Bayesian Pseudocoresets

Dionysis Manousakas
Department of Computer Science and Technology
University of Cambridge
Cambridge, UK, CB3 0FD
dm754@cam.ac.uk

Zuheng (David) Xu
Department of Statistics
University of British Columbia
Vancouver, BC V6T 1Z4
zuheng.xu@stat.ubc.ca

Cecilia Mascolo
Department of Computer Science and Technology
University of Cambridge & The Alan Turing Institute
Cambridge, UK, CB3 0FD
cm542@cam.ac.uk

Trevor Campbell
Department of Statistics
University of British Columbia
Vancouver, BC V6T 1Z4
trevor@stat.ubc.ca

Abstract

Standard Bayesian inference algorithms are prohibitively expensive in the regime of modern large-scale data. Recent work has found that a small, weighted subset of data (a *coreset*) may be used in place of the full dataset during inference, taking advantage of data redundancy to reduce computational cost. However, this approach has limitations in the increasingly common setting of sensitive, high-dimensional data. Indeed, we prove that there are situations in which the Kullback-Leibler (KL) divergence between the *optimal* coreset and the true posterior grows with data dimension; and as coresets include a subset of the original data, they cannot be constructed in a manner that preserves individual privacy. We address both of these issues with a single unified solution, *Bayesian pseudocoresets*—a small weighted collection of synthetic "pseudodata"—along with a variational optimization method to select both pseudodata and weights. The use of pseudodata (as opposed to the original datapoints) enables both the summarization of high-dimensional data and the differentially private summarization of sensitive data. Real and synthetic experiments on high-dimensional data demonstrate that Bayesian pseudocoresets achieve significant improvements in posterior approximation error compared to traditional coresets, and that pseudocoresets provide privacy without a significant loss in approximation quality.

1 Introduction

Large-scale data—which has become the norm in many scientific and commercial applications of statistical machine learning—creates an inherently difficult setting for the modern data analyst. Exploring such data is difficult because it cannot all be obtained and directly visualized at once; one is typically limited to accessing potentially nonrepresentative random subsets of data. Exploring models is similarly hard, as training even a single model can be a computationally expensive, slow, and unreliable process. And as many sources of large-scale data contain sensitive information about individuals (e.g., electronic health records and social network data), these challenges are coupled with growing privacy concerns that preclude direct access to individual datapoints completely.

Large-scale data does offer one reprieve to the analyst: it often exhibits a significant degree of redundancy. Most data are not unique or particularly informative for modelling and exploration. Based on this notion, data summarization methods have been developed that provide the practitioner with a compressed—but still statistically representative—version of the large dataset for analysis. Summarizations have been developed for a variety of purposes, e.g., reducing the cost of computing

Preprint. Under review.

with kernel matrices via Nyström-type approximations [10, 32, 3] or sparse pseudo-input parameterizations [36], Bayesian inference [23, 22, 7, 8, 6], maximum likelihood parameter estimation [12, 30], linear regression [42, 20], geometric shape approximation [2], clustering [16, 29, 4, 5], and dimensionality reduction [18].

A common form of summarization is that of a sparse, weighted subset of the original dataset—a *coreset* [2]. Coresets have two distinct advantages over other possible summarization modalities: they are easily interpreted, and can often be used as the input to standard data analysis algorithms without modification. But as the dimensionality of a dataset grows, its constituent datapoints tend to become more "unique" and cannot represent one another well. Indeed, in the context of Bayesian inference—the focus of the present work—we show that the *optimal* coreset posterior approximation to the true posterior has KL divergence that scales with the dimension of the data in a simple problem setting (Proposition 1). Furthermore, directly releasing a subset of the original data precludes any possibility of individual privacy under the current standard of differential privacy [14, 15]. Past work addresses this issue in the context of clustering and computational geometry [17, 19]—with the remarkable property that the privatized coreset may be queried *ad infinitum* without loss of privacy—but no such method exists for Bayesian posterior inference.

In this work, we develop a novel technique for data summarization in the context of Bayesian inference under the constraints that the method is scalable and easy to use, creates an intuitive summarization, applies to high-dimensional data, and enables privacy control. Inspired by past work [30, 42, 36], instead of using constituent datapoints, we use synthetic *pseudodata* to summarize the large dataset, resulting in a *pseudocoreset*. We show that in the high-dimensional problem setting of Proposition 1, the optimal pseudocoreset with just one pseudodata point recovers the exact posterior, a significant improvement upon the optimal standard coreset of any size. As in past work on Bayesian coresets [6], we formulate pseudocoreset construction as variational inference, and provide a stochastic optimization method. As a consequence of the use of pseudodata—as well as privacy-preserving stochastic gradient descent mechanisms [1, 34, 27]—we show that our method can easily be modified to output a privatized pseudocoreset. The paper concludes with experimental results demonstrating the performance of pseudocoresets on real and synthetic data.

2 Bayesian Coresets

In this work, the goal is to approximate expectations under a density $\pi(\theta)$, $\theta \in \Theta$ expressed as the product of N potentials $(f(x_n, \theta))_{n=1}^N$ and a base density $\pi_0(\theta)$:

$$\pi(\theta) := \frac{1}{Z} \exp\left(\sum_{n=1}^N f(x_n, \theta)\right) \pi_0(\theta).$$

In the setting of Bayesian inference with conditionally independent data, the potentials are data log-likelihoods, i.e. $f(x_n, \theta) := \log p(x_n|\theta)$, π_0 is the prior density, π is the posterior, and Z is the marginal likelihood of the data. Rather than working directly with $\pi(\theta)$ for posterior inference—which requires a $\Theta(N)$ computation per evaluation—a Bayesian coreset approximation of the form

$$\pi_w(\theta) := \frac{1}{Z(w)} \exp\left(\sum_{n=1}^N w_n f(x_n, \theta)\right) \pi_0(\theta)$$

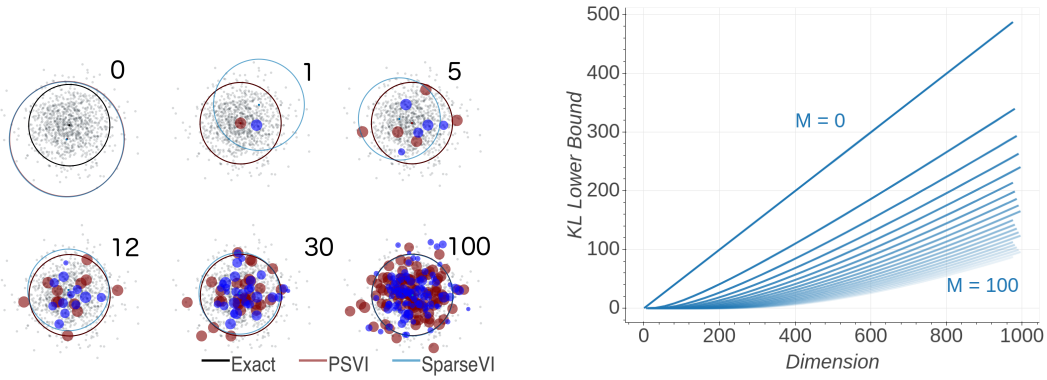
for $w \in \mathbb{R}^N$, $w \geq 0$ may be used in most popular posterior inference schemes [33, 28, 35]. If the number of nonzero entries $\|w\|_0$ of w is small, this results in a significant reduction in computational burden. Recent work has formulated the problem of constructing a Bayesian coreset of size $M \in \mathbb{N}$ as sparse variational inference [6],

$$w^\star = \underset{w \in \mathbb{R}^N}{\arg\min}\, \mathrm{D}_{\mathrm{KL}}\left(\pi_w || \pi_1\right) \qquad \text{s.t.} \quad w \geq 0,\ \|w\|_0 \leq M, \tag{1}$$

and showed that the objective can be minimized using stochastic estimates of $\nabla_w \mathrm{D}_{\mathrm{KL}}\left(\pi_w||\pi_1\right)$ based on samples from the coreset posterior π_w.

2.1 High-dimensional data

Coresets, as formulated in Eq. (1), are limited to using the original datapoints themselves to summarize the whole dataset. Proposition 1 shows that this is problematic when summarizing high-dimensional data; in the common setting of posterior inference for a Gaussian mean, the KL divergence $\mathrm{D}_{\mathrm{KL}}\left(\pi_{w^\star}||\pi_1\right)$ of the *optimal* coreset of any size scales with the dimension of the data. The proof may be found in Supp. A.



(a) (b)

Figure 1: Gaussian mean inference under pseudocoreset (`PSVI`) against standard coreset (`SparseVI`) summarization for $N = 1,000$ datapoints. (a) Progression of `PSVI` vs. `SparseVI` construction for coreset sizes $M = 0, 1, 5, 12, 30, 100$, in 500 dimensions (displayed are datapoint projections on 2 random dimensions). `PSVI` and `SparseVI` coreset predictive 3σ ellipses are displayed in red and blue respectively, while the true posterior 3σ ellipse is shown in black. `PSVI` has the ability to immediately move pseudopoints towards the true posterior mean, while `SparseVI` has to add a larger number of existing points in order to obtain a good posterior approximation. See Fig. 2 for the quantitative KL comparison. (b) Optimal coreset KL divergence lower bound from Proposition 1 as a function of dimension with $\delta = 0.5$, and coreset size M evenly spaced from 0 to 100 in increments of 5.

Proposition 1. *Suppose we use* $(X_n)_{n=1}^N \overset{i.i.d.}{\sim} \mathcal{N}(0, I)$ *in* $\mathbb{R}^d$ *to perform posterior inference in a Bayesian model with prior* $\mu \sim \mathcal{N}(0, I)$ *and likelihood* $(X_n)_{n=1}^N \overset{i.i.d.}{\sim} \mathcal{N}(\mu, I)$. *Then* $\forall M < d$ *and* $\delta \in [0, 1]$, *with probability at least* $1 - \delta$ *the optimal size-*M *coreset* $w^\star$ *satisfies*

$$\mathrm{D}_{\mathrm{KL}}\left(\pi_{w^\star} || \pi_1\right) \geq \frac{1}{2} \frac{N - M}{1 + N} F_{d-M}^{-1}\left(\delta \binom{N}{M}^{-1}\right),$$

where F_k *is the CDF of a* χ^2 *random variable with* k *degrees of freedom.*

The bound in Proposition 1 depends on d through the χ^2 distribution inverse CDF. Although difficult to see directly, the bound is reasonably large for typical values of N, M, d, δ, and increasing linearly in d; Fig. 1b visualizes the value of the lower bound as a function of dimension d for various coreset sizes M. Note that the above bound requires the data to be high-dimensional such that $d > M$; if $d \leq M$ the proof technique in Supp. A results in a vacuous $\mathrm{D}_{\mathrm{KL}}\left(\pi_{w^\star} || \pi_1\right) = 0$ lower bound.

