T. E. Raghunathan. Synthetic data. *Annual review of statistics and its application*, 8:129–140, 2021.

P. Rao. Some notes on misspecification in multiple regressions. *The American Statistician*, 25(5):37–39, 1971.

D. Rezende and S. Mohamed. Variational inference with normalizing flows. In *International conference on machine learning*, pages 1530–1538. PMLR, 2015.

D. J. Rezende, S. Mohamed, and D. Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *International conference on machine learning*, pages 1278–1286. PMLR, 2014.

C. P. Robert, G. Casella, and G. Casella. *Monte Carlo statistical methods*, volume 2. Springer, 1999.

R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022.

B. Roskams-Hieter, J. Wells, and S. Wade. Leveraging variational autoencoders for multiple data imputation. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 491–506. Springer, 2023.

D. B. Rubin. Inference and missing data. *Biometrika*, 63(3):581–592, 1976.

D. B. Rubin. Multiple imputations in sample surveys-a phenomenological bayesian approach to nonresponse. In *Proceedings of the survey research methods section of the American Statistical Association*, volume 1, pages 20–34. American Statistical Association Alexandria, VA, USA, 1978.

D. B. Rubin. Statistical disclosure limitation. *Journal of official Statistics*, 9(2): 461–468, 1993.

L. Ruthotto and E. Haber. An introduction to deep generative modeling. *GAMM-Mitteilungen*, 44(2):e202100008, 2021.

R. Salakhutdinov. Learning deep generative models. *Annual Review of Statistics and Its Application*, 2:361–385, 2015.

R. Salakhutdinov and G. Hinton. Deep boltzmann machines. In *Artificial intelligence and statistics*, pages 448–455. PMLR, 2009.

T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen. Improved techniques for training gans. *Advances in neural information processing systems*, 29, 2016.

M. Schmitt, P.-C. Bürkner, U. Köthe, and S. T. Radev. Detecting model misspecification in amortized bayesian inference with neural networks. *arXiv preprint arXiv:2112.08866*, 2021.

M. Sklar. Fonctions de repartition an dimensions et leurs marges. *Publ. inst. statist. univ. Paris*, 8:229–231, 1959.

A. Smith and U. Makov. A quasi-bayes sequential procedure for mixtures. *Journal of the Royal Statistical Society: Series B (Methodological)*, 40(1):106–112, 1978.

J. Sohl-Dickstein, E. Weiss, N. Maheswaranathan, and S. Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, pages 2256–2265. PMLR, 2015.

C. K. Sønderby, T. Raiko, L. Maaløe, S. K. Sønderby, and O. Winther. Ladder variational autoencoders. In *Advances in Neural Information Processing Systems*, pages 3738–3746, 2016.

Y. Song and S. Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in Neural Information Processing Systems*, 32, 2019.

Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020.

T. Stadler, B. Oprisanu, and C. Troncoso. Synthetic data–anonymisation groundhog day. *arXiv preprint arXiv:2011.07018*, 2021.

D. J. Stekhoven and P. Bühlmann. Missforest—non-parametric missing value imputation for mixed-type data. *Bioinformatics*, 28(1):112–118, 2012.

Sundar Pichai. An important next step on our AI journey, 2023. URL `https://blog.google/technology/ai/bard-google-ai-search-updates/`. Accessed: 2023-11-11.

E. G. Tabak and C. V. Turner. A family of nonparametric density estimation algorithms. *Communications on Pure and Applied Mathematics*, 66(2):145–164, 2013.

E. G. Tabak and E. Vanden-Eijnden. Density estimation by dual ascent of the log-likelihood. *Communications in Mathematical Sciences*, 8(1):217–233, 2010.

Y. W. Teh et al. Dirichlet process. *Encyclopedia of machine learning*, 1063:280–287, 2010.

L. Ter-Minassian, S. Ghalebikesabi, K. Diaz-Ordaz, and C. Holmes. Challenges and opportunities of shapley values in a clinical context. *ICML 2022 Workshop on Interpretable Machine Learning in Healthcare*, 2022.

