Noise-Aware Statistical Inference with Differentially Private Synthetic Data

Ossi Räisä
University of Helsinki
ossi.raisa@helsinki.fi

Joonas Jälkö
University of Helsinki*
joonas.jalko@helsinki.fi

Samuel Kaski
Aalto University, University of Manchester
samuel.kaski@aalto.fi

Antti Honkela
University of Helsinki
antti.honkela@helsinki.fi

Abstract

While generation of synthetic data under differential privacy (DP) has received a lot of attention in the data privacy community, analysis of synthetic data has received much less. Existing work has shown that simply analysing DP synthetic data as if it were real does not produce valid inferences of population-level quantities. For example, confidence intervals become too narrow, which we demonstrate with a simple experiment. We tackle this problem by combining synthetic data analysis techniques from the field of multiple imputation (MI), and synthetic data generation using noise-aware (NA) Bayesian modeling into a pipeline NA+MI that allows computing accurate uncertainty estimates for population-level quantities from DP synthetic data. To implement NA+MI for discrete data generation using the values of marginal queries, we develop a novel noise-aware synthetic data generation algorithm NAPSU-MQ using the principle of maximum entropy. Our experiments demonstrate that the pipeline is able to produce accurate confidence intervals from DP synthetic data. The intervals become wider with tighter privacy to accurately capture the additional uncertainty stemming from DP noise.

1 INTRODUCTION

Availability of data for research is constrained by the dilemma between privacy preservation and potential gains obtained from sharing. As a result, many datasets are kept confidential to mitigate the possibility of privacy violations, with access only granted to researchers after a lengthy approval process, if at all, slowing down research.

One approach to solving the dilemma between free access and confidentiality is releasing synthetic data, as proposed by Rubin (1993). The idea is that the *data holder* releases a synthetic dataset that is based on a real dataset. *Data analysts* can use the synthetic dataset instead of the real one for their *downstream analysis*.

The synthetic dataset should maintain population-level statistical properties of the original, which are of interest to the analysts. Privacy-protection of the synthetic data can be guaranteed by employing differential privacy (DP) (Dwork, McSherry, et al. 2006), which offers provable protection, unlike non-DP synthetic data generation methods.

The analysts of synthetic data should be able to draw valid conclusions on the data generating process (DGP) of the real data using the synthetic data. An important component of the conclusion in any scientific research is estimation of uncertainty, usually in the form of a confidence interval or p-value. However, as Wilde et al. (2021) point out, simply using synthetic data as if it were real data only allows drawing conclusions about the synthetic data generating process, not the real DGP.

The issues of using synthetic data in place of real data are especially apparent with DP, as DP requires adding noise to the synthetic data generation process. We illustrate this with a simple toy data experiment. We generate 3-dimensional binary data, where one variable is generated from logistic regression on the other two, serving as the original dataset. Then, we generate synthetic data from the original data and compute confidence intervals for the coefficients of the logistic regression from the synthetic data. A more detailed description of the setup is

---

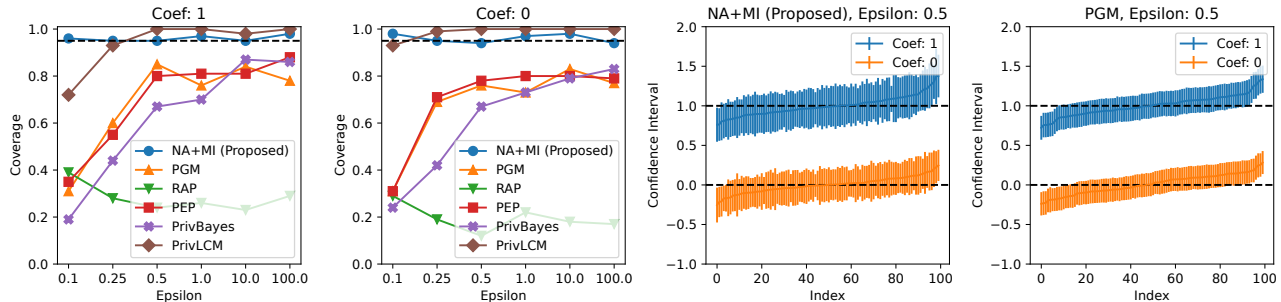*Work done while at Aalto University.

Figure 1: Toy data experiment results of logistic regression on 3 binary variables, showing that PGM, PEP, RAP and PrivBayes are overconfident, even with almost no privacy ($\epsilon = 100$), PrivLCM is underconfident, while our algorithm is well-calibrated. The first two panels from the left show the fraction of the 95% confidence intervals that contain the true parameter value in 100 repeats, with the target confidence level of 95% highlighted by a black line. ϵ quantifies the strength of the privacy guarantee, ranging from strong ($\epsilon = 0.1$) to meaningless ($\epsilon = 100$). The other privacy parameter is fixed at $\delta = 2.5 \cdot 10^{-7}$. The third panel shows the confidence intervals from synthetic data generated by our mechanism for $\epsilon = 0.5$, and the fourth panel shows the confidence intervals from PGM. The third and fourth panels show that the overconfidence stems from intervals that are too narrow, a result of failing to account for all uncertainty.

given in Section 5.1. As shown by Figure 1, the existing algorithms PGM (McKenna et al. 2019), PEP (T. Liu et al. 2021), RAP (Aydore et al. 2021) and PrivBayes (Zhang et al. 2017) do not provide valid confidence intervals, as they treat the synthetic data as real data. Only our method NA+MI and PrivLCM (Nixon et al. 2022) produce valid confidence intervals. However, PrivLCM is too conservative, producing much wider intervals, as shown by Figure 3a, and does not scale to complex datasets.

Our solution to overconfident uncertainty estimates builds on Rubin's original work on synthetic data generation (Rubin 1993). He proposed generating multiple synthetic datasets, running the analysis task on each of them, and combining the results with simple combining rules called *Rubin's rules* (Raghunathan et al. 2003; Reiter 2002). This workflow is modeled after *multiple imputation* (Rubin 1987), where it is used to deal with missing data. Generating multiple synthetic datasets allows the combining rules to account for the additional uncertainty that comes from the synthetic data generation process, which includes DP noise when the synthetic data is generated using a Bayesian model that accounts for the DP noise. We call the combined pipeline of noise-aware (NA) private synthetic data generation and analysis with multiple imputation (MI) the *NA+MI pipeline*. We give a more detailed description in Section 2.

To implement the NA step, we develop an algorithm called *Noise-Aware Private Synthetic data Using Marginal Queries* (NAPSU-MQ), that generates synthetic data from discrete tabular datasets using the noisy values of preselected marginal queries. We describe NAPSU-MQ in Section 4. In Section 5 and Supplemental Section G, we evaluate NAPSU-MQ on the UCI Adult and UCI US Census (1990) datasets, showing that it can produce accurate confidence intervals.

1.1 Related Work

There is a sizable literature on DP synthetic data generation. Most recent work in the area either releases the values of a set of simple queries, such as counting queries, under DP and uses them as the basis of synthetic data (Aydore et al. 2021; Bernstein et al. 2017; Cai et al. 2021; R. Chen et al. 2015; Hardt et al. 2012; T. Liu et al. 2021; McKenna et al. 2018; McKenna et al. 2021; McKenna et al. 2022; McKenna et al. 2019; Nixon et al. 2022; Vietri et al. 2020; Zhang et al. 2017), or trains some kind of generative model, often a GAN, using the whole real dataset under DP (D. Chen et al. 2020; Jälkö et al. 2021; Long et al. 2021; Xie et al. 2018; Yoon et al. 2019). There are also hybrid approaches that use sophisticated queries that can capture all features of the dataset, and train a generative model using those (Harder et al. 2021; Liew et al. 2022). Of the existing DP synthetic data generation algorithms, NAPSU-MQ is closest to the PGM algorithm (McKenna et al. 2019), which does maximum likelihood estimation with the same data model as NAPSU-MQ instead of noise-aware Bayesian inference. We describe this connection in more detail in Supplemental Section D.

Rubin's rules were originally developed for analyses on missing data, as part of an approach called multiple imputation (Rubin 1987), which was later applied to generate and analyse synthetic data (Rubin 1993) without DP. The variant of Rubin's rules that we use, and describe in Supplemental Section B, was developed specifically for synthetic data generation (Raghunathan et al. 2003; Reiter 2002). Raab et al. (2018) have developed simpler alternatives to Rubin's rules under more restrictive assumptions, but these assumptions rule out DP data synthesisers.

Rubin's rules have not been widely used with DP synthetic

data generation, and we are only aware of four existing works studying the combination. Charest (2010) studied Rubin's rules with a very simple early synthetic data generation algorithm, and concluded that Rubin's rules are not appropriate for that algorithm. Zheng (2015) found that some simple one-dimensional methods developed by the multiple imputation community are in fact DP, but not with practically useful privacy bounds. Nixon et al. (2022) propose using Rubin's rules with the noise-aware synthetic data generation algorithm PrivLCM, but they only consider computing confidence intervals of query values on the real dataset, not confidence intervals of population parameters of arbitrary downstream analyses. F. Liu (2022) proposes generating multiple synthetic datasets like we do, but their pipeline requires splitting the privacy budget between each synthetic dataset, severely limiting the number of datasets that can be generated with acceptable utility, and their convergence theory assumes weak privacy[1].

Noise-aware uncertainty estimates have been developed for specific DP analyses. Examples include frequentist linear regression (Evans and King 2022) and general recipes for DP analyses without synthetic data (Covington et al. 2021; Ferrando et al. 2022). Bayesian examples include posterior inference for simple exponential family models (Bernstein and Sheldon 2018), linear regression (Bernstein and Sheldon 2019), generalised linear models (Kulkarni et al. 2021), and approximate Bayesian computation (Gong 2022). The AIM algorithm for synthetic data generation (McKenna et al. 2022) provides valid confidence intervals on the query values of the real dataset, while NAPSU-MQ and the other methods mentioned provide confidence intervals on population values.

Several works study techniques for mitigating the effect of DP noise. Wilde et al. (2021) point out the importance of noise-aware synthetic data analysis with DP and use publicly available data to augment the analysis and correct for the DP noise in Bayesian inference. Other examples include bias reduction (Ghalebikesabi et al. 2021) and averaging GANs (Neunhoeffer et al. 2021).

While some of the existing works address uncertainty estimates for specific analyses of synthetic data under DP, there is no existing method for proper uncertainty estimation for general downstream analyses of population-level quantities. We fill this gap with the NA+MI pipeline, which we implement for discrete tabular data with NAPSU-MQ.

2 THE NA+MI PIPELINE

The early work on synthetic data generation with multiple imputation showed that computing accurate uncertainty estimates with synthetic data requires accounting for the additional uncertainty that comes from generating the syn-

thetic data (Raghunathan et al. 2003; Rubin 1993). Rubin (1993) proposed generating multiple synthetic datasets X_i^{Syn} from the posterior predictive distribution $p(X^*|X)$, where X^* is a prediction of a future dataset, and X is the observed real dataset. The downstream analysis is run on each X_i^{Syn} as if X_i^{Syn} were the real dataset, and the results are combined using specialised combining rules (Raghunathan et al. 2003).

The generation of multiple datasets from $p(X^*|X)$ is necessary to give the combining rules a way to estimate the variance of the synthetic data generation process, which would not be possible if only a single dataset was generated. For a parametric model, $p(X^*|X) = \int_\Theta p(X^*|\theta)p(\theta|X)\mathrm{d}\theta$, where $\theta \in \Theta$ is the parameter, $p(X^*|\theta)$ is the likelihood, and $p(\theta|X)$ is the posterior of the parameter. X_i^{Syn} is then generated in two steps: first $\theta_i \sim p(\theta|X)$ is sampled, then $X_i^{Syn} \sim p(X^*|\theta_i)$.

The combining rules require including the posterior distribution $p(\theta|X)$ (Raghunathan et al. 2003) for sampling X_i^{Syn}, so a non-Bayesian model that samples $X_i^{Syn} \sim p(X^*|\hat{\theta})$ for some point-estimate $\hat{\theta}$ is not suitable for synthetic data generation with Rubin's rules.

Requiring the synthetic data generation to be DP complicates the picture, as only a noisy observation $\tilde{s}$ of X can be made, which means that we must use $p(\theta|\tilde{s})$ instead of $p(\theta|X)$. We call inference algorithms for $p(\theta|\tilde{s})$ that account for the noise added for DP *noise-aware*. The combination of noise-aware inference and multiple imputation is the NA+MI pipeline, which we summarise in Figure 2. First, the data holder runs inference on a noise-aware Bayesian model using the private data, which we call the NA step. Different implementations of the NA step may set different requirements on the form of X, the type of DP noise, and may provide different privacy guarantees.

After the inference, the data holder generates multiple synthetic datasets. The data holder can also release the posterior distribution in addition to the synthetic datasets, so that the analyst can also generate synthetic datasets if needed. Due to the post-processing immunity of DP (Theorem 3.7), releasing multiple synthetic datasets, or the posterior distribution, does not compromise privacy.