3 Bayesian Pseudocoresets

Proposition 1 shows that there is room for improvement in coreset construction in the high-dimensional data regime. Indeed, consider again the same problem setting; the coreset posterior distribution is a Gaussian with mean μ_w and covariance Σ_w,

$$\Sigma_w = \left(1 + \sum_{n=1}^N w_n\right)^{-1} I \qquad \mu_w = \Sigma_w \sum_{n=1}^N w_n X_n. \tag{2}$$

Examining Eq. (2), we can replicate any coreset posterior exactly by using a single synthetic *pseudodata* point $U = \left(\sum_{n=1}^N w_n\right)^{-1} \sum_{n=1}^N w_n X_n$ with weight $\sum_{n=1}^N w_n$. In particular, the true posterior is equivalent to the posterior conditioned on the single pseudodata point $U = \frac{1}{N} \sum_{n=1}^N X_n$ with weight N (with corresponding KL divergence equal to 0). This is not surprising; the mean of the data is precisely a sufficient statistic for the data in this simple setting. However, it does illustrate that carefully-chosen pseudodata may be able to represent the overall dataset—as "approximate sufficient statistics"—far better than any reasonably small collection of the original data. This intuition has been used before, e.g., for scalable Gaussian process inference [36, 37], privacy-preserving compression in linear regression [42], herding [41, 9, 25], and deep generative models [38].

In this section, we extend the realm of applicability of pseudopoint compression methods to the general class of Bayesian posterior inference problems with conditionally independent data, resulting in *Bayesian pseudocoresets*. Building on recent work [6], we formulate pseudocoreset construction as a variational inference problem where both the weights and pseudopoint locations are parameters of the variational posterior approximation, and develop a stochastic algorithm to solve the optimization.

3.1 Pseudocoreset variational inference

A Bayesian pseudocoreset takes the form

$$\pi_{u,w}(\theta) = \frac{1}{Z(u,w)} \exp\left(\sum_{m=1}^{M} w_m f(u_m, \theta)\right) \pi_0(\theta),$$

where $u := (u_m)_{m=1}^M$ are M pseudodata points $u_m \in \mathbb{R}^d$, $(w_m)_{m=1}^M$ are nonnegative weights, $f : \mathbb{R}^d \times \Theta \to \mathbb{R}$ is a potential function parametrized by a pseudodata point, and $Z(u, w)$ is the corresponding normalization constant rendering $\pi_{u,w}$ a probability density. In the setting of Bayesian posterior inference, u_m will take the same form as the data, while the potentials are the log-likelihood functions, i.e., $f(u_m, \theta) = \log p(u_m|\theta)$. We construct a coreset by minimizing the KL divergence over both the pseudodata locations and weights,

$$u^\star, w^\star = \underset{u \in \mathbb{R}^{d\times M}, w \in \mathbb{R}_+^M}{\arg\min} \mathrm{D}_{\mathrm{KL}}\left(\pi_{u,w} || \pi\right). \tag{3}$$

As opposed to previous Bayesian coreset construction optimization problems [7, 8, 6], we do not need an explicit sparsity constraint; the coreset size is limited to M directly through the selection of the number of pseudodata and weights.

Denote the vectors of original data potentials $f(\theta) \in \mathbb{R}^N$ and synthetic pseudodata potentials $\tilde{f}(\theta) \in \mathbb{R}^M$ as $f(\theta) := [f_1(\theta) \dots f_N(\theta)]^T$ and $\tilde{f}(\theta) := [f(u_1, \theta) \dots f(u_M, \theta)]^T$ respectively, where we suppress the (θ) for brevity where clear from context. Denote $\mathbb{E}_{u,w}$ and $\mathrm{Cov}_{u,w}$ to be the expectation and covariance operator for the pseudocoreset posterior $\pi_{u,w}$. Then we may write the KL divergence in Eq. (3) as

$$\begin{aligned}\mathrm{D}_{\mathrm{KL}}\left(\pi_{u,w} || \pi\right) =& \mathbb{E}_{u,w}[\log \pi_{u,w}(\theta)] - \mathbb{E}_{u,w}[\log \pi(\theta)] \\ =& \log Z(1) - \log Z(u,w) - 1^T \mathbb{E}_{u,w}[f] + w^T \mathbb{E}_{u,w}[\tilde{f}],\end{aligned} \tag{4}$$

where $1 \in \mathbb{R}^N$ is the vector of all 1 entries, and $w \in \mathbb{R}^M$ is the vector of pseudocoreset weights.

As we will employ gradient descent steps as part of our algorithm to minimize the variational objective over the parameters u, w, we need to evaluate the derivative of the KL divergence Eq. (4). Despite the presence of the intractable normalization constants and expectations, we show in Supp. B that gradients can be expressed using moments of the pseudodata and original data potential vectors. In particular, the gradients of the KL divergence with respect to the weights w and to a single pseudodata location u_m are

$$\nabla_w \mathrm{D}_{\mathrm{KL}} = -\mathrm{Cov}_{u,w}[\tilde{f}, f^T 1 - \tilde{f}^T w], \quad \nabla_{u_m} \mathrm{D}_{\mathrm{KL}} = -w_m \mathrm{Cov}_{u,w}\left[h(u_m), f^T 1 - \tilde{f}^T w\right], \tag{5}$$

where $h(\cdot, \theta) := \nabla_u f(\cdot, \theta)$, and the θ argument is again suppressed for brevity.

3.2 Stochastic optimization

The gradients in Eq. (5) involve expectations of (gradient) log-likelihoods from the model. Although there are a few particular Bayesian models where these can be evaluated in closed-form (e.g. the synthetic experiment in Section 4; see also Supp. C.1), this is not usually the case. In order to make the proposed pseudocoreset method broadly applicable, in this section we develop a black-box stochastic optimization scheme (Alg. 1) for Eq. (3).

To initialize the pseudocoreset, we subsample M datapoints from the large dataset and reweight them to match the overall weight of the full dataset,

$$\begin{aligned}&u_m \leftarrow x_{b_m}, \quad w_m \leftarrow N/M, \quad m = 1, \dots, M \\ &\mathcal{B} \sim \mathsf{UnifSubset}\left([N], M\right), \quad \mathcal{B} := \{b_1, \dots, b_M\}.\end{aligned}$$

After initializing the pseudodata locations and weights, we simultaneously optimize Eq. (3) over both. Each optimization iteration $t \in \{1, \dots, T\}$ consists of a stochastic gradient descent step with a learning rate $\gamma_t \propto t^{-1}$,

$$w_m \leftarrow \max\left(0, w_m - \gamma_t (\hat{\nabla}_w)_m\right), \quad u_m \leftarrow u_m - \gamma_t \hat{\nabla}_{u_m}, \quad 1 \le m \le M.$$

Algorithm 1 Pseudocoreset Variational Inference

procedure PSVI($f(\cdot,\cdot), \pi_0, x, M, B, S, T, (\gamma_t)_{t=1}^{\infty}$)
  ▷ Initialize the pseudocoreset using a uniformly chosen subset of the full dataset
  $N \leftarrow$ # datapoints in x, $\quad \mathcal{B} \sim \mathsf{UnifSubset}([N], M)$, $\quad \mathcal{B} := \{b_1, \ldots, b_M\}$
  $u_m \leftarrow x_{b_m}, \quad w_m \leftarrow {}^{N}\!/_{M}, \quad m = 1, \ldots, M$
  for $t = 1, \ldots, T$ **do**
    ▷ Take S samples from current pseudocoreset posterior
    $(\theta)_{s=1}^{S} \overset{\text{i.i.d.}}{\sim} \pi_{u,w}$ where $\pi_{u,w}(\theta) \propto \exp\left(\sum_{m=1}^{M} w_m f(u_m, \theta)\right) \pi_0(\theta)$
    $\mathcal{B} \sim \mathsf{UnifSubset}([N], B)$ ▷ Obtain a minibatch of B datapoints from the full dataset
    for $s = 1, \ldots, S$ **do** ▷ Compute (gradient) log-likelihood discretizations
      $g_s \leftarrow \left(f(x_b, \theta_s) - {}^{1}\!/_{S} \sum_{s'=1}^{S} f(x_b, \theta_{s'})\right)_{b \in \mathcal{B}} \in \mathbb{R}^B$
      $\tilde{g}_s \leftarrow \left(f(u_m, \theta_s) - {}^{1}\!/_{S} \sum_{s'=1}^{S} f(u_m, \theta_{s'})\right)_{m=1}^{M} \in \mathbb{R}^M$
      for $m = 1, \ldots, M$ **do**
        $\tilde{h}_{m,s} \leftarrow \nabla_u f(u_m, \theta_s) - {}^{1}\!/_{S} \sum_{s'=1}^{S} \nabla_u f(u_m, \theta_{s'}) \in \mathbb{R}^d$
    $\hat{\nabla}_w \leftarrow -{}^{1}\!/_{S} \sum_{s=1}^{S} \tilde{g}_s \left({}^{N}\!/_{B} g_s^T 1 - \tilde{g}_s^T w\right)$ ▷ Compute Monte-Carlo gradients for w
    for $m = 1, \ldots, M$ **do** and $(u_m)_{m=1}^{M}$
      $\hat{\nabla}_{u_m} \leftarrow -w_m {}^{1}\!/_{S} \sum_{s=1}^{S} \tilde{h}_{m,s} \left({}^{N}\!/_{B} g_s^T 1 - \tilde{g}_s^T w\right)$
    $w \leftarrow \max(w - \gamma_t \hat{\nabla}_w, 0)$ ▷ Take stochastic gradient step in w
    for $m = 1, \ldots, M$ **do** and $(u_m)_{m=1}^{M}$
      $u_m \leftarrow u_m - \gamma_t \hat{\nabla}_{u_m}$
  return w, $(u_m)_{m=1}^{M}$