L. Theis, A. v. d. Oord, and M. Bethge. A note on the evaluation of generative models. *arXiv preprint arXiv:1511.01844*, 2015.

S. T. Tokdar and R. E. Kass. Importance sampling: a review. *Wiley Interdisciplinary Reviews: Computational Statistics*, 2(1):54–60, 2010.

R. Torkzadehmahani, P. Kairouz, and B. Paten. Dp-cgan: Differentially private synthetic data and label generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0, 2019.

J. W. Tukey. A survey of sampling from contaminated distributions. *Contributions to probability and statistics*, pages 448–485, 1960.

S. Vansteelandt, M. Bekaert, and G. Claeskens. On model selection and model misspecification in causal inference. *Statistical methods in medical research*, 21 (1):7–30, 2012.

N. Vyas, S. Kakade, and B. Barak. Provable copyright protection for generative models. *arXiv preprint arXiv:2302.10870*, 2023.

I. Waernbaum. Model misspecification and robustness in causal inference: comparing matching with doubly robust estimation. *Statistics in medicine*, 31(15):1572–1581, 2012.

Y.-X. Wang, S. Fienberg, and A. Smola. Privacy for free: Posterior sampling and stochastic gradient monte carlo. In *International Conference on Machine Learning*, pages 2493–2502. PMLR, 2015.

J. L. Watson, D. Juergens, N. R. Bennett, B. L. Trippe, J. Yim, H. E. Eisenach, W. Ahern, A. J. Borst, R. J. Ragotte, L. F. Milles, et al. De novo design of protein structure and function with rfdiffusion. *Nature*, 620(7976):1089–1100, 2023.

H. White. Maximum likelihood estimation of misspecified models. *Econometrica: Journal of the econometric society*, pages 1–25, 1982.

V. D. Wild, S. Ghalebikesabi, D. Sejdinovic, and J. Knoblauch. A rigorous link between deep ensembles and (variational) bayesian methods. *arXiv preprint arXiv:2305.15027*, 2023.

H. Wilde, J. Jewson, S. Vollmer, and C. Holmes. Foundations of bayesian learning from synthetic data. In *International Conference on Artificial Intelligence and Statistics*, pages 541–549. PMLR, 2021.

W. Williams, S. Ringer, T. Ash, D. MacLeod, J. Dougherty, and J. Hughes. Hierarchical quantized autoencoders. *Advances in Neural Information Processing Systems*, 33:4524–4535, 2020.

Z. Xiao, K. Kreis, and A. Vahdat. Tackling the generative learning trilemma with denoising diffusion gans. *arXiv preprint arXiv:2112.07804*, 2021.

L. Xie, K. Lin, S. Wang, F. Wang, and J. Zhou. Differentially private generative adversarial network. *arXiv preprint arXiv:1802.06739*, 2018.

W. Yang, X. Zhang, Y. Tian, W. Wang, J.-H. Xue, and Q. Liao. Deep learning for single image super-resolution: A brief review. *IEEE Transactions on Multimedia*, 21(12):3106–3121, 2019.

H. Zhang, T. Xu, H. Li, S. Zhang, X. Wang, X. Huang, and D. N. Metaxas. Stackgan++: Realistic image synthesis with stacked generative adversarial networks. *IEEE transactions on pattern analysis and machine intelligence*, 41(8): 1947–1962, 2018.

T. Zhang. From $\epsilon$-entropy to kl-entropy: Analysis of minimum information complexity density estimation. *Ann. Statist*, 34:2180—-2210, 2006.

K. Zhu, P. Van Hentenryck, and F. Fioretto. Bias and variance of post-processing in differential privacy. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 11177–11184, 2021.

K. Zhu, F. Fioretto, P. Van Hentenryck, S. Das, and C. Task. Privacy and bias analysis of disclosure avoidance systems. *arXiv preprint arXiv:2301.12204*, 2023.