For each synthetic dataset, the analyst runs their analysis, and combines the results using multiple imputation (Raghunathan et al. 2003; Reiter 2002; Reiter 2005). We call this the MI step. For frequentist downstream analyses, we use Rubin's rules, which require that each analysis produces a point estimate q_i, and a variance estimate v_i for the point estimate. The point estimates $q_1, \ldots, q_m$ and the variance estimates $v_1, \ldots, v_m$ are fed to Rubin's rules (Raghunathan et al. 2003), which give a t-distribution for the estimated quantity that the analyst can use to compute confidence intervals or hypothesis tests. We describe

---

[1]The theory applies asymptotically when $\epsilon \to \infty$.
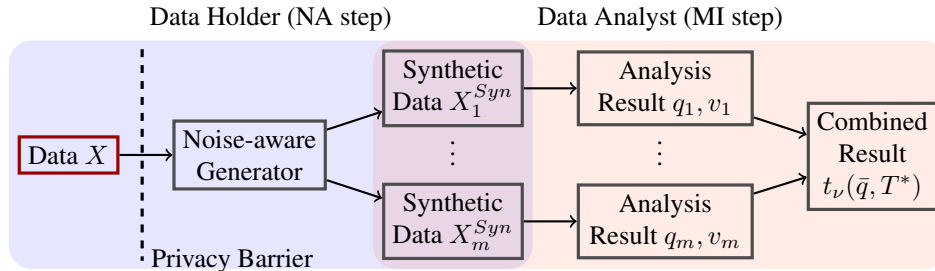
Figure 2: NA+MI pipeline for noise-aware DP synthetic data generation and statistical inference. The nodes shaded in blue are computed by the data holder, and the nodes shaded in orange are computed by the data analyst. All nodes except Data (with red border) can be released to the public. The synthetic datasets can be generated by either party because the Generator is also released by the data holder.

Rubin's rules in more detail in Supplemental Section B.

3   BACKGROUND FOR NAPSU-MQ

In this section we describe the datasets and queries NAPSU-MQ uses, and briefly describe key concepts from differential privacy, which we will use in Section 4.

Data and Marginal Queries   NAPSU-MQ uses tabular datasets of d discrete variables, where the domains of the variables, and the number datapoints n are known. We denote the set of possible datapoints by $\mathcal{X}$, and the set of possible datasets by $\mathcal{X}^n$. For a set of variables I and $x \in \mathcal{X}$, $x^{(I)}$ denotes the selection of the variables in I from x.

Definition 3.1. *A marginal query of variables I and value v is a function $a\colon \mathcal{X} \to \{0, 1\}$ that takes a datapoint x as input and returns 1 if $x^{(I)} = v$ and 0 otherwise. For a dataset $X \in \mathcal{X}^n$, we define $a(X) = \sum_{i=1}^{n} a(x_i)$, where x_i is the i:th datapoint in X.*

When I has k variables, a is called a k-way marginal query.

When evaluating multiple marginal queries $a_1, \ldots, a_{n_q}$, we concatenate their values to a vector-valued function $a\colon \mathcal{X} \to \{0, 1\}^{n_q}$. We call the concatenation of marginal queries for all possible values of variables I the *full set of marginals on I* [2].

As a concrete example, take the 3-dimensional binary data used in Figure 1. A single 2-way marginal query could, for example, look at the first two variables of a datapoint, and check that both are 0. The full set of marginal queries on the first two variables checks which of the 4 possible values the first two variables of a datapoint have, and returns a 4-component vector with a single 1 and 3 zeros, indicating the answer. As input to NAPSU-MQ, we could use the concatenation of the 3 full sets of marginal queries for each

pair of variables [3]. Including these kinds of marginals with more than one variable allows NAPSU-MQ to take the dependencies between variables into account.

Differential Privacy   Differential privacy (DP) (Dwork, Kenthapadi, et al. 2006; Dwork, McSherry, et al. 2006) is a definition aiming to quantify the privacy loss resulting from releasing the results of some algorithm. DP algorithms are also called *mechanisms*.

Definition 3.2. *A mechanism $\mathcal{M}$ is (ϵ, δ)-differentially private if for all neighbouring datasets X, X' and all measurable output sets S*

$$P(\mathcal{M}(X) \in S) \leq e^{\epsilon} P(\mathcal{M}(X') \in S) + \delta. \quad (1)$$

The neighbourhood relation in the definition is domain specific. We use the *substitute* neighbourhood relation for tabular datasets, where datasets are neighbouring if they differ in at most one datapoint.

Together, ϵ and δ bound the tradeoff between false positive and false negative rates for any hypothesis test (Kairouz et al. 2015). Their choice is a matter of policy (Dwork 2008). $\delta \approx \frac{1}{n}$ permits mechanisms that clearly violate privacy (Dwork and Roth 2014), so one should choose $\delta \ll \frac{1}{n}$.

The mechanism we use to release marginal query values under DP is the *Gaussian mechanism* (Balle and Wang 2018; Dwork, Kenthapadi, et al. 2006).

Definition 3.3. *The Gaussian mechanism with noise variance σ_{DP}^2 adds Gaussian noise to the value of a function $f\colon \mathcal{X}^n \to \mathbb{R}^k$ for input data X: $\mathcal{M}(X) = f(X) + \mathcal{N}(0, \sigma_{DP}^2 I)$.*

The privacy bounds of the Gaussian mechanism depend on the *sensitivity* of the function f, which is an upper bound on the change in the value of f for neighbouring datasets. Larger sensitivities require a larger noise variance.

---

[2] Some existing works (McKenna et al. 2019) use the term marginal query for the full set of marginal queries. We chose this terminology because we deal with individual marginal queries in Supplemental Section C.

[3] Our experiments for the toy dataset use the full set of 3-way marginals that includes all 3 variables.

Definition 3.4. *The L_2-sensitivity of a function f is $\Delta_2 f = \sup_{X \sim X'} ||f(X) - f(X')||_2$. $X \sim X'$ denotes that X and X' are neighbouring.*

The sensitivity of a concatenation of full sets of marginal queries has a simple form:

Theorem 3.5. *Let a be the concatenation of n_s full sets of marginal queries. Then $\Delta_2 a \leq \sqrt{2n_s}$.*

Proof. We defer the proof to Supplemental Section A. □

Theorem 3.6 (Balle and Wang (2018)). *The Gaussian mechanism for a function f with L_2-sensitivity Δ_2 and noise variance σ_{DP}^2 is (ϵ, δ)-DP with*

$$\delta \geq \Phi\left(\frac{\Delta_2}{2\sigma_{DP}} - \frac{\epsilon\sigma_{DP}}{\Delta_2}\right) - e^\epsilon \Phi\left(\frac{-\Delta_2}{2\sigma_{DP}} - \frac{\epsilon\sigma_{DP}}{\Delta_2}\right) \quad (2)$$

where Φ is the cumulative distribution function of the standard Gaussian distribution.

If ϵ, δ and Δ_2 are given, which is typical, σ_{DP}^2 can be solved from Theorem 3.6 using standard numerical methods.

An important property of DP is post-processing immunity: post-processing the result of a DP-algorithm does not weaken the privacy bounds.

Theorem 3.7 (Dwork and Roth (2014)). *Let $\mathcal{M}$ be an (ϵ, δ)-DP mechanism, and let f be any algorithm. Then the composition $f \circ \mathcal{M}$ is (ϵ, δ)-DP.*

Because of post-processing immunity, we can release marginal query values with the Gaussian mechanism to obtain privacy bounds, and make arbitrary use of the noisy query values without weakening the privacy bounds. This allows releasing an arbitrary number of synthetic datasets without compromising on privacy.

4 NOISE-AWARE SYNTHETIC DATA GENERATION

In order to implement the NA step, the data holder needs to generate synthetic data from the posterior of a noise-aware Bayesian model. Bernstein and Sheldon (2018) develop noise-aware Bayesian inference under DP for simple exponential family models, with noise added to sufficient statistics. However, their algorithm requires both computing unnormalised densities for the model, and sampling from the model's conjugate prior. In our setting, neither of these is trivial.

We implement the NA step by generalising their algorithm to use arbitrary marginal queries as the sufficient statistics, using the maximum entropy distribution of the marginal queries as our exponential family model. We develop a

computationally feasible method to compute the unnormalised density of this model, and sidestep the conjugate prior sampling by using standard posterior sampling methods, which also allows us to use a non-conjugate prior.

We start by considering an arbitrary set of marginal queries a. We would like to find an exponential family distribution that, in expectation, gives the same answers to a as the real data X. We do not want to assume anything else about the distribution besides these expected query values, so we use the principle of maximum entropy (Jaynes 1957) to choose the distribution.

The distribution with maximal entropy that satisfies the constraint $E_{X^* \sim P}(a(X^*)) = a(X)$ is

$$P(x) = \exp(\theta^T a(x) - \theta_0(\theta)) \quad (3)$$

for some parameters $\theta \in \mathbb{R}^k$ (Wainwright and Jordan 2008), where k is the number of queries. $\theta_0(\theta)$ is the normalising constant of the distribution, so it is given by $\theta_0(\theta) = \ln\left(\sum_{x \in \mathcal{X}} \exp(\theta^T a(x))\right)$. We denote this distribution by MED_θ, and use MED_θ^n to denote the distribution of n i.i.d. samples from MED_θ.

MED_θ is clearly an exponential family distribution, with sufficient statistic $a(x)$ and natural parameters θ. MED_θ is also a Markov network (Koller and Friedman 2009), given in log-linear form.

The Bayesian model we consider is derived from the generative process of the noisy query values, which are observed. Assuming that the data generating process is MED_θ, and knowing that the Gaussian mechanism adds noise with variance σ_{DP}^2, we get the probabilistic model

$$\theta \sim \text{Prior}, \qquad X \sim \text{MED}_\theta^n, \quad (4)$$
$$s = a(X), \qquad \tilde{s} \sim \mathcal{N}(s, \sigma_{DP}^2 I). \quad (5)$$

In principle, we could now sample from the posterior $p(\theta \mid \tilde{s})$, with s marginalised out. In practice, the marginalisation is not feasible, as s is a discrete variable with a very large domain.

However, s is a sum of the query values for individual datapoints, so asymptotically s has a normal distribution by the central limit theorem. We can substitute the normal approximation for s into the model, which allows us to easily marginalise s out, resulting in

$$\theta \sim \text{Prior}, \quad \tilde{s} \sim \mathcal{N}(n\mu(\theta), n\Sigma(\theta) + \sigma_{DP}^2 I), \quad (6)$$

where $\mu(\theta)$ and $\Sigma(\theta)$ denote the mean and covariance of $a(x)$ for a single sample $x \sim \text{MED}_\theta$.

To compute $\mu(\theta)$ and $\Sigma(\theta)$, we use both the exponential family and Markov network structure of MED_θ. Due to the exponential family structure,

$$\mu(\theta) = \nabla\theta_0(\theta), \qquad \Sigma(\theta) = H_{\theta_0}(\theta), \quad (7)$$

where H_{θ_0} is the Hessian of θ_0. Computing θ_0 naively requires summing over the exponentially large domain $\mathcal{X}$, which is not tractable for complex domains. The Markov network structure gives a solution: θ_0 can be computed with the variable elimination algorithm (Koller and Friedman 2009). We can then autodifferentiate variable elimination to compute $\mu(\theta)$ and $\Sigma(\theta)$. Alternatively, $\mu(\theta)$ can be computed by belief propagation (Koller and Friedman 2009), and $\Sigma(\theta)$ can be autodifferentiated from it.

For the prior, we choose another Gaussian distribution with mean 0 and standard deviation 10, which is a simple and weak prior, but other priors could be used.

To sample the posterior, we use existing posterior inference methods, namely the Laplace approximation (Gelman et al. 2014), which approximates the posterior with a Gaussian distribution centered at the posterior mode, and the NUTS algorithm (Hoffman and Gelman 2014), which is a *Markov chain Monte Carlo* (MCMC) algorithm that samples the posterior directly using the gradients of the posterior log-density.

The time complexities of computing $\mu(\theta)$ and $\Sigma(\theta)$ for inference, as well as sampling MED_θ after inference, depend on the sparsity[4] of the Markov network graph that the selected set of queries induces. Specifically, the time complexities are exponential in the *tree width* of the graph (Koller and Friedman 2009), which can be much lower than the dimensionality for sparse graphs, making inference and sampling tractable for sparse queries.

If we include all possible marginal queries from the selected variable sets, the parametrisation of MED_θ is not identifiable, as there are linear dependencies among the queries (Koller and Friedman 2009). Nonidentifiablity causes NUTS sampling to be very slow, so we prune the queries to remove linear dependencies while preserving the information in the queries using the *canonical parametrisation* for MED_θ (Koller and Friedman 2009). We give a detailed description of the process in Supplemental Section C.