The stochastic gradient estimates $\hat{\nabla}_w \in \mathbb{R}^M$ and $\hat{\nabla}_{u_m} \in \mathbb{R}^d$ are based on $S \in \mathbb{N}$ samples $\theta_s \overset{\text{i.i.d.}}{\sim} \pi_{u,w}$ from the coreset approximation and a minibatch of $B \in \mathbb{N}$ datapoints from the full dataset,

$$\hat{\nabla}_w := -\frac{1}{S} \sum_{s=1}^{S} \tilde{g}_s \left(\frac{N}{B} g_s^T 1 - \tilde{g}_s^T w\right), \quad \hat{\nabla}_{u_m} := -w_m \frac{1}{S} \sum_{s=1}^{S} \tilde{h}_{m,s} \left(\frac{N}{B} g_s^T 1 - \tilde{g}_s^T w\right),$$

where

$$\tilde{h}_{m,s} := \nabla_u f(u_m, \theta_s) - \frac{1}{S} \sum_{s'=1}^{S} \nabla_u f(u_m, \theta_{s'}), \qquad g_s := \left(f(\theta_s) - \frac{1}{S} \sum_{s'=1}^{S} f(\theta_{s'})\right)\Bigg|_{\mathcal{B}}$$

$$\tilde{g}_s := \tilde{f}(\theta_s) - \frac{1}{S} \sum_{s'=1}^{S} \tilde{f}(\theta_{s'}), \qquad \mathcal{B} \sim \mathsf{UnifSubset}([N], B),$$

and $(\cdot)|_{\mathcal{B}}$ denotes restriction of a vector to only those indices in $\mathcal{B} \subset [N]$. Crucially, note that this computation does not scale with N, but rather with the number of coreset points M, the sample and minibatch sizes S and B, and the dimension d. Obtaining $\theta_s \overset{\text{i.i.d.}}{\sim} \pi_{u,w}$ efficiently via Markov chain Monte Carlo sampling algorithms [21, 26] is (roughly) $O(M)$ per sample, because the coreset is always of size M; and we need not compute the entire vector $g_s \in \mathbb{R}^N$ per sample s, but rather only those $B \ll N$ indices in the minibatch $\mathcal{B}$, resulting in a cost of $O(B)$. Aside from that, all computations involving $\tilde{g}_s \in \mathbb{R}^M$ and $\tilde{h}_{m,s} \in \mathbb{R}^d$ are at most $O(Md)$. Each of these computations are repeated S times over the coreset posterior samples.

3.3 Differentially Private Scheme

Beyond better summarizations of high-dimensional data, pseudocoresets enable the generation of a data summarization that ensures the statistical privacy of individual datapoints under the model of (approximate) *differential privacy*. In this setting, a trusted curator holds an aggregate dataset of N datapoints, $x \in \mathcal{X}^N$, $\mathcal{X} \subseteq \mathbb{R}^d$, and builds and releases a pseudocoreset (u, w), $u \in \mathcal{X}^M$, $w \in \mathbb{R}_+^M$ via a randomized mechanism satisfying Definition 2 [14, 13].

Definition 2 ((ε, δ)-Differentially Private Coreset)**.** Fix $\varepsilon \geq 0, \delta \in [0, 1]$. A pseudocoreset construction algorithm $\mathcal{M} : \mathcal{X}^N \to \mathbb{R}_+^M \times \mathcal{X}^M$ is (ε, δ)-differentially private if for every pair of adjacent datasets $x \approx x'$ and all events $A \subseteq \mathbb{R}_+^M \times \mathcal{X}^M$, $\mathbb{P}[\mathcal{M}(x) \in A] \leq e^{\varepsilon} \mathbb{P}[\mathcal{M}(x') \in A] + \delta$.

We consider two datasets x, x' as adjacent (denoted $x \approx x'$) if x' can be obtained from x by adding or removing an element. ε controls the effect that removal or addition of an element can have on the output distribution of $\mathcal{M}$, while δ captures the failure probability, and is preferably $o(1/N)$.

In this section, we develop a differentially private version of pseudocoreset construction. Beyond modifying our initialization scheme, private pseudocoreset construction comes as natural extension of Alg. 1 via replacing gradient computation involving points of the true dataset with its differentially private counterpart.

Pseudodata points initialization In the standard (nonprivate) pseudocoreset construction (Alg. 1), pseudopoints are initialized from the dataset itself, incurring a privacy penalty. In differentially private pseudocoreset construction, we simply initialize pseudopoints by generating synthetic data from the statistical model at no privacy cost.

Optimization Examining lines 4–19 of Alg. 1, the only steps that involve handling the original data occur at lines 8, 12, and 14, when we use the minibatch subsample to compute log-likelihoods and gradients. Due to the post-processing property of differential privacy [15], all of the other computations in Alg. 1 (e.g. sampling from the pseudocoreset posterior, computing pseudopoint log-likelihoods, etc.) incur no privacy cost. Therefore, we need only to control the influence of private data entering the gradient computation through the vector of $(g_s^T 1)_{s=1}^S$ terms.

To accomplish this we do repeated applications of the *subsampled Gaussian mechanism*, since this also allows us to use a *moments accountant* technique to keep tight estimates of privacy parameters [1, 40]. As in the nonprivate scheme, in each optimization step we uniformly subsample a minibatch $\mathcal{B} = \{x_1, \dots, x_B\}$ of private datapoints. We then replace the $g_s^T 1$ term in lines 12 and 14 with a randomized privatization:

$$\text{replace} \quad (g_s^T 1)_{s=1}^S \quad \text{with} \quad Z + \sum_{i=1}^{B} \frac{G_i}{\max\left(1, \frac{||G_i||_2}{C}\right)}, \qquad Z \sim \mathcal{N}(0, \sigma^2 C^2 I), \tag{6}$$

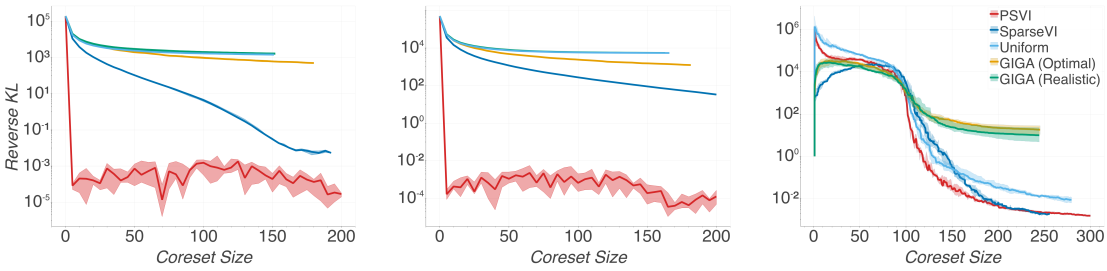
where $G_i := \left(f(x_i, \theta_s) - \frac{1}{S}\sum_{s'=1}^S f(x_i, \theta_{s'})\right)_{s=1}^S \in \mathbb{R}^S \ \forall x_i \in \mathcal{B}$, and $C, \sigma > 0$ are parameters controlling the amount of privacy. This modification to Alg. 1 has been shown in past work to obtain the privacy guarantee provided in Corollary 3; crucially, the privacy cost of our construction is independent of the pseudocoreset size. It also does not introduce any significant amount of additional computation. No sensitivity computation for privatisation noise calibration is required, as boundedness is enforced via clipping in Eq. (6). Finally, a manageable number of privacy specific hyperparameters is introduced: the clipping bound C and noise level σ.

Corollary 3 ([1]). *There exist constants c_1, c_2 such that Alg. 1 modified per Eq. (6) is (ε, δ)-differentially private for any $\varepsilon < c_1 q^2 T$, $\delta > 0$, and $\sigma \geq c_2 q \sqrt{T \log(1/\delta)}/\varepsilon$, where $q := \frac{B}{N}$ is the fraction of data in a minibatch and T is the number of optimization steps.*

4 Experimental Results

In this section, we evaluate the posterior approximation quality achieved by pseudocoreset sparse VI (`PSVI`) compared against uniform random subsampling (`Uniform`), Hilbert coresets (`GIGA` [7]) and `SparseVI` greedy coreset construction [6]. For black-box constructions of `SparseVI` and `PSVI` we used $S = 100$ Monte Carlo samples per gradient estimation. For `GIGA` we used a 100-dimensional random projection from a Gaussian approximate posterior $\hat{\pi}$ with two choices for mean and covariance: one set to the exact posterior (`Optimal`), which is not tractable to obtain in practice and forms an optimistic estimate of achievable approximation quality; and one with mean and covariance set to a random point on the interpolant between the prior and the exact posterior point estimates, and subsequently corrupted with 75% additive relative noise (`Realistic`). Notably, Hilbert coresets and `SparseVI` develop incremental schemes for construction, while `PSVI` relies on batch optimization with random initialization (Alg. 1), and does not use any information from pseudocoresets of smaller size. An incremental scheme for `SparseVI` is included in Supp. C. Code for the presented experiments is available at anonymous_public_repo.

Gaussian mean inference We first evaluate the performance of `PSVI` on a synthetic dataset of $N = 10^3$ datapoints, where we aim to infer the posterior mean $\theta \sim \mathcal{N}(\mu_0, \Sigma_0)$ of a d-dimensional Gaussian conditioned on Gaussian observations $(X_n)_{n=1}^N \overset{\text{i.i.d.}}{\sim} \mathcal{N}(\theta, \Sigma)$. In this example, the exact



(a) Gauss. mean inf., $d = 200$ (b) Gauss. mean inf., $d = 500$ (c) Bayes. lin. regr., $d = 100$

Figure 2: Comparison of coreset approximate posterior quality for experiments on synthetic datasets over 10 trials. Solid lines display the median KL divergence, with shaded areas showing 25^{th} and 75^{th} percentiles of KL divergence. In Fig. 2c, KL divergence is normalized by the prior.

pseudocoreset posterior for any set of weights $(w_m)_{m=1}^M$ and pseudopoint locations $(u_m)_{m=1}^M$ is available in closed-form:

$$\Sigma_{u,w} = (\Sigma_0^{-1} + \sum_{m=1}^{M} w_m \Sigma^{-1})^{-1} \qquad \mu_{u,w} = \Sigma_{u,w}(\Sigma_0^{-1}\mu_0 + \Sigma^{-1}\sum_{m=1}^{M} w_m u_m).$$

Using the exact posterior, we derive the exact moments used in the gradient formulae from Eq. (5) in closed form (see Supp. C.1),

$$\mathrm{Cov}_{u,w}[f_n, f_m] = v_n^T \Psi v_m + {}^1\!/_2 \operatorname{tr} \Psi^T \Psi, \quad \mathrm{Cov}_{u,w}[\tilde{f}_n, f_m] = \tilde{v}_n^T \Psi v_m + {}^1\!/_2 \operatorname{tr} \Psi^T \Psi,$$
$$\mathrm{Cov}_{u,w}[h(u_i), f_n] = Q^{-T} \Psi v_n, \qquad \mathrm{Cov}_{u,w}[h(u_i), \tilde{f}_n] = Q^{-T} \Psi \tilde{v}_n,$$

where Q is the lower triangular matrix of the Cholesky decomposition of Σ (i.e. $\Sigma = QQ^T$), $\Psi := Q^{-1}\Sigma_{u,w}Q^{-T}$, $v_n := Q^{-1}(x_n - \mu_{u,w})$, and $\tilde{v}_m := Q^{-1}(u_m - \mu_{u,w})$. We vary the pseudocoreset size from $M = 1$ to 200, and set the total number of iterations to $T = 500$. We use learning rates $\gamma_t(M) = \alpha(M)t^{-1}$, where $\alpha(M) = 1$ for `SparseVI` and $\alpha(M) = \max(1.1 - 0.005M, 0.2)$ for `PSVI`. As verified in Figs. 2a and 2b, Hilbert coresets provide poor quality summarizations in the high-dimensional regime, even for large coreset sizes. Despite showing faster decrease of approximation error for a larger range of coreset sizes, `SparseVI` is also fundamentally limited by the use of the original datapoints, per Proposition 1. Furthermore, we observe that the quality of all previous coreset methods when $d = 500$ is significantly lower compared to $d = 200$. On the other hand, the KL divergence for `PSVI` decreases significantly more quickly, giving a near perfect approximation for true posterior with a single pseudodata point regardless of data dimension. As shown earlier in Fig. 1a, `PSVI` has the capacity to move the pseudodata points in order to capture the true posterior very efficiently.

Bayesian linear regression In the second experiment, we use a set of $N = 2,000$ 101-dimensional datapoints $(x_n, y_n)_{n=1}^N$ generated as follows: $(x_n)_{n=1}^N \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, I)$, $(y_n)_{n=1}^N \sim [1, x_n]^T\theta + \epsilon_n$, $(\epsilon_n)_{n=1}^N \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, \sigma^2)$, and aim to infer $\theta \in \mathbb{R}^{101}$. We assume a prior $\theta \sim \mathcal{N}(\mu_0, \sigma_0^2 I)$, where μ_0, σ_0^2 are the dataset empirical mean and second moment, and set the noise parameter σ to the variance of $(y_n)_{n=1}^N$. We apply stochastic optimization for `PSVI` construction (also see Supp. C.2.1). We use learning rates $\gamma_t = t^{-1}$ for `SparseVI`, and $\gamma_t = 0.1t^{-1}$ for `PSVI`, $B = 200$, $T = 1000$, while selection step for `SparseVI` is carried out over the full dataset. Fig. 2c shows that Hilbert coresets cannot improve posterior approximation in this setting with 100 random projections (see Supp. C.2.2), while `PSVI` achieves the fastest decay rate over sizes $100 < M < 250$, surpassing `SparseVI`.

Bayesian logistic regression Finally, we compare (pseudo)coreset construction methods on Bayesian logistic regression applied to 3 large (8.4–$100K$ datapoints, 50–237 dimensions) datasets. For brevity, equations and gradients for the logistic regression model, additional experiments on 3 smaller-scale datasets, full dataset descriptions, hyperparameter selection, time performance evaluation and results on an incremental scheme for pseudocoreset construction are deferred to Supp. C.3. For `PSVI` and `SparseVI` we use minibatch size $B = 200$, number of gradient updates $T = 500$, and learning rate schedules $\gamma_t = \alpha t^{-1}$. For TRANSACTIONS, CHEMREACT100 and MUSIC, α is respectively set to 0.1, 0.1, 1 for `SparseVI`, and 1, 10, 10 for `PSVI`. In the selection step of `SparseVI` we use a uniform subsample of $1,000$ datapoints. For the differentially private pseudocoreset constructions (`DP-PSVI`), we use a subsampling ratio $q = 2 \times 10^{-3}$. At each iteration we adapt the clipping norm

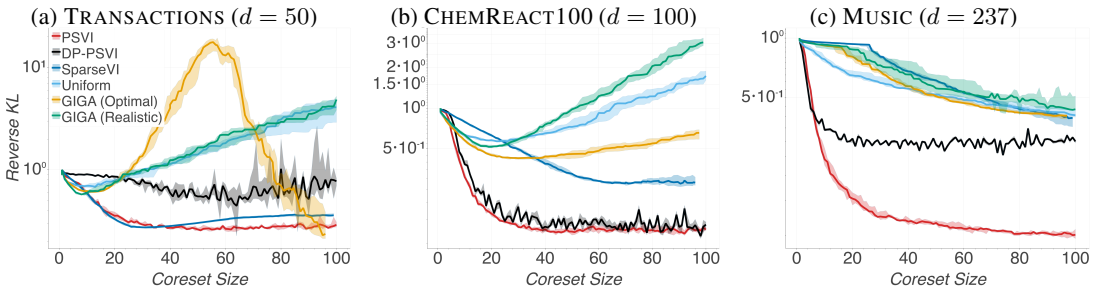


Figure 3: Comparison of (pseudo)coreset approximate posterior quality vs coreset size for logistic regression over 10 trials on 3 large-scale datasets. Presented differentially private pseudocoresets correspond to $(0.2, 1/N)$-DP. Reverse KL divergence is displayed normalized by the prior.

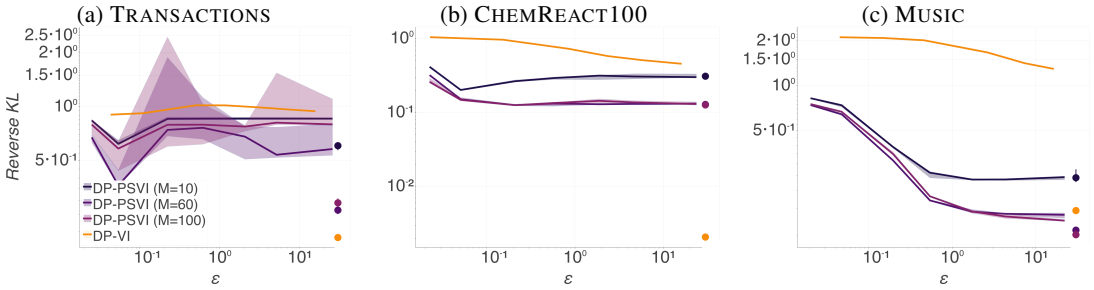


Figure 4: Approximate posterior quality over decreasing differential privacy guarantees for private pseudocoresets of varying size plotted against private variational inference [27]. δ is always kept fixed at $1/N$. Markers on the right end of each plot display the errorbar of approximation achieved by the corresponding nonprivate posteriors. Results are displayed over 5 trials for each construction.

value C to the median norm of $(f(u_m, \theta_s) - \frac{1}{S}\sum_{s'=1}^{S} f(u_m, \theta_{s'}))_{s=1}^{S}$ computed over pseudodata point values u_m, and use noise level $\sigma = 5$. We initialise each pseudocoreset of size M via sampling $(x_m)_{m=1}^{M} \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, I)$, and sampling θ, $(y_m)_{m=1}^{M}$ from the statistical model.

Results presented in Fig. 3 demonstrate that `PSVI` achieves consistently the smallest posterior approximation error in the small coreset size regime, offering improvement compared to `SparseVI` and being competitive with `GIGA (Optimal)`, without the requirement for specifying a weighting function. In Fig. 3a, for $M \geq d$ `GIGA (Optimal)` follows a much steeper decrease in KL divergence, reflecting the dependence of its approximation quality on dataset dimension per Proposition 1. In contrast, `PSVI` typically reaches its minimum at $M < d$. The difference in approximation quality becomes clearer in higher dimensions (e.g. MUSIC, where $d = 237$). Perhaps surprisingly, the private pseudocoreset construction has only marginally worse approximation quality compared to nonprivate `PSVI` and generally achieves better peformance in comparison to the other state-of-the-art nonprivate coreset constructions. In Fig. 4 we present achieved posterior approximation quality via `DP-PSVI`, against a competitive state-of-the-art method (`DP-VI`, [27]). For logistic regression, `DP-VI` infers an approximate posterior from the family of Gaussians with diagonal covariance via ADVI [28], followed by an additional Laplace approximation. Note that by design, `DP-VI` is constrained by the usual Gaussian variational approximation, while `DP-PSVI` is more flexible and can approach the true posterior as M increases—this effect is reflected in nonprivate posteriors as well as data dimensionality grows (see for example Fig. 4c). Indeed, we verify that in the high-privacy regime `DP-PSVI` for sufficient pseudocoreset size (which is typically small for tested real-world datasets) offers posterior approximation with better KL divergence compared to `DP-VI`. Our findings indicate that private `PSVI` offers efficient releases of big data via informative pseudopoints, which enable arbitrary post processing (e.g. running any *nonprivate* black-box algorithm for Bayesian inference), under strong privacy guarantees and without reducing the quality of inference.