4.1 NAPSU-MQ Properties

We summarise NAPSU-MQ in Algorithm 1. The privacy bounds for NAPSU-MQ follow from the material of Section 3:

Theorem 4.1. *NAPSU-MQ (Algorithm 1) is (ϵ, δ)-DP with regards to the real data X.*

Proof. The returned values of Algorithm 1 only depend on the real data X through $\tilde{s}$. Releasing $\tilde{s}$ is (ϵ, δ)-DP due to the selection of σ_{DP}^2 with Theorem 3.6. Computing the returned values from $\tilde{s}$ is post-processing, so by Theorem 3.7,

NAPSU-MQ is (ϵ, δ)-DP. □

In Supplemental Section B, we list the conditions under which Rubin's rules are unbiased (Raghunathan et al. 2003). As is typical with statistical methods, these assumptions are asymptotic in nature. Our experiments in Section 5 show that NAPSU-MQ is robust to these asymptotics and works with large enough samples.

The most important of the assumptions behind Rubin's rules[5] for synthetic data generation requires that the synthetic data generation does not bias the downstream analysis. This means that the marginal queries input to NAPSU-MQ must contain the relevant information for the downstream analysis. For our experiments, we include a fixed set of queries that gives enough information for the downstream task, and select other queries automatically with an existing query selection algorithm (McKenna et al. 2021). We leave further work on query selection for noise-aware inference to future work.

Algorithm 1: NAPSU-MQ

Input: Real data X, marginal queries a, number of synthetic datasets m, size of synthetic datasets n_{Syn}, privacy bounds ϵ, δ.
Output: Posterior distribution $p(\theta|\tilde{s})$, synthetic datasets $X_1^{Syn}, \ldots, X_m^{Syn}$.
$a^* \leftarrow$ Canonical queries for a (Section C);
$s \leftarrow a^*(X)$;
$\Delta_2 \leftarrow$ Sensitivity of s (Theorem 3.5);
$\sigma_{DP}^2 \leftarrow$ Required noise variance for (ϵ, δ)-DP with sensitivity Δ_2 (Theorem 3.6);
Sample $\tilde{s} \sim \mathcal{N}(s, \sigma_{DP}^2)$;
Run Bayesian inference algorithm to find $p(\theta|\tilde{s})$ (Section 4);
Sample $\theta_i \sim p(\theta|\tilde{s})$ and $X_i^{Syn} \sim \text{MED}_{\theta_i}^{n_{Syn}}$ for $1 \leq i \leq m$;
return $p(\theta|\tilde{s})$, $X_1^{Syn}, \ldots, X_m^{Syn}$

---

5 EXPERIMENTS

In this section, we give detailed descriptions on our two main experiments: a simple toy data experiment, and our experiment with the UCI Adult dataset, which demonstrate that NAPSU-MQ is able to compute accurate confidence intervals. In Supplemental Section G, we describe an additional experiment with the UCI US Census (1990) dataset, which confirms the results of the other experiments. Our code is available under an open source license[6].

---

[4] In a sparse graph, the fraction of pairs of nodes with an edge between them is small.

[5] Assumptions B.3 and B.4 in Supplemental SectionB.

[6] A library implementation of NAPSU-MQ is available at https://github.com/DPBayes/twinify, while code to replicate our experiments is at https://github.com/DPBayes/NAPSU-MQ-experiments.

Ossi Räisä, Joonas Jälkö, Samuel Kaski, Antti Honkela
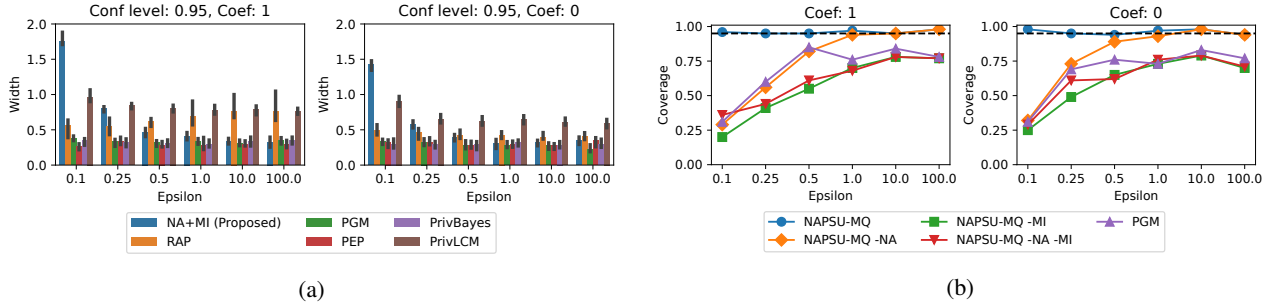

(a)   (b)

Figure 3: (a) Toy data confidence intervals widths. NA+MI produces slightly wider intervals than PGM or PEP for $\epsilon > 0.1$, which is necessary to account for DP noise. PrivLCM produces much wider intervals. (b) Ablation study on the toy data. "-NA" refers to removing noise-awareness, and "-MI" refers to removing multiple imputation. Unless both are included, NAPSU-MQ is overconfident like PGM except for $\epsilon \geq 1$ where noise-awareness is not necessary. $\delta = 2.5 \cdot 10^{-7}$ in both (a) and (b).

5.1 Toy Data

To demonstrate the necessity of noise-awareness in synthetic data generation, we measure the coverage of confidence intervals computed from DP synthetic data on a generated toy dataset where the data generation process is known. We test the existing algorithms PGM (McKenna et al. 2019), PEP (T. Liu et al. 2021), RAP (Aydore et al. 2021), PrivLCM (Nixon et al. 2022), PrivBayes (Zhang et al. 2017) and our pipeline NA+MI, where data generation is implemented with NAPSU-MQ. The authors of PrivLCM also propose using multiple imputation (Nixon et al. 2022), so we use Rubin's rules (Raghunathan et al. 2003) with the output of PrivLCM. We also ran PGM, PEP, RAP and PrivBayes with Rubin's rules, which did not produce useful results, as these algorithms do not meet the assumptions of Rubin's rules.

The original data consists of $n = 2000$ datapoints of 3 binary variables. The first two are sampled by independent coin flips. The third is sampled from logistic regression on the other two variables with coefficients $(1, 0)$.

For all algorithms except PrivLCM and PrivBayes, we use the full set of 3-way marginal queries released with the Gaussian mechanism. PrivLCM doesn't implement these, and instead uses all full sets of 2-way marginals, and a different mechanism, which is $(\epsilon, 0)$-DP (Nixon et al. 2022) instead of (ϵ, δ)-DP like the other algorithms. PrivBayes requires specifying a Bayesian network (Zhang et al. 2017), which we set to match the data generating process, and takes a single full set of 1-way marginals and a single full set of 2-way marginals in addition to the full set of 3-way marginals the other algorithms take. We use the Laplace approximation for NAPSU-MQ inference, as it is much faster than NUTS and works well for this simple setting.

DP algorithms are typically evaluated under changing privacy bounds by fixing $\delta \leq \frac{1}{n}$, and varying ϵ, which is the setting used by the authors of PGM (McKenna et al. 2019),

PEP (T. Liu et al. 2021) and RAP (Aydore et al. 2021). We follow this setting, fixing $\delta = n^{-2} = 2.5 \cdot 10^{-7}$.

We generate $m = 100$ synthetic datasets of size $n_{Syn} = n$ for all algorithms except RAP, where the synthetic dataset size is a function of two hyperparameters. We describe the hyperparameters in detail in Supplemental Section E.

The downstream task is inferring the logistic regression coefficients from synthetic data. We repeated all steps 100 times to measure the probability of sampling a dataset giving a confidence interval that includes the true parameter values.

Figure 1 shows the coverages, and Figure 3a shows the widths for the resulting confidence intervals. All of the algorithms apart from ours and PrivLCM are overconfident, even with very loose privacy bounds. Examining the confidence intervals shows the reason: PGM is unbiased, but it produces too narrow confidence intervals, while NAPSU-MQ produces wider confidence intervals. On the other hand, for $\epsilon > 0.25$, PrivLCM produces much wider and too conservative confidence intervals.

Ablation Study We also conducted an ablation study on the toy data to show that both multiple imputation and noise-awareness are necessary for accurate confidence intervals. The results are presented in Figure 3b. Without both multiple imputation and noise-awareness, NA+MI is overconfident like PGM, except for $\epsilon \geq 1$, where noise-awareness is not required. We show the confidence intervals produced by each method for $\epsilon = 0.5$ in Figure S2 in the Supplement.

5.2 Adult Dataset

Our main experiment evaluates the performance of NAPSU-MQ on the UCI Adult dataset (Kohavi and Becker 1996). We include 10 of the original 15 columns to remove redundant columns and keep runtimes manageable,
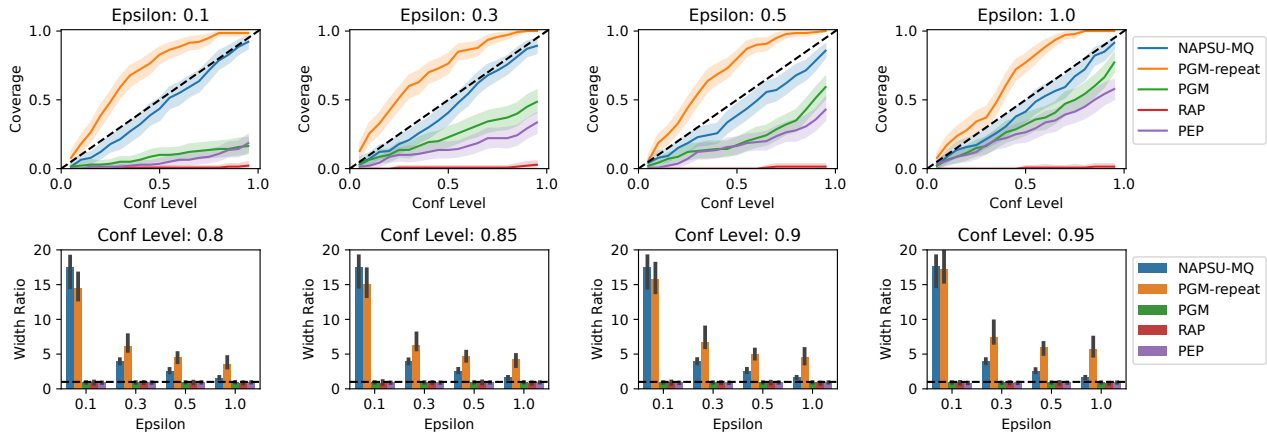
Figure 4: Top row: the fraction of downstream coefficients where the synthetic confidence interval contains the real data coefficient, averaged over 20 repeated runs on the Adult dataset using regularisation. NAPSU-MQ is the only algorithm that is consistently around the diagonal, showing good calibration. The error bands are bootstrap 95% confidence intervals of the average over the 20 repeats. Bottom row: confidence intervals widths divided by real data confidence interval widths. Each bar is a median over the different coefficients and repeats, and the black lines are 95% bootstrapped confidence intervals. The dashed line is at $y = 1$, showing where synthetic confidence intervals have the same width as original confidence intervals. $\delta \approx 4.7 \cdot 10^{-10}$ in all panels.

and discretise the continuous columns. After dropping rows with missing values, there are $n = 46\,043$ rows. The discretised domain has $1\,792\,000$ distinct values. We give a detailed description of the dataset, query selection and the downstream task in Supplemental Section F.

As our downstream task, we use logistic regression with income as the dependent variable and a subset of the columns as the independent variables, which allows us to include all the relevant marginals for synthetic data generation. The synthetic dataset was still generated with all 10 columns.

We compare NAPSU-MQ against PGM (McKenna et al. 2019), RAP (Aydore et al. 2021) and PEP (T. Liu et al. 2021). We used the published implementations of their authors for all of them, with small modifications to ensure compatibility with new library versions and our experiments. The published implementation of PrivLCM only supports binary data, and does not scale to datasets of this size, so it was not included in this experiment. PrivBayes was also excluded, as it doesn't support the set of queries we use in this experiment. We also include a naive noise-aware baseline that runs m completely independent repeats of PGM, splitting the privacy budget appropriately, and uses Rubin's rules with the m generated synthetic datasets.

NAPSU-MQ and PGM-repeat sometimes generate synthetic datasets with no people of some race with high income. Logistic regression will produce an extremely wide confidence intervals for the corresponding coefficients. Rubin's rules average over estimates, so even a single bad estimate makes the combined confidence interval extremely wide. This can be fixed in two ways: a simple solution is

removing estimates with extremely large variances before applying Rubin's rules. A more principled way is to add a very small regularisation term to the logistic regression, which fixes the extremely wide confidence intervals, but will require bootstrapping to get variance estimates, which increases the computational cost of the downstream analysis. We used an l_2-regularisation term of 10^{-5}, and used 50 bootstrap samples. Because of post-processing immunity, neither of these fixes affects the privacy bounds.