5 Conclusion

We introduced a new variational formulation for Bayesian coreset construction, which yields efficient summarizations for big and high-dimensional datasets via simultaneously learning pseudodata points

locations and weights. We proved limitations of existing variational formulations for coresets and demonstrated that they can be resolved with our new methodology. We proposed an efficient construction scheme via black-box stochastic optimization and showed how it can be adapted for differentially private Bayesian summarization. Finally, we demonstrated the applicability of our methodology on synthetic and real-world datasets, and practical statistical models.

Broader Impact

Pseudocoreset variational inference is a general-purpose Bayesian inference algorithm, hence shares implications mostly encountered in approximate inference methods. For example, replacing the full dataset with a pseudocoreset has the potential to cause inferential errors; these can be partially tempered by using a pseudocoreset of larger size. Note also that the optimization algorithm in this work aims to reduce KL divergence: however the proposed variational objective might be misleading in many applications and lead to incorrect conclusions in certain statistical models (e.g. point estimates and uncertainties might be far off despite KL being almost zero [24]). Moreover, Bayesian inference in general is prone to model misspecification. Therefore, a pseudocoreset summarization based on a wrong statistical model will lead to non-representative compression for inferential purposes. Constructing the coreset on a statistical model suited for robust inference instead of the original one [31, 39], can offer protection against modelling mismatches. Naturally, the utility of generated dataset summary becomes task-dependent, as it has been optimized for a specific learning objective, and cannot be fully transferable to multiple different inference tasks on the same dataset.

Our learnable pseudodata are also generally not as interpretable as the points of previous coreset methods, as they are not real data. And the level of interpretability is model specific. This creates a risk of misinterpretation of pseudocoreset points in practice. On the other hand, our optimization framework does allow the introduction of interpretability constraints (e.g. pseudodata sparsity) to explicitly capture interpretability requirements.

Pseudocoreset-based summarization is susceptible to reproducing potential biases and unfairness existing in the original dataset. Majority-group datapoints in the full dataset which capture information relevant to the statistical task of interest are expected to remain over-represented in the learned summary; while minority-group datapoints might be eliminated, if their distinguishing features are not related to inference. Amending the initialization step to contain such datapoints or using a prior that strongly favors a debiased version of the dataset could both mitigate these concerns; but more study is warranted.

6 Acknowledgements

DM and CM have been partially supported by Nokia Bell Labs through their donation for the Centre of Mobile, Wearable Systems and Augmented Intelligence to the University of Cambridge. DM also gratefully acknowledges the financial support from Lundgren Fund and Darwin College Cambridge. DM and CM thank Rik Sarkar for useful conversations in the early stages of this work.

References

[1] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang. Deep learning with differential privacy. In *ACM SIGSAC Conference on Computer and Communications Security*, 2016.

[2] P. Agarwal, S. Har-Peled, and K. Varadarajan. Geometric approximation via coresets. *Combinatorial and computational geometry*, 52, 2005.

[3] R. Agrawal, T. Campbell, J. Huggins, and T. Broderick. Data-dependent compression of random features for large-scale kernel approximation. In *International Conference on Artificial Intelligence and Statistics*, 2019.

[4] O. Bachem, M. Lucic, and A. Krause. Coresets for nonparametric estimation—the case of DP-means. In *International Conference on Machine Learning*, 2015.

[5] V. Braverman, D. Feldman, and H. Lang. New frameworks for offline and streaming coreset constructions. *arXiv:1612.00889*, 2016.

[6] T. Campbell and B. Beronov. Sparse variational inference: Bayesian coresets from scratch. In *Advances in Neural Information Processing Systems*, 2019.

[7] T. Campbell and T. Broderick. Bayesian coreset construction via greedy iterative geodesic ascent. In *International Conference on Machine Learning*, 2018.

[8] T. Campbell and T. Broderick. Automated scalable Bayesian inference via Hilbert coresets. *Journal of Machine Learning Research*, 20(15), 2019.

[9] Y. Chen, M. Welling, and A. Smola. Super-samples from kernel herding. In *Proceedings of the Twenty-Sixth Conference on Uncertainty in Artificial Intelligence*, 2010.

[10] P. Drineas and M. Mahoney. On the Nyström method for approximating a Gram matrix for improved kernel-based learning. *Journal of Machine Learning Research*, 6, 2005.

[11] J. Duchi, E. Hazan, and Y. Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(61):2121–2159, 2011.

[12] W. DuMouchel, C. Volinsky, T. Johnson, C. Cortes, and D. Pregibon. Squashing flat files flatter. In *ACM Conference on Knowledge Discovery and Data Mining*, 1999.

[13] C. Dwork, K. Kenthapadi, F. McSherry, I. Mironov, and M. Naor. Our data, ourselves: Privacy via distributed noise generation. In *International Conference on The Theory and Applications of Cryptographic Techniques*, 2006.

[14] C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. In *Conference on Theory of Cryptography*, 2006.

[15] C. Dwork and A. Roth. The algorithmic foundations of differential privacy. *Foundations and Trends in Theoretical Computer Science*, 9(3–4), 2014.

[16] D. Feldman, M. Faulkner, and A. Krause. Scalable training of mixture models via coresets. In *Advances in Neural Information Processing Systems*, 2011.

[17] D. Feldman, A. Fiat, H. Kaplan, and K. Nissim. Private coresets. In *ACM Symposium on Theory of Computing*, 2009.

[18] D. Feldman, M. Volkov, and D. Rus. Dimensionality reduction of massive sparse datasets using coresets. In *Advances in Neural Information Processing Systems*, 2016.

[19] D. Feldman, C. Xiang, R. Zhu, and D. Rus. Coresets for differentially private k-means clustering and applications to privacy in mobile sensor networks. In *International Conference on Information Processing in Sensor Networks*, 2017.

[20] R. Guhaniyogi and D. Dunson. Bayesian compressed regression. *Journal of the American Statistical Association*, 110(512), 2015.

[21] M. D. Hoffman and A. Gelman. The No-U-Turn sampler: Adaptively setting path lengths in Hamiltonian Monte Carlo. *Journal of Machine Learning Research*, 15(1):1593–1623, 2014.

[22] J. Huggins, R. Adams, and T. Broderick. PASS-GLM: polynomial approximate sufficient statistics for scalable Bayesian GLM inference. In *Advances in Neural Information Processing Systems*, 2017.

[23] J. Huggins, T. Campbell, and T. Broderick. Coresets for scalable Bayesian logistic regression. In *Advances in Neural Information Processing Systems*, 2016.

[24] J. Huggins, T. Campbell, M. Kasprzak, and T. Broderick. Validated variational inference via practical posterior error bounds. In *International Conference on Artificial Intelligence and Statistics*, 2020.

[25] F. Huszár and D. Duvenaud. Optimally-weighted herding is Bayesian quadrature. In *Proceedings of the Twenty-Eighth Conference on Uncertainty in Artificial Intelligence*, 2012.

[26] P. Jacob, J. O'Leary, and Y. Atchadé. Unbiased Markov chain Monte Carlo with couplings. *arXiv:1708.03625*, 2017.

[27] J. Jälkö, O. Dikmen, and A. Honkela. Differentially private variational inference for non-conjugate models. In *Uncertainty in Artificial Intelligence*, 2017.

[28] A. Kucukelbir, D. Tran, R. Ranganath, A. Gelman, and D. M. Blei. Automatic differentiation variational inference. *Journal of Machine Learning Research*, 18(14), 2017.

[29] M. Lucic, O. Bachem, and A. Krause. Strong coresets for hard and soft Bregman clustering with applications to exponential family mixtures. In *International Conference on Artificial Intelligence and Statistics*, 2016.

[30] D. Madigan, N. Raghavan, and W. DuMouchel. Likelihood-based data squashing: a modeling approach to instance construction. *Data Mining and Knowledge Discovery*, 6, 2002.

[31] J. W. Miller and D. B. Dunson. Robust Bayesian inference via coarsening. *Journal of the American Statistical Association*, 114(527), 2019.

[32] C. Musco and C. Musco. Recursive sampling for the Nyström method. In *Advances in Neural Information Processing Systems*, 2017.

[33] R. Neal. MCMC using Hamiltonian dynamics. In S. Brooks, A. Gelman, G. Jones, and X.-L. Meng, editors, *Handbook of Markov chain Monte Carlo*, chapter 5. CRC Press, 2011.

[34] M. Park, J. R. Foulds, K. Chaudhuri, and M. Welling. Variational Bayes in private settings (VIPS). *J. Artif. Intell. Res.*, 68, 2020.

[35] R. Ranganath, S. Gerrish, and D. Blei. Black Box Variational Inference. In *International Conference on Artificial Intelligence and Statistics*, 2014.

[36] E. Snelson and Z. Ghahramani. Sparse Gaussian processes using pseudo-inputs. In *Advances in Neural Information Processing Systems*, 2005.

[37] M. Titsias. Variational learning of inducing variables in sparse Gaussian processes. In *International Conference on Artificial Intelligence and Statistics*, 2009.

[38] J. M. Tomczak and M. Welling. VAE with a VampPrior. In *International Conference on Artificial Intelligence and Statistics*, 2018.

[39] Y. Wang, A. Kucukelbir, and D. M. Blei. Robust probabilistic modeling with Bayesian data reweighting. In *International Conference on Machine Learning*, 2017.

[40] Y.-X. Wang, B. Balle, and S. P. Kasiviswanathan. Subsampled Rényi differential privacy and analytical moments accountant. In *International Conference on Artificial Intelligence and Statistics*, 2019.

[41] M. Welling. Herding dynamical weights to learn. In *Proceedings of the 26th Annual International Conference on Machine Learning*, 2009.

[42] S. Zhou, J. Lafferty, and L. Wasserman. Compressed regression. In *Advances in Neural Information Processing Systems*, 2007.