As the input queries, we pick 2-way marginals that are relevant for the downstream task, and select the rest of the queries with the MST algorithm (McKenna et al. 2021). This selection was kept constant throughout the experiment. For the privacy budget, we use $\delta = n^{-2} \approx 4.7 \cdot 10^{-10}$ for all runs, and vary ϵ.

Reiter (2002) discusses the choice of n_{Syn} and n for non-DP synthetic data generation in detail. Based on his results, choosing $n_{Syn} = n$ is very safe, and we use it for all algorithms except RAP, as in the toy data experiment (Section 5.1). For NAPSU-MQ and PGM-repeat, we choose the number of generated synthetic datasets with a preliminary experiment, presented in Figures S4 and S3 in the Supplement. For NAPSU-MQ, the theory of Reiter (2002), suggests that a larger m is better, but has diminishing returns. Our results in Figure S4 validate this, as all values of m produce similar results. We describe the other hyperparameters in detail in Supplemental Section E.

The Laplace approximation for NAPSU-MQ does not work well for this setting because many of the queries have small values, so we use NUTS (Hoffman and Gelman 2014) for

posterior inference. To speed up NUTS, we normalise the posterior before running the inference using the mean and covariance of the Laplace approximation

Results from 20 repeats of the experiment are shown in Figure 4. PGM, RAP and PEP produce overconfident confidence intervals that do not meet the given confidence levels. With the repeats, PGM becomes overly conservative, and produces confidence intervals that are too wide. NAPSU-MQ is the only algorithm that produces properly calibrated intervals, although repeated PGM is able to produce narrower intervals than NAPSU-MQ with $\epsilon = 0.1$. With the more realistic $\epsilon = 1$, the confidence intervals from NAPSU-MQ are not much wider than non-DP intervals, while PGM-repeat produces much wider intervals. Figure S8 shows that NAPSU-MQ reproduces 1- and 2-way marginals nearly as accurately as PGM.

The results in Figure 4 were obtained using the small regularisation term. Figure S5 shows the results with the trick of dropping large variances, which are very close to Figure 4.

Noise-awareness, especially with the increased accuracy from NUTS, comes with a steep computational cost, as PGM ran in 15s, while the Laplace approximation took several minutes, and NUTS took up to ten hours. All of the algorithms were run on 4 CPU cores of a cluster. The complete set of runtimes for all algorithms and values of ϵ are shown in Table S1 of the Supplement.

6 DISCUSSION

While our general pipeline NA+MI is applicable to all kinds of datasets in principle, the data generation algorithm NAPSU-MQ is currently only applicable to discrete tabular data due to the reliance on perturbing query values, and only supports sparse marginal queries perturbed with the Gaussian mechanism as input due to the techniques we use to make the algorithm practical.

We aim to generalise NAPSU-MQ to more general query classes, such as linear queries, in the future. Getting rid of the dependency on queries completely is likely to be much more challenging, as it will require developing noise-aware Bayesian inference without perturbing sufficient statistics. This may be possible by adding noise-awareness to other types of DP Bayesian inference methods, like DP variational inference (Jälkö et al. 2017) or DP MCMC (Heikkilä et al. 2019; Yildirim and Ermis 2019).

Only handling discrete data is not a major limitation, as the combination of discretisation (Zhang et al. 2016) and synthetic data generation with marginal-based algorithms like PGM (McKenna et al. 2019) have been shown to perform very well on tabular synthetic data generation tasks (Tao et al. 2021).

The Gaussian mechanism adds noise uniformly to all queries, making small queries relatively more noisy. This may reduce the accuracy of query-based algorithms like NAPSU-MQ and the others we examined for groups with rare combinations of data values, such as minorities.

Although we left query selection outside the scope of this paper, selecting the right queries to support downstream analysis is important, as NA+MI cannot guarantee confidence interval coverage if the selected queries do not contain enough information for the downstream task. We plan to study whether existing methods giving confidence bounds on query accuracy (McKenna et al. 2022; Nixon et al. 2022) can be adapted to give confidence intervals for arbitrary downstream analyses.

The runtime is a significant limitation in the current implementation of NAPSU-MQ when using NUTS. As NAPSU-MQ is compatible with any non-DP posterior sampling method, recent (Hoffman et al. 2021) and future advances in MCMC and other sampling techniques are likely able to cut down on the runtime.

Conclusion The analysis of DP synthetic data has not received much attention in existing research. Our work patches a major hole in the current generation and analysis methods by developing the NA+MI pipeline that allows computing accurate confidence intervals and p-values from DP synthetic data. We develop the NAPSU-MQ algorithm in order to implement NA+MI on nontrivial discrete datasets. NA+MI only depends on noise-aware posterior inference, not NAPSU-MQ specifically, and can thus be extended to other settings in the future. With the noise-aware inference algorithm, NA+MI allows conducting valid statistical analyses that include uncertainty estimates with DP synthetic data, potentially unlocking existing privacy-sensitive datasets for widespread analysis.

References

Abbeel, P., D. Koller, and A. Y. Ng (2006). "Learning Factor Graphs in Polynomial Time and Sample Complexity". In: *Journal of Machine Learning Research* 7, pp. 1743–1788.

Aydore, S., W. Brown, M. Kearns, K. Kenthapadi, L. Melis, A. Roth, and A. A. Siva (2021). "Differentially Private Query Release Through Adaptive Projection". In: *Proceedings of the 38th International Conference on Machine Learning*. Vol. 139. Proceedings of Machine Learning Research. PMLR, pp. 457–467.

Balle, B. and Y.-X. Wang (2018). "Improving the Gaussian Mechanism for Differential Privacy: Analytical Calibration and Optimal Denoising". In: *Proceedings of the 35th International Conference on Machine Learning*. Vol. 80. Proceedings of Machine Learning Research. PMLR, pp. 394–403.

Bernstein, G., R. McKenna, T. Sun, D. Sheldon, M. Hay, and G. Miklau (2017). "Differentially Private Learning of Undirected Graphical Models Using Collective Graphical Models". In: *Proceedings of the 34th International Conference on Machine Learning*. Vol. 70. Proceedings of Machine Learning Research. PMLR, pp. 478–487.

Bernstein, G. and D. Sheldon (2018). "Differentially Private Bayesian Inference for Exponential Families". In: *Advances in Neural Information Processing Systems*. Vol. 31, pp. 2924–2934.

Bernstein, G. and D. Sheldon (2019). "Differentially Private Bayesian Linear Regression". In: *Advances in Neural Information Processing Systems*. Vol. 32, pp. 523–533.

Cai, K., X. Lei, J. Wei, and X. Xiao (2021). "Data Synthesis via Differentially Private Markov Random Fields". In: *Proceedings of the VLDB Endowment* 14.11, pp. 2190–2202.

Charest, A.-S. (2010). "How Can We Analyze Differentially-Private Synthetic Datasets?" In: *Journal of Privacy and Confidentiality* 2.2.

Chen, D., T. Orekondy, and M. Fritz (2020). "GS-WGAN: A Gradient-Sanitized Approach for Learning Differentially Private Generators". In: *Advances in Neural Information Processing Systems*. Vol. 33, pp. 12673–12684.

Chen, R., Q. Xiao, Y. Zhang, and J. Xu (2015). "Differentially Private High-Dimensional Data Publication via Sampling-Based Inference". In: *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, pp. 129–138.

Covington, C., X. He, J. Honaker, and G. Kamath (2021). *Unbiased Statistical Estimation and Valid Confidence Intervals Under Differential Privacy*. arXiv Preprint. arXiv: 2110.14465.

Dwork, C. (2008). "Differential Privacy: A Survey of Results". In: *International Conference on Theory and Applications of Models of Computation*. Springer, pp. 1–19.

Dwork, C., K. Kenthapadi, F. McSherry, I. Mironov, and M. Naor (2006). "Our Data, Ourselves: Privacy Via Distributed Noise Generation". In: *Advances in Cryptology - EUROCRYPT*. Vol. 4004. Lecture Notes in Computer Science. Springer, pp. 486–503.

Dwork, C., F. McSherry, K. Nissim, and A. D. Smith (2006). "Calibrating Noise to Sensitivity in Private Data Analysis". In: *Third Theory of Cryptography Conference*. Vol. 3876. Lecture Notes in Computer Science. Springer, pp. 265–284.

Dwork, C. and A. Roth (2014). "The Algorithmic Foundations of Differential Privacy". In: *Foundations and Trends in Theoretical Computer Science* 9.3-4, pp. 211–407.

Evans, G. and G. King (2022). "Statistically Valid Inferences from Differentially Private Data Releases, with Application to the Facebook URLs Dataset". In: *Political Analysis*, pp. 1–21.

Ferrando, C., S. Wang, and D. Sheldon (2022). "Parametric Bootstrap for Differentially Private Confidence Intervals". In: *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics*. Vol. 151. Proceedings of Machine Learning Research. PMLR, pp. 1598–1618.

Gelman, A., J. B. Carlin, H. S. Stern, D. B. Dunson, A. Vehtari, and D. B. Rubin (2014). *Bayesian Data Analysis*. Third edition. Chapman & Hall/CRC Texts in Statistical Science Series. Boca Raton: CRC Press.

Ghalebikesabi, S., H. Wilde, J. Jewson, A. Doucet, S. J. Vollmer, and C. Holmes (2021). *Bias Mitigated Learning from Differentially Private Synthetic Data: A Cautionary Tale*. arXiv Preprint. arXiv: 2108.10934.

Gong, R. (2022). "Exact Inference with Approximate Computation for Differentially Private Data via Perturbations". In: *Journal of Privacy and Confidentiality* 12.2.

Harder, F., K. Adamczewski, and M. Park (2021). "DP-MERF: Differentially Private Mean Embeddings with RandomFeatures for Practical Privacy-preserving Data Generation". In: *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*. Vol. 130. Proceedings of Machine Learning Research. PMLR, pp. 1819–1827.

Hardt, M., K. Ligett, and F. McSherry (2012). "A Simple and Practical Algorithm for Differentially Private Data Release". In: *Advances in Neural Information Processing Systems*. Vol. 25, pp. 2348–2356.

Heikkilä, M. A., J. Jälkö, O. Dikmen, and A. Honkela (2019). "Differentially Private Markov Chain Monte Carlo". In: *Advances in Neural Information Processing Systems*. Vol. 32, pp. 4115–4125.

Hoffman, M. D. and A. Gelman (2014). "The No-U-Turn Sampler: Adaptively Setting Path Lengths in Hamilto-

nian Monte Carlo." In: *Journal of Machine Learning Research* 15.1, pp. 1593–1623.

Hoffman, M. D., A. Radul, and P. Sountsov (2021). "An Adaptive-MCMC Scheme for Setting Trajectory Lengths in Hamiltonian Monte Carlo". In: *The 24th International Conference on Artificial Intelligence and Statistics*. Vol. 130. Proceedings of Machine Learning Research. PMLR, pp. 3907–3915.

Jälkö, J., A. Honkela, and O. Dikmen (2017). "Differentially Private Variational Inference for Non-conjugate Models". In: *Proceedings of the Thirty-Third Conference on Uncertainty in Artificial Intelligence*. AUAI Press.

Jälkö, J., E. Lagerspetz, J. Haukka, S. Tarkoma, A. Honkela, and S. Kaski (2021). "Privacy-Preserving Data Sharing via Probabilistic Modeling". In: *Patterns* 2.7, p. 100271.

Jaynes, E. T. (1957). "Information Theory and Statistical Mechanics". In: *Physical Review* 106.4, pp. 620–630.

Kairouz, P., S. Oh, and P. Viswanath (2015). "The Composition Theorem for Differential Privacy". In: *Proceedings of the 32nd International Conference on Machine Learning*. Vol. 37. Proceedings of Machine Learning Research. PMLR, pp. 1376–1385.

Kohavi, R. and B. Becker (1996). *Adult*. UCI Machine Learning Repository.

Koller, D. and N. Friedman (2009). *Probabilistic Graphical Models: Principles and Techniques*. MIT press.

Kulkarni, T., J. Jälkö, A. Koskela, S. Kaski, and A. Honkela (2021). "Differentially Private Bayesian Inference for Generalized Linear Models". In: *Proceedings of the 38th International Conference on Machine Learning*. Vol. 139. Proceedings of Machine Learning Research. PMLR, pp. 5838–5849.

Liew, S. P., T. Takahashi, and M. Ueno (2022). "PEARL: Data Synthesis via Private Embeddings and Adversarial Reconstruction Learning". In: *International Conference on Learning Representations*.

Liu, F. (2022). "Model-Based Differentially Private Data Synthesis and Statistical Inference in Multiple Synthetic Datasets". In: *Trans. Data Priv.* 15.3, pp. 141–175.

Liu, T., G. Vietri, and S. Z. Wu (2021). "Iterative Methods for Private Synthetic Data: Unifying Framework and New Methods". In: *Advances in Neural Information Processing Systems*. Vol. 34, pp. 690–702.