A Technical Results and Proofs

A.1 Proof of Proposition 1

In the setting of Proposition 1, both the exact posterior and the coreset posterior are multivariate Gaussian distributions, denoted as $\mathcal{N}(\mu_1, \Sigma_1)$ and $\mathcal{N}(\mu_w, \Sigma_w)$ respectively. The mean and covariance are

$$\Sigma_1 = \frac{1}{1+N} I_d, \quad \mu_1 = \Sigma_1 \left(\sum_{n=1}^{N} X_n \right), \tag{7}$$

and

$$\Sigma_w = \frac{I_d}{1 + \left(\sum_{n=1}^{N} w_n \right)}, \quad \mu_w = \Sigma_w \left(\sum_{n=1}^{N} w_n X_n \right). \tag{8}$$

Proof of Proposition 1. By Eqs. (7) and (8),

$$\begin{aligned}
\mathrm{D}_{\mathrm{KL}}\left(\pi_w || \pi_1\right) =& \frac{1}{2}\left[\log \frac{|\Sigma_1|}{|\Sigma_w|} - d + \operatorname{tr}\left(\Sigma_1^{-1}\Sigma_w\right) + (\mu_1 - \mu_w)^T \Sigma_1^{-1}(\mu_1 - \mu_w)\right] \\
=& \frac{1}{2}\left[-d \log\left(\frac{1+N}{1+\sum_{n=1}^{N} w_n}\right) - d + d\left(\frac{1+N}{1+\sum_{n=1}^{N} w_n}\right) + (\mu_1 - \mu_w)^T \Sigma_1^{-1}(\mu_1 - \mu_w)\right].
\end{aligned}$$

Note that $\forall x > 0, x - 1 \geq \log x$, implying that

$$d \log\left(\frac{1+N}{1+\sum_{n=1}^{N} w_n}\right) - d + d\left(\frac{1+N}{1+\sum_{n=1}^{N} w_n}\right) > 0.$$

Thus,

$$\mathrm{D}_{\mathrm{KL}}\left(\pi_w || \pi_1\right) \geq \frac{1}{2}(\mu_1 - \mu_w)^T \Sigma_1^{-1}(\mu_1 - \mu_w).$$

Suppose we pick a set $\mathcal{I} \subseteq [N]$, $|\mathcal{I}| = M$ of active indices n where the optimal $w_n \geq 0$, and enforce that all others $n \notin \mathcal{I}$ satisfy $w_n = 0$. Then denoting

$$Y = [X_n : n \notin \mathcal{I}] \in \mathbb{R}^{d \times (N-M)}, \quad X = [X_n : n \in \mathcal{I}] \in \mathbb{R}^{d \times M},$$

we have that for any $w \in \mathbb{R}_+^M$ for those indices $\mathcal{I}$,

$$\begin{aligned}
\mathrm{D}_{\mathrm{KL}}\left(\pi_w || \pi_1\right) \geq& \frac{1}{2(N+1)} 1^T Y^T Y 1 + 1^T Y^T X \left(\frac{1}{N+1} - \frac{w}{1+1^T w}\right) \\
&+ \frac{N+1}{2}\left(\frac{1}{N+1} - \frac{w}{1+1^T w}\right)^T X^T X \left(\frac{1}{N+1} - \frac{w}{1+1^T w}\right).
\end{aligned}$$

Relaxing the nonnegativity constraint, replacing $w/(1+1^T w)$ with a generic $x \in \mathbb{R}^M$, and noting that $X^T X$ is invertible almost surely when $M < d$, we can optimize this expression yielding a lower bound on the optimal KL divergence using active index set $\mathcal{I}$,

$$\mathrm{D}_{\mathrm{KL}}\left(\pi_{w_{\mathcal{I}}^\star} || \pi_1\right) \geq \frac{1^T Y^T \left(I - X(X^T X)^{-1} X^T\right) Y 1}{2(N+1)}.$$

The numerator is the squared norm of $Y1$ minus its projection onto the subspace spanned by the M columns of X. Since $Y1 \sim \mathcal{N}(0, (N-M)I)$, $Y1 \in \mathbb{R}^d$ is an isotropic Gaussian, then its projection into the orthogonal complement of any fixed subspace of dimension M is also an isotropic Gaussian of dimension $d - M$ with the same variance. Since the columns of X are also independent and isotropic, its column subspace is uniformly distributed. So therefore, for each possible choice of $\mathcal{I}$

$$\mathrm{D}_{\mathrm{KL}}\left(\pi_{w_{\mathcal{I}}^\star} || \pi_1\right) \geq \frac{N-M}{2(N+1)} Z_{\mathcal{I}}, \quad Z_{\mathcal{I}} \sim \chi^2(d-M).$$

Note that the $Z_{\mathcal{I}}$ will have dependence across the $\binom{N}{M}$ different choices of index subset $\mathcal{I}$. Thus, the probability that *all* $Z_{\mathcal{I}}$ are large is

$$\begin{aligned}
\mathbb{P}\left(\min_{\mathcal{I} \subseteq [N], |\mathcal{I}| = M} Z_{\mathcal{I}} > \epsilon\right) \geq& 1 - \binom{N}{M} \mathbb{P}\left(Z_{\mathcal{I}} \leq \epsilon\right) \\
=& 1 - \binom{N}{M} F_{d-M}(\epsilon),
\end{aligned}$$

where F_k is the CDF for the χ^2 distribution with k degrees of freedom. The result follows. □

B Gradient Derivations

Throughout, expectations and covariances over the random parameter θ with no explicit subscripts are taken under pseudocoreset posterior $\pi_{u,w}$. We also interchange differentiation and integration without explicitly verifying that sufficient conditions to do so hold.

B.1 Weights gradient

First, we compute the gradient with respect to weights vector $w \in \mathbb{R}_+^M$, which is written as

$$\nabla_w \mathrm{D}_{\mathrm{KL}} = -\nabla_w \log Z(u,w) - \nabla_w \mathbb{E}[f(\theta)^T 1] + \nabla_w \mathbb{E}[\tilde{f}(\theta)^T w].$$

For any function $a : \Theta \to \mathbb{R}$, we have that

$$\begin{aligned}\nabla_w \mathbb{E}\left[a(\theta)\right] =& \int \nabla_w \left(\exp\left(w^T \tilde{f}(\theta) - \log Z(u,w)\right)\right) a(\theta)\pi_0(\theta)\mathrm{d}\theta \\ =& \mathbb{E}\left[\left(\tilde{f}(\theta) - \nabla_w \log Z(u,w)\right) a(\theta)\right].\end{aligned}$$

Next, we compute the gradient of the log normalization constant via

$$\begin{aligned}\nabla_w \log Z(u,w) =& \int \frac{1}{Z(u,w)} \nabla_w \left(\exp\left(w^T \tilde{f}(\theta)\right)\right) \pi_0(\theta)\mathrm{d}\theta \\ =& \mathbb{E}\left[\tilde{f}(\theta)\right].\end{aligned}$$

Combining, we have

$$\nabla_w \mathbb{E}\left[a(\theta)\right] = \mathbb{E}\left[\left(\tilde{f}(\theta) - \mathbb{E}\left[\tilde{f}(\theta)\right]\right) a(\theta)\right].$$

Subtracting $0 = \mathbb{E}\left[a(\theta)\right] \mathbb{E}\left[\tilde{f}(\theta) - \mathbb{E}\left[\tilde{f}(\theta)\right]\right]$ yields

$$\nabla_w \mathbb{E}\left[a(\theta)\right] = \mathrm{Cov}\left[\tilde{f}(\theta), a(\theta)\right].$$

The gradient with respect to w in Eq. (5) follows by substituting $1^T f(\theta)$ and $w^T \tilde{f}(\theta)$ for $a(\theta)$ and using the product rule.

B.2 Location gradients

Here we take the gradient with respect to a single pseudopoint $u_i \in \mathbb{R}^d$. First note that

$$\nabla_{u_i} \mathrm{D}_{\mathrm{KL}} = -\nabla_{u_i} \log Z(u,w) - \nabla_{u_i} \mathbb{E}[f(\theta)^T 1] + \nabla_{u_i} \mathbb{E}[\tilde{f}(\theta)^T w].$$

For any function $a(u,\theta) : \mathbb{R}^{d\times M} \times \Theta \to \mathbb{R}$, we have

$$\nabla_{u_i} \mathbb{E}\left[a(u,\theta)\right] = \int \nabla_{u_i} \Big(\exp\left(w^T \tilde{f}(\theta) - \log Z(u,w)\right) a(u,\theta)\Big) \pi_0(\theta)\mathrm{d}\theta.$$

Using the product rule and recalling from the main text that $h(\cdot,\theta) := \nabla_u f(\cdot,\theta)$,

$$\nabla_{u_i} \mathbb{E}\left[a(u,\theta)\right] = \mathbb{E}\left[\nabla_{u_i} a(u,\theta)\right] + \mathbb{E}\left[a(u,\theta)\left(w_i h(u_i,\theta) - \nabla_{u_i} \log Z(u,w)\right)\right].$$

Taking the gradient of the log normalization constant using similar techniques,

$$\nabla_{u_i} \log Z(u,w) = w_i \mathbb{E}\left[h(u_i,\theta)\right].$$

Combining,

$$\nabla_{u_i} \mathbb{E}\left[a(u,\theta)\right] = \mathbb{E}\left[\nabla_{u_i} a(u,\theta)\right] + w_i \mathbb{E}\left[a(u,\theta)\left(h(u_i,\theta) - \mathbb{E}\left[h(u_i,\theta)\right]\right)\right].$$

Subtracting $0 = \mathbb{E}\left[a(u,\theta)\right]\mathbb{E}\left[(h(u_i,\theta) - \mathbb{E}\left[h(u_i,\theta)\right])\right]$ yields

$$\nabla_{u_i}\mathbb{E}\left[a(u,\theta)\right] = \mathbb{E}\left[\nabla_{u_i}a(u,\theta)\right] + w_i \operatorname{Cov}\left[a(u,\theta), h(u_i,\theta)\right].$$

The gradient with respect to u_i in Eq. (5) follows by substituting $f(\theta)^T 1$ and $\tilde{f}(\theta)^T w$ for $a(u,\theta)$.

C Details on Experiments

C.1 Gaussian mean inference

Let the coreset posterior have mean $\mu_{u,w}$ and covariance matrix $\Sigma_{u,w}$. Throughout, expectations and covariances over the random parameter θ with no explicit subscripts are taken under pseudocoreset posterior $\pi_{u,w}$. Define $\Psi := Q^{-1}\Sigma_{u,w}Q^{-T}$, $v_n := Q^{-1}(x_n - \mu_{u,w})$, $\tilde{v}_n := Q^{-1}(u_n - \mu_{u,w})$, and Q to be the Cholesky decomposition of Σ, i.e. $\Sigma := QQ^T$. In order to compute the gradients in Eq. (5), we need expressions for $\operatorname{Cov}[f_n, f_m]$, $\operatorname{Cov}[\tilde{f}_n, f_m]$, $\operatorname{Cov}[h(u_i), f_n]$, and $\operatorname{Cov}[h(u_i), \tilde{f}_n]$. Following [6], we have that

$$\operatorname{Cov}[f_n, f_m] = v_n^T\Psi v_m + \frac{1}{2}\operatorname{tr}\Psi^T\Psi$$
$$\operatorname{Cov}[\tilde{f}_n, f_m] = \tilde{v}_n^T\Psi v_m + \frac{1}{2}\operatorname{tr}\Psi^T\Psi.$$

We now evaluate the remaining covariance $\operatorname{Cov}[h(u_i), f_m]$; the derivation of $\operatorname{Cov}[h(u_i), \tilde{f}_m]$ follows similarly. We begin by explicitly evaluating the log likelihood gradient and its expectation,

$$h(u_i) = -\Sigma^{-1}(u_i - \theta)$$
$$\mathbb{E}\left[h(u_i)\right] = -\Sigma^{-1}(u_i - \mu_{u,w}),$$

and again following [6], we have (up to a constant) that

$$f_n = -\frac{1}{2}(x_n - \theta)^T\Sigma^{-1}(x_n - \theta)$$
$$\mathbb{E}\left[f_n\right] = -\frac{1}{2}\operatorname{tr}\Psi - \frac{1}{2}\|v_n\|^2.$$

Thus using the above definitions,

$$\mathbb{E}\left[h(u_i)\right]\mathbb{E}\left[f_n\right] = \frac{\left(\operatorname{tr}\Psi + \|v_n\|^2\right)}{2}Q^{-T}\tilde{v}_i.$$

Next,

$$\mathbb{E}\left[h(u_i)f_n\right] = \frac{1}{2}\Sigma^{-1}\mathbb{E}\left[(u_i - \theta)(x_n - \theta)^T\Sigma^{-1}(x_n - \theta)\right].$$

Defining $z \sim \mathcal{N}(0, \Psi)$, and using the above definitions,

$$\mathbb{E}\left[h(u_i)f_n\right] = \frac{1}{2}Q^{-T}\mathbb{E}\left[(\tilde{v}_i - z)(v_n - z)^T(v_n - z)\right].$$

Evaluating the expectation, noting that odd order moments of z are equal to 0,

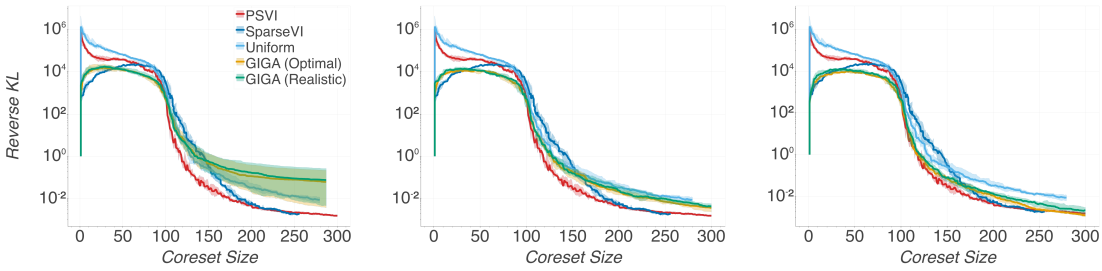
$$\mathbb{E}\left[h(u_i)f_n\right] = \frac{\|v_n\|^2 + \operatorname{tr}\Psi}{2}Q^{-T}\tilde{v}_i + Q^{-T}\Psi v_n.$$

Therefore,

$$\operatorname{Cov}[h(u_i), f_n] = Q^{-T}\Psi v_n,$$

and likewise,

$$\operatorname{Cov}[h(u_i), \tilde{f}_n] = Q^{-T}\Psi\tilde{v}_n.$$



project. dim. $= 200$ project. dim. $= 2,000$ project. dim. $= 10,000$

Figure 5: Comparison of Hilbert coresets performance on Bayesian linear regression experiment for increasing projection dimension (over 10 trials).

C.2 Bayesian linear regression

C.2.1 Model and gradients details

Here we present the terms involving pseudodata points—the corresponding expressions for original datapoints are the same, after replacing u_m with x_m.
For individual points, dropping normalization constants, we get log-likelihood terms of the form

$$f_m(\theta) = -\frac{1}{2\sigma^2}\left(y_m - \theta^T u_m\right)^2.$$

Hence, we obtain for the pseudocoreset posterior

$$\pi_{u,w} = \mathcal{N}(\mu_{u,w}, \Sigma_{u,w}), \quad \text{where}$$
$$\Sigma_{u,w} = (\sigma_0^{-2} I + \sigma^{-2} \sum_{m=1}^{M} w_m u_m u_m^T)^{-1}, \quad \mu_{u,w} = \Sigma_{u,w}(\sigma_0^{-2} I \mu_0 + \sigma^{-2} \sum_{m=1}^{M} w_m y_m u_m).$$

To scale up computation on large datasets, in our experiment we made use of stochastic gradients for black-box construction of `PSVI` and `SparseVI`. Beyond the expressions for individual log-likelihood and (pseudo)coreset posteriors presented above, for pseudocoreset construction we also need the expression for log-likelihood gradient with respect to the pseudodata points, for which we can immediately see that $\nabla_{u_m} f(u_m, \theta) = \frac{1}{\sigma^2}(y_m - \theta^T u_m)\theta$. Over our experiment, we optimized initial learning rates for `SparseVI` and `PSVI` via a grid search over $\{0.1, 1, 10\}$.

C.2.2 Additional plots

Here we present some more plots demonstating the dependence of Hilbert coresets approximation quality on the number of random dimensions in the Bayesian linear regression setting presented in Fig. 2c. We remind that dimension used at this experiment and throughout the entire experiments section was set to 100. Increasing this number is typically expensive to obtain in practice. As demonstrated in Fig. 5, getting higher projection dimension enables better posterior approximation in the problem, while `GIGA (Optimal)` starts offering better quality of approximation than `GIGA (Realistic)`. However, `PSVI` remains competitive in the small coreset regime even for Hilbert coresets with extremely large projection dimensionality, demonstrating the information-geometric limitations that Hilbert coreset constructions are known to face [6].

C.3 Bayesian Logistic Regression

C.3.1 Model

In logistic regression we have a set of datapoints $(x_n, y_n)_{n=1}^N$ each corresponding to a feature vector $x_n \in \mathbb{R}^d$ and a label $y_n \in \{-1, 1\}$. Datapoints are assumed to be generated according to following statistical model

$$y_n | x_n, \theta \sim \mathsf{Bern}\left(\frac{1}{1 + e^{-z_n \theta}}\right) \quad z_n := \begin{bmatrix} x_n \\ 1 \end{bmatrix}.$$

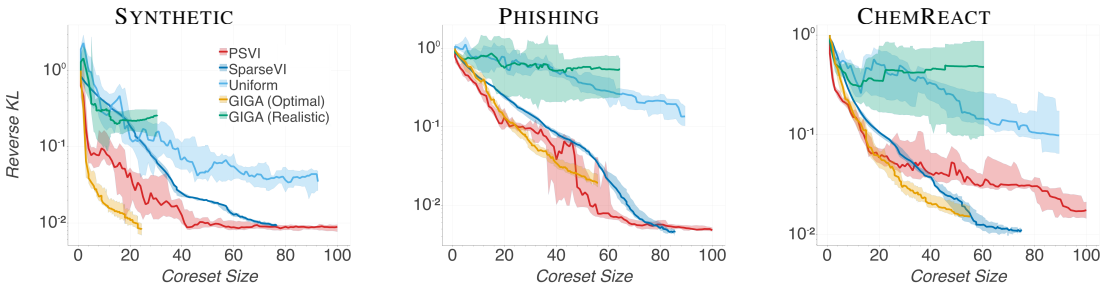


Figure 6: Comparison of (pseudo)coreset approximate posterior quality vs coreset size for logistic regression over 10 trials.

The aim of inference is to compute the posterior over the latent parameter $\theta = [\theta_0 \dots \theta_d]^T \in \mathbb{R}^{d+1}$. Log-likelihood of each datapoint can be expressed as

$$\begin{aligned} f_n(x_n, y_n|\theta) =& \mathbb{1}[y_n = -1] \log\left(1 - \frac{1}{1 + e^{-z_n^T\theta}}\right) - \mathbb{1}[y_n = 1] \log\left(1 + e^{-z_n^T\theta}\right) \\ =& - \log\left(1 + \exp(-y_n z_n^T \theta)\right). \end{aligned}$$

Hence in pseudocoreset construction we can optimize pseudodata point locations with respect to continuous variable x_n, using the gradient

$$\nabla_{x_n} f_n = \frac{e^{-y_n z_n^T\theta}}{1 + e^{-y_n z_n^T\theta}} y_n \begin{bmatrix} \theta_1 \\ \vdots \\ \theta_d \end{bmatrix}.$$

C.3.2 Datasets description

For logistic regression experiments, we used subsampled and full versions of datasets presented in Table 1: a synthetic dataset with $x \in \mathbb{R}^2$ sampled i.i.d. from a $\mathcal{N}(0, I)$ and $y \in \{-1, 1\}$ sampled from respective logistic likelihood with $\theta = [3, 3, 0]^T$ (SYNTHETIC); a phishing websites dataset reduced to $D = 10$ via PCA (PHISHING); a chemical reactivity dataset with real-valued features corresponding to its first 10 and 100 principal components (CHEMREACT and CHEMREACT100 respectively); a dataset with 50 real-valued features associated with whether each of $100K$ customers of a bank will make a specific transaction (TRANSACTIONS); and a dataset for music analysis, where we consider "classical vs all" genre classification task (MUSIC). Original versions of the four latter datasets are available online respectively at https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/binary.html, http://komarix.org/ac/ds, https://www.kaggle.com/c/santander-customer-transaction-prediction/data, and https://github.com/mdeff/fma.