Long, Y., B. Wang, Z. Yang, B. Kailkhura, A. Zhang, C. Gunter, and B. Li (2021). "G-PATE: Scalable Differentially Private Data Generator via Private Aggregation of Teacher Discriminators". In: *Advances in Neural Information Processing Systems*. Vol. 34, pp. 2965–2977.

Marshall, A., D. G. Altman, R. L. Holder, and P. Royston (2009). "Combining Estimates of Interest in Prognostic Modelling Studies after Multiple Imputation: Current Practice and Guidelines". In: *BMC Medical Research Methodology* 9.1, p. 57.

McKenna, R., G. Miklau, M. Hay, and A. Machanavajjhala (2018). "Optimizing Error of High-Dimensional Statistical Queries under Differential Privacy". In: *Proceedings of the VLDB Endowment* 11.10.

McKenna, R., G. Miklau, and D. Sheldon (2021). "Winning the NIST Contest: A Scalable and General Approach to Differentially Private Synthetic Data". In: *Journal of Privacy and Confidentiality* 11.3.

McKenna, R., B. Mullins, D. Sheldon, and G. Miklau (2022). "AIM: An Adaptive and Iterative Mechanism for Differentially Private Synthetic Data". In: *Proceedings of the VLDB Endowment* 15.11, pp. 2599–2612.

McKenna, R., D. Sheldon, and G. Miklau (2019). "Graphical-Model Based Estimation and Inference for Differential Privacy". In: *Proceedings of the 36th International Conference on Machine Learning*. Vol. 97. Proceedings of Machine Learning Research. PMLR, pp. 4435–4444.

Meek, C., B. Thiesson, and D. Heckerman (2001). *US Census Data (1990)*. UCI Machine Learning Repository.

Neunhoeffer, M., S. Wu, and C. Dwork (2021). "Private Post-GAN Boosting". In: *International Conference on Learning Representations*.

Nixon, M., A. Barrientos, J. Reiter, and A. Slavkovic (2022). "A Latent Class Modeling Approach for Generating Synthetic Data and Making Posterior Inferences from Differentially Private Counts". In: *Journal of Privacy and Confidentiality* 12.1.

Raab, G. M., B. Nowok, and C. Dibben (2018). "Practical Data Synthesis for Large Samples". In: *Journal of Privacy and Confidentiality* 7.3, pp. 67–97.

Raghunathan, T. E., J. P. Reiter, and D. B. Rubin (2003). "Multiple Imputation for Statistical Disclosure Limitation". In: *Journal of Official Statistics* 19.1, p. 1.

Reiter, J. P. (2002). "Satisfying Disclosure Restrictions with Synthetic Data Sets". In: *Journal of Official Statistics* 18.4, p. 531.

Reiter, J. P. (2005). "Significance Tests for Multi-Component Estimands from Multiply Imputed, Synthetic Microdata". In: *Journal of Statistical Planning and Inference* 131.2, pp. 365–377.

Rubin, D. B. (1987). "Multiple Imputation for Nonresponse in Surveys". In: *Multiple Imputation for Nonresponse in Surveys*. New York: John Wiley \& Sons.

Rubin, D. B. (1993). "Discussion: Statistical Disclosure Limitation". In: *Journal of Official Statistics* 9.2, pp. 461–468.

Si, Y. and J. P. Reiter (2011). "A Comparison of Posterior Simulation and Inference by Combining Rules for Multiple Imputation". In: *Journal of Statistical Theory and Practice* 5.2, pp. 335–347.

Tao, Y., R. McKenna, M. Hay, A. Machanavajjhala, and G. Miklau (2021). *Benchmarking Differentially Private Synthetic Data Generation Algorithms*. arXiv Preprint. arXiv: 2112.09238.

Vietri, G., G. Tian, M. Bun, T. Steinke, and Z. S. Wu (2020). "New Oracle-Efficient Algorithms for Private Synthetic Data Release". In: *Proceedings of the 37th International Conference on Machine Learning*. Vol. 119. Proceedings of Machine Learning Research. PMLR, pp. 9765–9774.

Wainwright, M. J. and M. I. Jordan (2008). "Graphical Models, Exponential Families, and Variational Inference". In: *Foundations and Trends in Machine Learning* 1.1-2, pp. 1–305.

Wilde, H., J. Jewson, S. J. Vollmer, and C. Holmes (2021). "Foundations of Bayesian Learning from Synthetic Data". In: *The 24th International Conference on Artificial Intelligence and Statistics*. Vol. 130. Proceedings of Machine Learning Research. PMLR, pp. 541–549.

Xie, L., K. Lin, S. Wang, F. Wang, and J. Zhou (2018). *Differentially Private Generative Adversarial Network*. arXiv Preprint. arXiv: 1802.06739.

Yildirim, S. and B. Ermis (2019). "Exact MCMC with Differentially Private Moves - Revisiting the Penalty Algorithm in a Data Privacy Framework". In: *Statistics and Computing* 29.5, pp. 947–963.

Yoon, J., J. Jordon, and M. van der Schaar (2019). "PATE-GAN: Generating Synthetic Data with Differential Privacy Guarantees". In: *International Conference on Learning Representations*.

Zhang, J., G. Cormode, C. M. Procopiuc, D. Srivastava, and X. Xiao (2017). "PrivBayes: Private Data Release via Bayesian Networks". In: *ACM Transactions on Database Systems* 42.4, 25:1–25:41.

Zhang, J., X. Xiao, and X. Xie (2016). "PrivTree: A Differentially Private Algorithm for Hierarchical Decompositions". In: *Proceedings of the 2016 International Conference on Management of Data, SIGMOD Conference*. ACM, pp. 155–170.

Zheng, S. (2015). "The Differential Privacy of Bayesian Inference". Bachelor's Thesis. Harvard College.

Zhou, X. and J. P. Reiter (2010). "A Note on Bayesian Inference After Multiple Imputation". In: *The American Statistician* 64.2, pp. 159–163.

A PROOF OF THEOREM 3.5

Theorem 3.5. *Let a be the concatenation of n_s full sets of marginal queries. Then $\Delta_2 a \leq \sqrt{2n_s}$.*

Proof. Let $a_1, \ldots, a_{n_s}$ be the full sets of marginal queries that form a. Because all of the queries of a_i have the same set of variables, the vector $a_i(x)$ has a single component of value 1, and the other components are 0 for any $x \in \mathcal{X}$. Then, for any neighbouring $X, X' \in \mathcal{X}^n$, $||a_i(X) - a_i(X')||_2^2 \leq 2$. Then

$$\Delta_2 a = \sup_{X \sim X'} ||a(X) - a(X')||_2 = \sup_{X \sim X'} \sqrt{\sum_{i=1}^{n_s} ||a_i(X) - a_i(X')||_2^2} \leq \sup_{X \sim X'} \sqrt{\sum_{i=1}^{n_s} 2} = \sqrt{2n_s} \tag{8}$$

$\square$

B MULTIPLE IMPUTATION

In order to compute uncertainty estimates for downstream analyses from the noise-aware posterior with NA+MI, we use Rubin's rules for synthetic data (Raghunathan et al. 2003; Reiter 2002).

After the synthetic datasets X_i^{Syn} for $1 \leq i \leq m$ are released by the data holder, the data analyst runs their downstream analysis on each X_i^{Syn}. For each synthetic dataset, the analysis produces a point estimate q_i and a variance estimate v_i for q_i.

The estimates $q_1, \ldots, q_m$ and $v_1, \ldots, v_m$ are combined as follows (Raghunathan et al. 2003):

$$\bar{q} = \frac{1}{m} \sum_{i=1}^{m} q_i, \quad \bar{v} = \frac{1}{m} \sum_{i=1}^{m} v_i, \quad b = \frac{1}{m-1} \sum_{i=1}^{m} (q_i - \bar{q})^2. \tag{9}$$

We use $\bar{q}$ as the combined point estimate, and set

$$T = \left(1 + \frac{1}{m}\right) b - \bar{v}, \qquad T^* = \begin{cases} T & \text{if } T \geq 0 \\ \frac{n_{Syn}}{n} \bar{v} & \text{otherwise.} \end{cases} \tag{10}$$

T is an estimate of the combined variance. T can be negative, which is corrected using T^* instead (Reiter 2002).

We compute confidence intervals and hypothesis tests using the t-distribution with mean $\bar{q}$, variance T^*, and degrees of freedom

$$\nu = (m-1)(1 - r^{-1})^2, \tag{11}$$

where $r = (1 + \frac{1}{m})\frac{b}{\bar{v}}$ (Reiter 2002).

These combining rules apply when q is a univariate estimate. Reiter (2005) derives appropriate combining rules for multivariate estimates, which can be applied with NA+MI.

Rubin's rules make many assumptions on the different distributions that are involved (Raghunathan et al. 2003; Si and Reiter 2011), such as the normality of the distribution of q_i when sampling data from the population. These assumptions may not hold for some types of estimates, such as probabilities (Marshall et al. 2009) or quantile estimates (Zhou and Reiter 2010). Further work (Si and Reiter 2011) tries to reduce these assumptions, especially in the context of missing data. Their results for synthetic data generation can be applied with our method.

Si and Reiter (2011) propose to remove some of these assumptions by approximating the integral that Rubin's rules are derived from by sampling instead of using the analytical approximations in (9) and (10). They find that their sampling-based approximation can be effective, especially with a small number of datasets, but is computationally more expensive.

B.1 Unbiasedness of Rubin's Rules

Rubin's rules make several assumptions on the downstream analysis method, and several normal approximations when deriving the rules. Raghunathan et al. (2003) derive conditions under which Rubin's rules give an unbiased estimate.

Rubin's rules aim to estimate a quantity Q of the entire population P, of which X is a sample. Conceptually, the sampling of the synthetic datasets is done in two stages: first, synthetic populations P_i^{Syn} for $1 \leq i \leq m$ are sampled. Second, a synthetic dataset X_i^{Syn} is sampled i.i.d from P_i^{Syn}. This is equivalent to the sampling process for X_i^{Syn} described in Section 2, and makes stating the assumptions of Rubin's rules easier.

While the sampling process of synthetic data has to be i.i.d., this is not required for the original data. This means that there are two versions of the downstream analysis: the one for i.i.d. data, and the one for the complex sampling method of the real data. The latter method is not used at any point, so it is assumed to exist for the theory, but does not have to be practically implementable.

We take advantage of this handling of complex sampling of real data by including the computation of s and adding noise to get $\tilde{s}$ in the sampling scheme, so we are considering $\tilde{s}$ to be the original data from the point of view of the theory. The fact that $\tilde{s}$ is a noisy summary statistic instead of a dataset is not an issue, as the theory only requires having a theoretical method to estimate Q from $\tilde{s}$. If the chosen marginal queries contain the relevant information for the downstream task, so s is a (approximate) sufficient statistic, this theoretical method will exist.

Let Q_i denote the quantity of interest Q computed from the synthetic population P_i^{Syn} instead of P. Let V_i denote the sampling variance of q_i from the synthetic population P_i^{Syn}. Note that q_i and it's variance estimate v_i are obtained using the downstream analysis method for i.i.d. data. Let $\hat{Q}_D$ and $\hat{U}_D$ be the point and variance estimates of Q derived from $\tilde{s}$ when sampling the population P, which are obtained using the theoretical inference method for complex samples.

Now the assumptions of Raghunathan et al. (2003) are

Assumption B.1. *For all $1 \leq i \leq m$, q_i is unbiased for Q_i and asymptotically normal with respect to sampling from the synthetic population P_i^{Syn}, with sampling variance V_i.*

Assumption B.2. *For all $1 \leq i \leq m$, v_i is unbiased for V_i, and the sampling variability in v_i is negligible. That is $v_i \mid P_i^{Syn} \approx V_i$. Additionally, the variation in V_i across the synthetic populations is negligible.*

Assumptions B.1-B.2 ensure that the downstream analysis method used to estimate Q is accurate, for both point and variance estimates, when applied to i.i.d. real data, regardless of the population.

Assumption B.3. $\hat{Q}_D \mid P \sim \mathcal{N}(Q, \hat{U}_D)$

Assumption B.3 ensures that the analysis for complex sampling is accurate for point and variance estimates when applied to the real population.

Assumption B.4. $Q_i \mid \tilde{s} \sim \mathcal{N}(\hat{Q}_D, \hat{U}_D)$

Assumption B.4 requires that the generation of synthetic datasets does not bias the downstream analysis.

For query-based methods like NAPSU-MQ, Assumptions B.3 and B.4 may not hold when the queries do not contain the relevant information for the downstream task.

With Assumptions B.1-B.4, Raghunathan et al. (2003) show that $\bar{q}$ is an unbiased estimate of Q, and T is an asymptotically unbiased variance estimate.

Theorem B.5 (Raghunathan et al. (2003)). *Assumptions B.1-B.4 imply that*

1. *$E(\bar{q} \mid P) = Q$,*

2. *$E(T \mid P) = \text{Var}(\bar{q} \mid P)$,*

3. *Asymptotically $\frac{\bar{q} - Q}{\sqrt{T}} \sim \mathcal{N}(0, 1)$,*

4. *For moderate m, $\frac{\bar{q} - Q}{\sqrt{T}} \sim t_\nu(0, 1)$ (Reiter 2002).*

C FINDING AND IDENTIFIABLE PARAMETRISATION

In this section, we describe the process we use to ensure the parametrisation of the posterior in NAPSU-MQ is identifiable. We ensure identifiability by dropping some of the selected queries, chosen using the the canonical parametrisation of MED_θ to ensure no information is lost. First, we give some background on Markov networks, which is necessary to understand the canonical parametrisation.

Markov Networks A Markov network is a representation of a probability distribution that is factored according to an undirected graph. Specifically, a Markov network distribution P is a product of *factors*. A factor is a function from a subset of the variables to non-negative real numbers. The subset of variables is called the *scope* of the factor. The joint distribution is given by

$$P(x) = \frac{1}{Z} \prod_{I \subset S} \phi_I(x_I) \tag{12}$$

where S is the set of scopes for the factors. The undirected graph is formed by representing each variable as a node, and adding edges such that the scope of each factor is a clique in the graph.

Canonical Parametrisation The canonical parametrisation is given in terms of *canonical factors* (Abbeel et al. 2006). The canonical factors depend on an arbitrary assignment of variables x^*. We simply choose $x^* = (0, \ldots, 0)$. In the following, x_U denotes the selection of components in the set U from the vector x, and x_{-U} denotes the selection of all components except those in U.

Definition C.1. *A canonical factor ϕ_D^* with scope D is defined as*

$$\phi_D^*(x) = \exp\left(\sum_{U \subseteq D} (-1)^{|D-U|} \ln P(x_U, x_{-U}^*)\right)$$

The sum is over all subsets of D, including D itself and the empty set. $|D - U|$ is the size of the set difference of D and U.

Theorem C.2 (Abbeel et al. (2006)(Theorem 3))**.** *Let P be a Markov network with factor scopes S. Let $S^* = \cup_{D \in S} \mathcal{P}(D) - \emptyset$. Then*

$$P(x) = P(x^*) \prod_{D^* \in S^*} \phi_{D^*}^*(x_{D^*})$$

There are more canonical factors than original factors, so it might seem that there are more parameters in the canonical parametrisation than in the original parametrisation. However, many values in the canonical factors turn out to be ones. We can select the queries corresponding to non-one canonical factor values to obtain a set of queries with the same information as the original queries, but without linear dependencies (Koller and Friedman 2009). We call this set of queries the *canonical queries*.

Many of the canonical factor scopes are subsets of the original factor scopes, so using the canonical queries as is would introduce new marginal query sets and potentially increase the sensitivity of the queries. As all of the new queries are sums of existing queries, we can replace each new query with the old queries that sum to the new query, and use the same θ value for all of the added queries to preserve identifiability. After the replacements, the queries are a subset of the original non-canonical queries, so their sensitivity is at most the original sensitivity. If one of the added queries was already included, it does not need to be added again, because two instances of a single query can be collapsed into a single instance with it's own parameter value. Because of this, we did not need to fix the θ values of any queries to the same value in the settings we studied.

D NAPSU-MQ VS. PGM

The PGM algorithm (McKenna et al. 2019) generates synthetic data based on the same marginal queries a and noise addition as NAPSU-MQ. PGM also models the original data using the MED_θ distribution. Unlike NAPSU-MQ, PGM finds the parameters θ by minimising the l_2-distance $||\tilde{s} - n\mu(\theta)||_2$ between the observed noisy query values $\tilde{s}$ and the expected query values $n\mu(\theta) = nE_{x \sim \text{MED}_\theta}(a(x))$ In the following, we'll replace the query values s and $\tilde{s}$ that are summed over datapoints with $u = \frac{s}{n}$ and $\tilde{u} = \frac{\tilde{s}}{n}$ that represent mean query values over datapoints. Then the PGM objective is equivalent to $||\tilde{u} - \mu(\theta)||_2$.

We can view the PGM minimisation problem as a maximum likelihood estimation in the NAPSU-MQ probabilistic model

$$X \sim \text{MED}_\theta^n, \quad s = a(X), \quad \tilde{s} \sim \mathcal{N}(s, \sigma_{DP}^2 I), \tag{13}$$

where we replace normal approximation that NAPSU-MQ uses with a law of large numbers approximation. Specifically, first replace s with u in (13):

$$X \sim \text{MED}_\theta^n, \quad u = \frac{a(X)}{n}, \quad \tilde{u} \sim \mathcal{N}(u, \sigma_{DP}^2 I/n^2). \tag{14}$$

Because u is a mean of sufficient statistics for individual datapoints, by the law of large numbers, asymptotically $u \sim \delta_{\mu(\theta)}$. With this approximation, the probabilistic model is

$$u \sim \delta_{\mu(\theta)}, \quad \tilde{u} \sim \mathcal{N}(u, \sigma_{DP}^2 I/n^2). \tag{15}$$

u can be marginalised from the likelihood of this model:

$$p(\tilde{u}|\theta) = \int p(\tilde{u}, u|\theta)\mathrm{d}u \tag{16}$$

$$= \int p(\tilde{u}|u)p(u|\theta)\mathrm{d}u \tag{17}$$

$$= \int \mathcal{N}(\tilde{u}|u, \sigma_{DP}^2 I/n^2)\delta_{\mu(\theta)}(u)\mathrm{d}u \tag{18}$$

$$= \mathcal{N}(\tilde{u}|\mu(\theta), \sigma_{DP}^2 I/n^2) \tag{19}$$

The marginalised log-likelihood is then

$$\ln p(\tilde{u}|\theta) = -\frac{n^2}{\sigma_{DP}^2}||\tilde{u} - \mu(\theta)||_2^2 + \text{constant}, \tag{20}$$

so maximising the log-likelihood is equivalent to minimising the PGM objective.

If we made a normal approximation instead of the law of large numbers approximation in (14), we would get

$$\tilde{u} \sim \mathcal{N}(\mu(\theta), \Sigma(\theta)/n + \sigma_{DP}^2 I/n^2), \tag{21}$$

so maximising the likelihood is still possible. Unlike PGM, this maximum likelihood objective includes the covariance $\Sigma(\theta)$. We leave any comparisons between maximising this objective and PGM to future work.

E HYPERPARAMETERS

NAPSU-MQ The hyperparameters of NAPSU-MQ are the choice of prior, choice of inference algorithm, and the parameters of that algorithm. For the toy data experiment, we used the Laplace approximation for inference, which approximates the posterior with a Gaussian centered at the maximum aposteriori estimate (MAP). We find the MAP for the Laplace approximation with the LBFGS optimisation algorithm, which we run until the loss improves by less than 10^{-5} in an iteration, up to a maximum of 500 iterations. Sometimes LBFGS failed to converge, which we detect by checking if the loss increased by over 1000 in one iteration, and fix by restarting optimisation from a different starting point. We also restarted if the maximum number of iterations was reached without convergence. For the vast majority of runs, no restarts were needed, and at most 3 were needed.

For the Adult experiment, we used NUTS (Hoffman and Gelman 2014). We ran 4 chains of 800 warmup samples and 2000 kept samples. We set the maximum tree depth of NUTS to 12. We normalised the posterior using the mean and covariance from the Laplace approximation. For the Laplace approximation, we used the same hyperparameters as with the toy data set, except we set the maximum number of iterations to 6000.

For the prior, we used a Gaussian distribution with mean 0 and standard deviation 10 for all components, without dependencies between components, for both experiments.

PGM and Repeated PGM PGM finds the MED_θ parameters θ that minimise the L_2-error between the expected query values and the noisy query values. The PGM implementation offers several algorithms for this optimisation problem, but we found that the default algorithm (mirror descent) and number of iterations works well for both experiments.

RAP RAP minimises the error on the selected queries of a continuous relaxation of the discrete synthetic dataset. After the optimal relaxed synthetic dataset is found, a discrete synthetic dataset is constructed by sampling. This gives two hyperparameters that control the size of the synthetic data: the size of the continuous dataset, and the number of samples for each datapoint in the continuous relaxation. We set the size of the continuous dataset to 1000 for both experiments, as recommended by the paper (Aydore et al. 2021). For the Adult data experiment, we set the number of samples per datapoint to 46, so that the total size of the synthetic dataset is close to the size of the original dataset. For the toy data experiment,

we set the number of samples per datapoint to 50. The RAP paper (Aydore et al. 2021) finds that much smaller values are sufficient, but higher values should only increase accuracy.

In both cases, we weight the synthetic datapoints by $\frac{n}{n_{Syn}}$ before the downstream logistic regression to ensure that the logistic regression does not over- or underestimate variances because of a different sample size from the original data.

RAP also has two other hyperparameters that are relevant in our experiments: the number of iterations and the learning rate for the query error minimisation. After preliminary runs, we set the learning rate at 0.1 for both experiments, and set the number of iterations to 5000 for the toy data experiment, and 10000 for the Adult data experiment.

PEP   PEP has two hyperparameters: the number of iterations used to find a distribution with maximum entropy that has approximately correct query values, and the allowed bound on the difference of the query values. The PEP implementation hardcodes the allowed difference to 0. We set the number of iterations to 1000 after preliminary runs for both experiments.

PrivLCM   PrivLCM samples the posterior of a Bayesian latent class model, where the number of classes in limited to make inference tractable. The model has hyperparameters for the prior, and the number of latent classes. We leave the prior hyperparameters to their defaults, and set the number of latent classes to 10, which the PrivLCM authors used in a 5-dimensional binary data experiment (Nixon et al. 2022). The remaining hyperparameter of PrivLCM is the number of posterior samples that are obtained. To keep the runtime of PrivLCM manageable, we set the number of samples to 500 after ensuring that the lower number of samples did not degrade the accuracy of the estimated probabilities for the joint distribution compared to using the default of 5000 samples.

PrivBayes   As we focus on synthetic data generation, we did not use the query selection features of PrivBayes in the toy data experiment, and instead set the queries according to the Bayesian network of the data generating process. Denoting the three components of the data as $X = (X_1, X_2, X_3)$, the queries are the full set of 1-way marginals on X_1, the full set of 2-way marginals on (X_1, X_2), and the full set of 3-way marginals on all three variables.

F   ADULT EXPERIMENT DETAILS AND EXTRA RESULTS

We include the columns Age, Workclass, Education, Marital Status, Race, Gender, Capital gain, Capital loss, Hours per week and Income of the Adult dataset, and discard the rest to remove redundant columns and keep computation times manageable. We discretise Age and Hours per week to 5 buckets, and discretise Capital gain and Capital loss to binary values indicating whether their value is positive. The Income column is binary from the start, and indicates whether a person has an income $> \$50\,000$.

In the downstream logistic regression, we use income as the dependent variable, and Age, Race and Gender as independent variables. Age is transformed back to a continuous value for the logistic regression by picking the middle value of each discretisation bucket. We did not use all variables for the downstream task, as a smaller set of variables allows including the relevant marginals for synthetic data generation. The regularisation for the logistic regression is l_2 with a very small multiplier of 10^{-5}. When the regularisation is used, variances are estimated with bootstrapping using 50 bootstrap samples.

All of the Adult experiment figures, except Figures S5 and S6 use the small regularisation term. Figure S5 shows the results with the trick of removing large variance estimates ($\geq 10^3$), and Figure S6 shows how many estimates were removed.

For the input queries, we include the 2-way marginals with Hours per week and each of the independent variables Age, Race and Gender and income, as well as the 2-way marginal between Race and Gender. The rest of the queries were selected with the MST algorithm (McKenna et al. 2021). For MST, we used $\epsilon = 0.5$ and $\delta = \frac{1}{n^2} \approx 4.7 \cdot 10^{-10}$, but we do not include this in our figures, as we focus on the synthetic data generation, not query selection. The selected queries are shown in Figure S1. The selection is very stable: in 100 repeats of query selection, these queries were selected 99 times.

We chose the number of generated synthetic datasets for NAPSU-MQ and the number of repeats for repeated PGM by comparing the results of the Adult experiment for different choices. The results are shown in Figure S4 for NAPSU-MQ and Figure S3 for repeated PGM. We chose $m = 100$ for NAPSU-MQ because it produces the narrowest confidence intervals, and $m = 5$ repeats for repeated PGM because it had the best calibration overall.