Dataset name	N	D
SYNTHETIC	500	2
PHISHING	500	10
CHEMREACT	500	10
TRANSACTIONS	100,000	50
CHEMREACT100	26,733	100
MUSIC	8,419	237

Table 1: Details for datasets used in logistic regression experiments.

C.3.3 Small-scale experiments

In the small-scale experiment, the number of overall gradient updates was set to $T = 1,500$, while minibatch size was set to $B = 400$. Learning rate schedule for `SparseVI` and `PSVI` was $\gamma_t = 0.1t^{-1}$. Results presented in Fig. 6 indicate that `PSVI` achieves superior quality to `SparseVI` for small coreset sizes, and is competitive to `GIGA (Optimal)`, while the latter unrealistically uses true posterior samples to tune a weighting function required over construction.

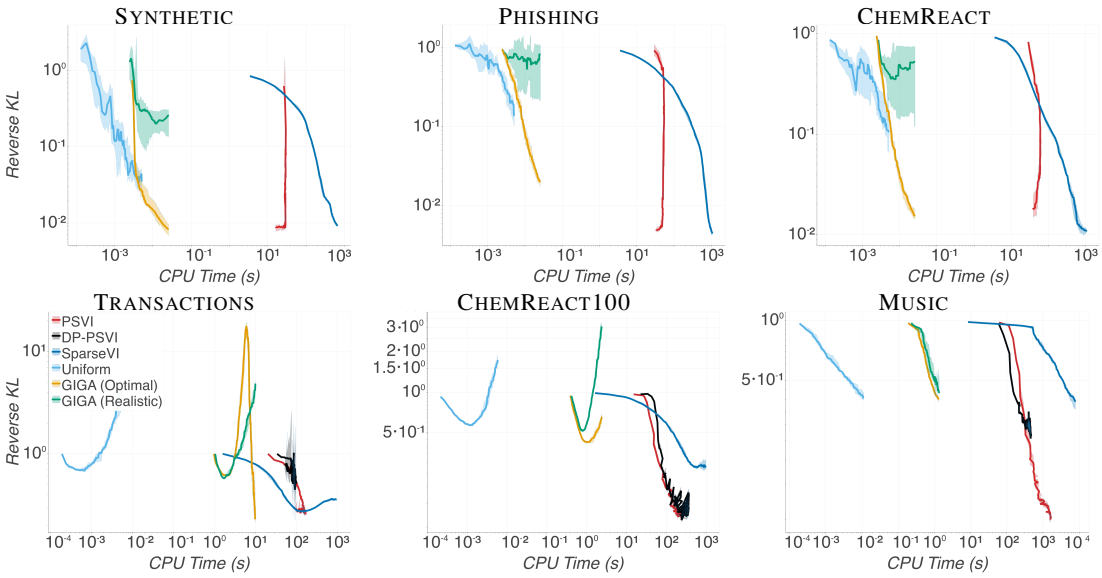


Figure 7: Comparison of `PSVI` and `SparseVI` approximate posterior quality vs CPU time requirements for logistic regression experiment of Section 4.

C.3.4 Reproducibility of Bayesian Logistic Regression experiment

In this subsection we provide additional details for reproducibility of the experimental setup for the Bayesian Logistic Regression experiment presented in Section 4.

Posterior approximation metrics, coreset gradients and learning rates Posterior approximation quality was estimated via computing KL divergence between Gaussian distributions fitted on coreset and full data posteriors via Laplace approximation. For both `SparseVI` and `PSVI`, gradients were estimated using samples drawn from a Laplace approximation fitted on current coreset weights and points. To optimize initial learning rates for `SparseVI` and `PSVI`, we did a grid search over $\{0.1, 1, 10\}$.

Differential privacy loss accounting and hyperparameter selection In the differential privacy experiment, we were not concerned with the extra privacy cost of hyperparameter optimization task. Estimation of differential privacy cost at all experiments was based on TensorFlow privacy implementation of moments accountant for the subsampled Gaussian mechanism[1]. For `DP-PSVI` we used the best learning hyperparameters found for `PSVI` on the corresponding dataset. The demonstrated range of privacy budgets was generated by decreasing the variance σ of additive Gaussian noise and keeping the rest of hyperparameters involved in privacy accounting fixed. Regarding `DP-VI`, over our experiments we also kept subsampling ratio fixed. We based our implementation of `DP-VI` on authors code,[2] adapting noise calibration according to the adjacency relation used in Section 3.3, and the standard differential privacy definition [15]. In our experiment, we used the AdaGrad optimizer [11], with learning rate 0.01, number of iterations $2,000$, and minibatch size 200. Gradient clipping values for `DP-VI` results presented in Fig. 4, for TRANSACTIONS, CHEMREACT100, and MUSIC datasets were tuned via grid search over $\{1, 5, 10, 50\}$. The values of gradient clipping constant giving best privacy profiles for each dataset, used in Fig. 4, were 10, 5, and 5 respectively.

C.3.5 Additional Plots

Evaluation of CPU time requirements Experiments were performed on a CPU cluster node with a 2x Intel Xeon Gold 6142 and 12GB RAM. In the case of `PSVI` the computation of coreset sizes from 1 to 100 was parallelized per single size over 32 cores in total. Fig. 7 shows posterior approximation error vs required CPU time for all coreset construction algorithms over logistic regression on the small-scale and large-scale datasets. As opposed to existing incremental coreset construction schemes, batch construction of `PSVI` reduces the dependence between coreset size and processing cost: for

[1] https://github.com/tensorflow/privacy
[2] https://github.com/DPBayes/DPVI-code

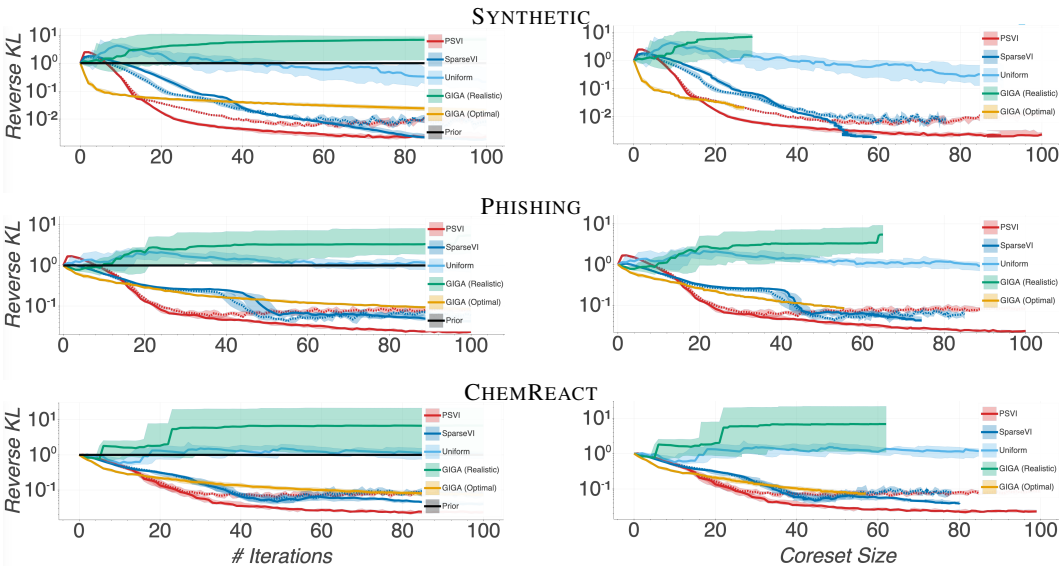


Figure 8: Comparison of incremental PSVI and SparseVI approximate posterior quality vs iterations of incremental construction (*left*) and coreset size (*right*) for logistic regression on small-scale experiment. With dashed lines is displayed the posterior quality achieved by incremental PSVI and SparseVI constructions using gradients computed on data subsets of size 256.

SparseVI $\Theta(M^2)$ gradient computations are required, as this method builds up a coreset one point at a time; in contrast, PSVI requires $\Theta(M)$ gradients since it learns all pseudodata points jointly. Although each gradient step of PSVI is more expensive, practically this implies a steeper decrease in approximation error over processing time compared to SparseVI. In the case of differentially private PSVI, some extra CPU requirements are added due to the subsampled Gaussian mechanism computations.

Incremental scheme for pseudocoreset construction We also experimented with an *incremental scheme for pseudocoreset* construction. According to this scheme, pseudodata points are added sequentially to the pseudocoreset. Similarly to SparseVI, in the beginning of each coreset iteration, we initialize a new pseudodata point at the true datapoint which maximizes correlation with current residual approximation error. Next, we jointly optimize the most recently added pseudodata point location, along with the pseudocoreset weights vector, over a gradient descent loop. As opposed to batch construction, for large coreset sizes the incremental scheme for PSVI does not achieve savings in CPU time compared to SparseVI.

We evaluated coreset construction methods on Bayesian logistic regression. We used $M = 100$ iterations for construction, $S = 100$ Monte Carlo samples per gradient estimation, $T = 100$ iterations for optimization, and learning rate $\gamma_t \propto 0.5t^{-1}$. Coreset posterior samples over the course of construction for SparseVI and incremental PSVI were drawn from a Laplace approximation using current coreset weights and points. We implemented SparseVI and incremental PSVI via computing gradients on the full dataset, as well as using stochastic gradients on subsets of size $B = 256$ for lowering computational cost.

Results presented in Fig. 8 demonstrate that incremental PSVI achieves consistently the smallest posterior approximation error, offering improvement compared to SparseVI and even achieving better performance than GIGA (Optimal). We observe that stochastic gradients implementation (dashed lines) reaches a plateau at higher values of KL compared to full gradients (solid lines), but still achieves performance comparable with GIGA (Optimal).