Table S1: Runtimes of each inference run for the Adult experiment. Does not include the time taken to generate synthetic data, or run any downstream analysis. The LA rows record the runtime for obtaining the Laplace approximation for NAPSU-MQ that is used in the NUTS inference, so the total runtime for a NAPSU-MQ run with NUTS is the sum of the LA and NUTS rows. Experiments were run on 4 CPU cores of a cluster.

| Algorithm	Epsilon	Mean	Standard Deviation
LA	0.1	2 min 53 s	18.5 s
	0.3	3 min 53 s	29.4 s
	0.5	3 min 38 s	35.0 s
	1.0	3 min 25 s	25.5 s
NUTS	0.1	9 h 59 min 6 s	6506 s
	0.3	7 h 33 min 28 s	2701 s
	0.5	4 h 57 min 40 s	3185 s
	1.0	3 h 51 min 34 s	1274 s
PEP	0.1	6 min 50 s	25.4 s
	0.3	7 min 18 s	31.2 s
	0.5	7 min 0 s	33.1 s
	1.0	7 min 7 s	33.7 s
PGM	0.1	15 s	0.5 s
	0.3	17 s	1.5 s
	0.5	15 s	0.4 s
	1.0	15 s	0.6 s
PGM-repeat-10	0.1	2 min 35 s	3.3 s
	0.3	2 min 53 s	13.0 s
	0.5	2 min 37 s	5.0 s
	1.0	2 min 36 s	4.4 s
PGM-repeat-20	0.1	5 min 15 s	10.9 s
	0.3	5 min 58 s	28.4 s
	0.5	5 min 10 s	10.2 s
	1.0	5 min 13 s	12.6 s
PGM-repeat-5	0.1	1 min 17 s	2.7 s
	0.3	1 min 28 s	6.7 s
	0.5	1 min 18 s	2.6 s
	1.0	1 min 18 s	1.9 s
RAP	0.1	32 s	2.4 s
	0.3	34 s	2.2 s
	0.5	32 s	2.1 s
	1.0	31 s	2.1 s
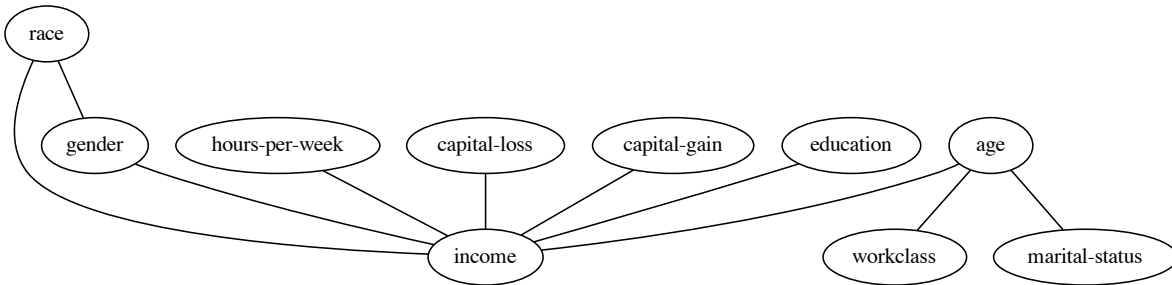
Figure S1: Markov network of selected queries for the Adult experiment. Each edge in the graph represents a selected 2-way marginal.
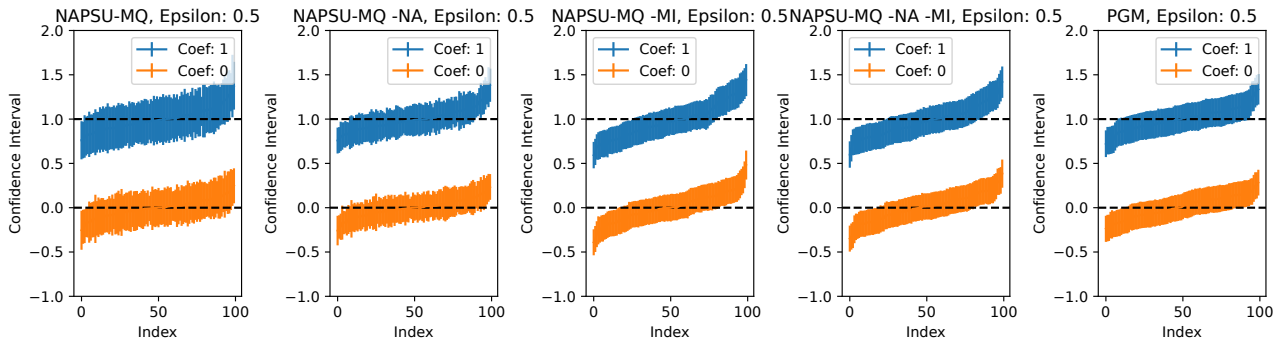

Figure S2: Ablation Study Confidence Intervals.

G   US CENSUS DATA EXPERIMENT

We conducted an additional experiment on US Census data from the UCI repository (Meek et al. 2001). We limited the data to individuals who have served in the US Military, and picked 9 columns[7], most relating to military service. Even this subset of the data is large, with $n = 320\,754$. All columns are discrete, and have $10\,800$ possible values, much fewer than the Adult experiment.

As the downstream task, we use logistic regression with dPoverty as the dependent variable and iSex, iKorean, iVietnam and iMilitary as the independent variables. dPoverty has three categories, so we combine the two categories denoting people below the powerty line to make the dependent variable binary for the logistic regression, but not synthetic data generation.

As our queries we use 4 three-way marginals covering the independent and dependent variables, and 3 two-way marginals that include the other variables that are synthesised, but not included in the regression. As the published implementation of RAP (Aydore et al. 2021) does not support a mix of two- and three-way marginals, we replace the two-way marginals with three-way marginals for RAP. As in the Adult experiment, we set $\delta = n^{-2} \approx 9.7 \cdot 10^{-12}$, and vary ϵ.

As in the adult experiment, we use $n_{Syn} = n$ for all algorithms except RAP. For PGM-repeat and NAPSU-MQ, we choose m with a preliminary experiment. For NAPSU-MQ, we set $m = 100$, although the differences between the choices are not large. For PGM-repeat, we set $m = 10$. We set the other hyperparameters for all algorithms after testing runs to the same values used in the Adult experiment, except we increased the number of optimisation iterations for PGM to $10\,000$ from the default of 1000, the number of iterations for PEP to $10\,000$ from 1000, and increased the number of kept samples in NUTS to 4000 for NAPSU-MQ. We did not use the trick of dropping estimates with very high variances, or using very small regularisation in the logistic regression with the US Census data.

---

[7] The columns are dYrsserv, iSex, iVietnam, iKorean, iMilitary, dPoverty, iMobillim, iEnglish and iMarital.
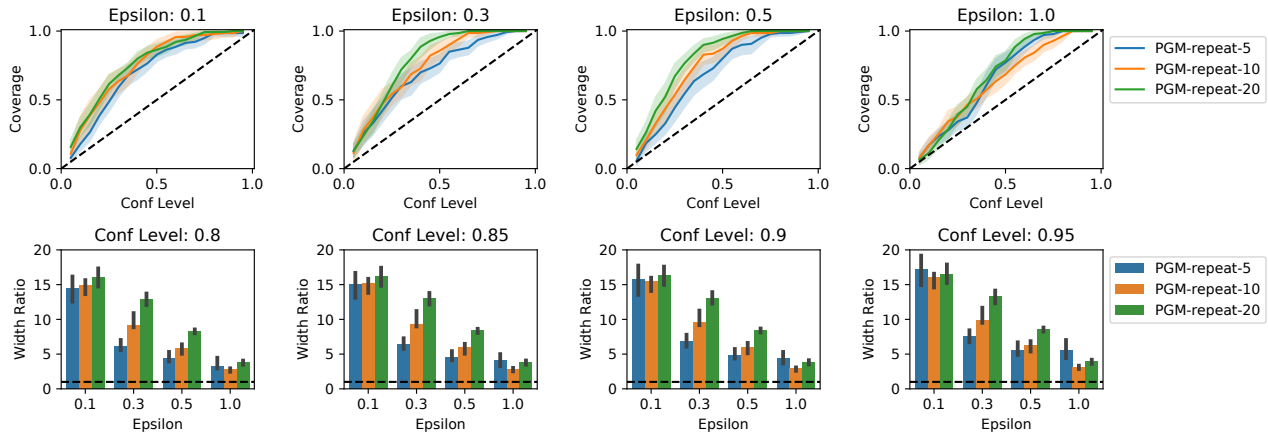
Figure S3: Comparison of different numbers of repetitions for repeated PGM on the Adult dataset with regularisation. We chose $m = 5$ repeats to represent repeated PGM in the main experiment, although the differences between the numbers of repeats are small.
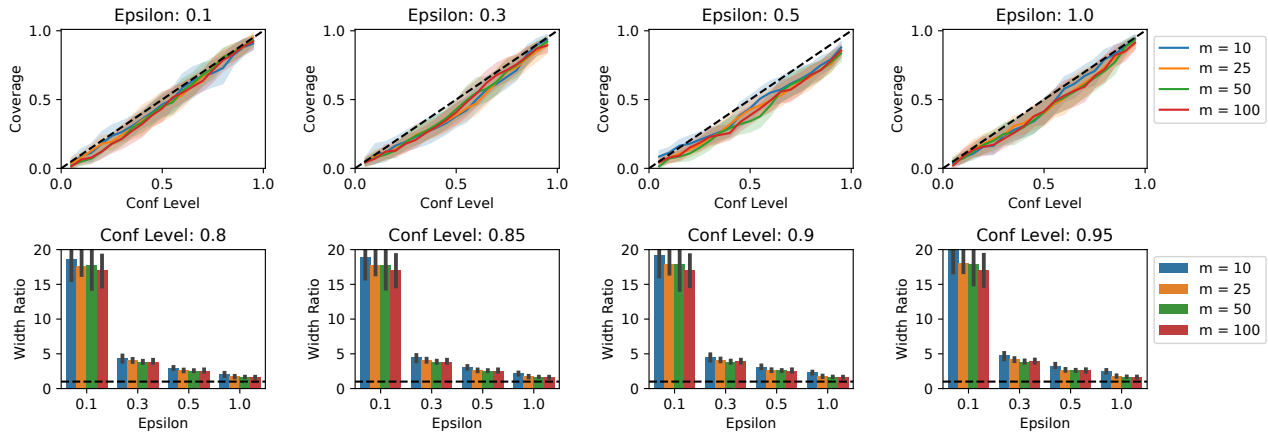

Figure S4: Comparison of different numbers of generated synthetic datasets for NAPSU-MQ on the Adult dataset with regularisation. The differences are small, but $m = 100$ synthetic datasets produces the narrowest intervals, so we chose it for the main experiment.

The results are shown in Figure S9. While PGM is calibrated with $\epsilon \geq 0.3$, it is severely overconfident with $\epsilon = 0.1$. This is likely caused by the large size of the dataset: at larger values of ϵ, there is little noise compared to the large sample size, while at $\epsilon = 0.1$, the noise has a clear effect.

NAPSU-MQ and PGM-repeat are able to produce calibrated results at $\epsilon = 0.1$. Of these, NAPSU-MQ produces clearly narrower confidence intervals for all values of ϵ.

Figure S10 shows the accuracies of the produced synthetic datasets on all 1-way and 2-way marginal queries for the algorithms. As with the Adult dataset, shown in Figure S8, NAPSU-MQ is almost as accurate as PGM, and is equally accurate as PGM-repeat. For $\epsilon \geq 0.5$, all of the aforementioned algorithms are almost as accurate. RAP and PEP are nowhere close to these algorithms in accuracy, having errors that are several times larger than the other algorithms.

For some reason, PEP fails completely with this dataset. We are not sure what causes this, as the algorithm should work in this setting as well as it did with the Adult dataset, and the size of the dataset should not be an issue.

The runtimes for each algorithm are shown in Table S2. The difference between PGM-repeat and NAPSU-MQ is much smaller than in the Adult data experiment, but is still large.
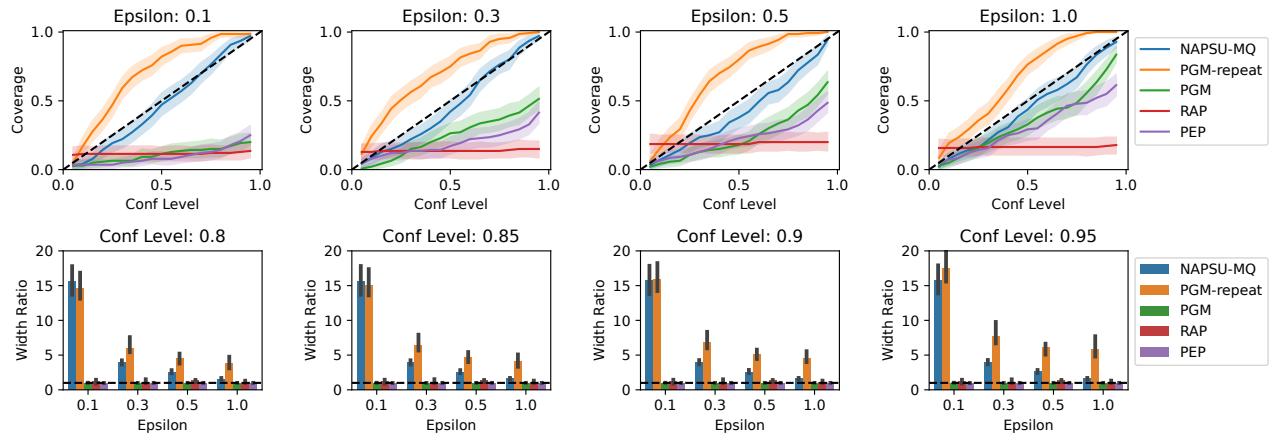
Figure S5: Results from the Adult data experiment with the trick of dropping large variances in the logistic regression instead of adding a small regularisation term. The results are almost identical to Figure 4, except for RAP, which suffers from the regularisation.
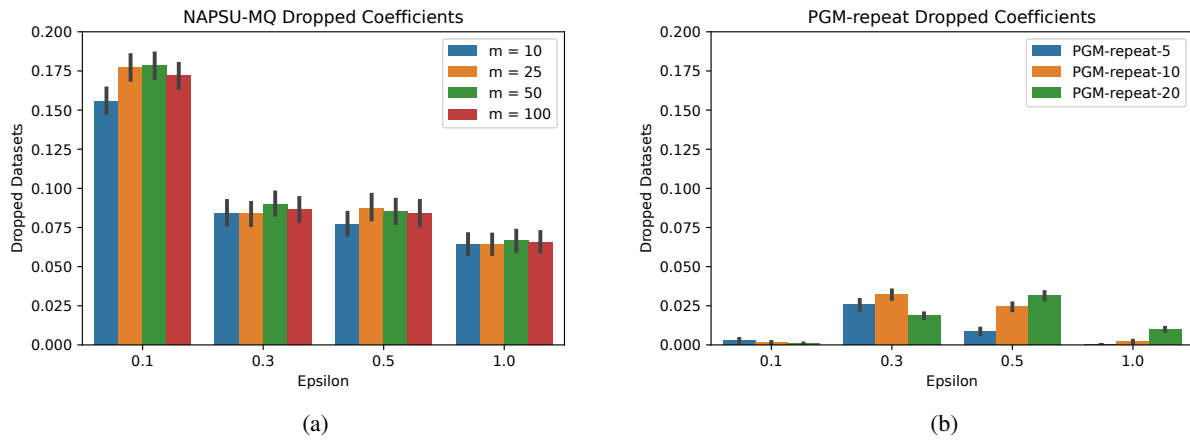

Figure S6: The fraction of coefficients dropped before Rubin's rules because of very high estimated variances from the downstream logistic regression in the Adult data experiment for NAPSU-MQ in (a) and PGM-repeat in (b).
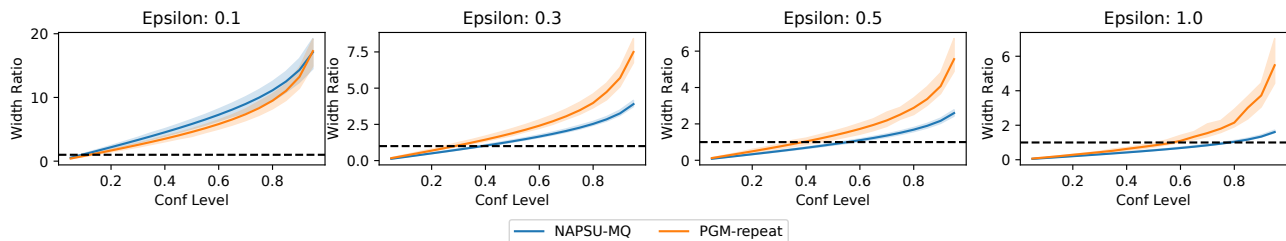

Figure S7: The tradeoff between the confidence level for DP confidence intervals and the width of the intervals on the Adult dataset with regularisation. The width ratio on the y-axis is with regards to the original 95% confidence interval, for all confidence levels, so the plot shows how much must the confidence level drop to obtain the same width from a DP confidence interval as a non-DP one. The horizontal line at $y = 1$ shows this point. For $\epsilon = 1$, the confidence level for NAPSU-MQ must be dropped to about 75%, and for PGM-repeat, it must be dropped to about 50%.
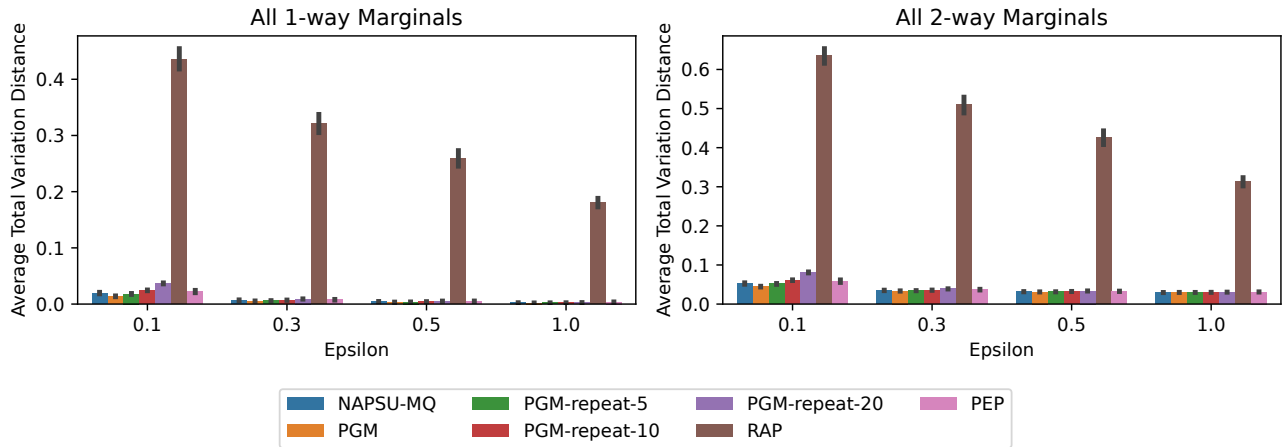
Figure S8: Comparison of marginal query accuracy for Adult data. NAPSU-MQ is almost as accurate as PGM for all values of ϵ, and is on par with PGM-repeat. The panels show the average total variation distance of all 1-way marginal distributions (left) or all 2-way marginal distributions (right) between the original discretised data and synthetic data, averaged over 20 repeats. For NAPSU-MQ and PGM-repeat-m, the synthetic marginal distributions were estimated by averaging over m synthetic datasets, with $m = 100$ for NAPSU-MQ.
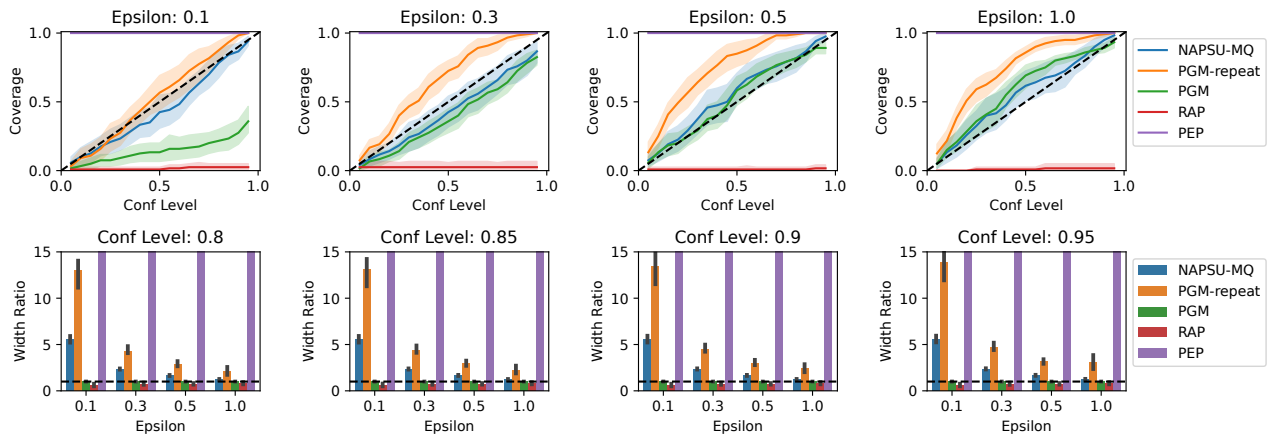

Figure S9: Results for the US Census experiment, showing that only NAPSU-MQ and PGM-repeat are calibrated for all values of ϵ, and NAPSU-MQ produces significantly narrower confidence intervals than PGM-repeat. Like Figure 4, the top row shows the mean coverage over all coefficients and 20 runs for different confidence levels. The bottom row shows median confidence interval widths divided by real data confidence interval widths. $\delta \approx 9.7 \cdot 10^{-12}$ in all panels.

Table S2: Runtimes of each inference run for the US Census experiment. Does not include the time taken to generate synthetic data, or run any downstream analysis. The LA rows record the runtime for obtaining the Laplace approximation for NAPSU-MQ that is used in the NUTS inference, so the total runtime for a NAPSU-MQ run with NUTS is the sum of the LA and NUTS rows. Experiments were run on 4 CPU cores of a cluster.

| Algorithm	Epsilon	Mean	Standard Deviation
LA	0.1	2 min 2 s	76.8 s
	0.3	1 min 35 s	23.8 s
	0.5	1 min 51 s	44.0 s
	1.0	1 min 51 s	44.8 s
NUTS	0.1	3 h 32 min 25 s	2836 s
	0.3	1 h 57 min 45 s	989 s
	0.5	1 h 31 min 15 s	951 s
	1.0	1 h 8 min 6 s	477 s
PEP	0.1	17 s	0.6 s
	0.3	18 s	1.0 s
	0.5	17 s	0.4 s
	1.0	17 s	0.5 s
PGM	0.1	1 min 57 s	2.8 s
	0.3	1 min 58 s	3.2 s
	0.5	1 min 57 s	4.2 s
	1.0	1 min 57 s	3.2 s
PGM-repeat-10	0.1	19 min 22 s	28.0 s
	0.3	19 min 18 s	21.9 s
	0.5	19 min 36 s	33.9 s
	1.0	19 min 25 s	22.0 s
PGM-repeat-20	0.1	38 min 38 s	50.2 s
	0.3	38 min 59 s	37.5 s
	0.5	38 min 57 s	40.5 s
	1.0	38 min 39 s	74.7 s
PGM-repeat-5	0.1	9 min 45 s	17.6 s
	0.3	9 min 38 s	8.9 s
	0.5	9 min 45 s	12.8 s
	1.0	9 min 43 s	9.8 s
RAP	0.1	28 s	2.4 s
	0.3	28 s	2.0 s
	0.5	27 s	1.2 s
	1.0	27 s	3.5 s
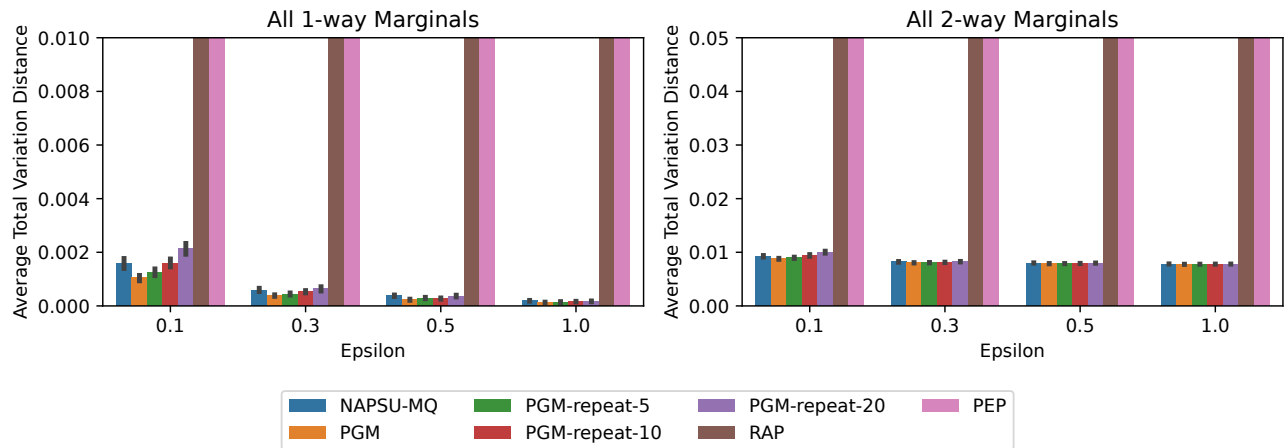
Figure S10: Comparison of marginal query accuracy for US Census data. NAPSU-MQ is almost as accurate as PGM for all values of ϵ, and is on par with PGM-repeat, as with the Adult data in Figure S8. The panels show the average total variation distance of all 1-way marginal distributions (left) or all 2-way marginal distributions (right) between the original discretised data and synthetic data, averaged over 20 repeats. For NAPSU-MQ and PGM-repeat-m, the synthetic marginal distributions were estimated by averaging over m synthetic datasets, with $m = 100$ for NAPSU-MQ. RAP and PEP have average total variation distances over 0.1 for both 1-way and 2-way marginals for all values of ϵ.