Bayesian Coreset Construction via Greedy Iterative Geodesic Ascent

Trevor Campbell [1]   **Tamara Broderick** [1]

Abstract

Coherent uncertainty quantification is a key strength of Bayesian methods. But modern algorithms for approximate Bayesian posterior inference often sacrifice accurate posterior uncertainty estimation in the pursuit of scalability. This work shows that previous *Bayesian coreset* construction algorithms—which build a small, weighted subset of the data that approximates the full dataset—are no exception. We demonstrate that these algorithms scale the coreset log-likelihood suboptimally, resulting in underestimated posterior uncertainty. To address this shortcoming, we develop *greedy iterative geodesic ascent (GIGA)*, a novel algorithm for Bayesian coreset construction that scales the coreset log-likelihood optimally. GIGA provides geometric decay in posterior approximation error as a function of coreset size, and maintains the fast running time of its predecessors. The paper concludes with validation of GIGA on both synthetic and real datasets, demonstrating that it reduces posterior approximation error by orders of magnitude compared with previous coreset constructions.

1. Introduction

Bayesian methods provide a wealth of options for principled parameter estimation and uncertainty quantification. But Markov chain Monte Carlo (MCMC) methods (Robert & Casella, 2004; Neal, 2011; Hoffman & Gelman, 2014), the gold standard for Bayesian inference, typically have complexity $\Theta(NT)$ for dataset size N and number of samples T and are intractable for modern large-scale datasets. Scalable methods (see (Angelino et al., 2016) for a recent survey), on the other hand, often sacrifice the strong guarantees of MCMC and provide unreliable posterior approximations. For example, variational methods and their scalable and

streaming variants (Jordan et al., 1999; Wainwright & Jordan, 2008; Hoffman et al., 2013; Ranganath et al., 2014; Broderick et al., 2013; Campbell & How, 2014; Campbell et al., 2015; Dieng et al., 2017) are both susceptible to finding bad local optima in the variational objective and tend to either over- or underestimate posterior variance depending on the chosen discrepancy and variational family.

Bayesian coresets (Huggins et al., 2016; Campbell & Broderick, 2017) provide an alternative approach—based on the observation that large datasets often contain redundant data—in which a small subset of the data of size $M \ll \min\{N, T\}$ is selected and reweighted such that it preserves the statistical properties of the full dataset. The coreset can be passed to a standard MCMC algorithm, providing posterior inference with theoretical guarantees at a significantly reduced $O(M(N + T))$ computational cost. But despite their advantages, existing Bayesian coreset constructions—like many other scalable inference methods—tend to underestimate posterior variance (Fig. 1). This effect is particularly evident when the coreset is small, which is the regime we are interested in for scalable inference.

In this work, we show that existing Bayesian coreset constructions underestimate posterior uncertainty because they scale the coreset log-likelihood suboptimally in order to remain unbiased (Huggins et al., 2016) or to keep their weights in a particular constraint polytope (Campbell & Broderick, 2017). The result is an overweighted coreset with too much "artificial data," and therefore an overly certain posterior. Taking this intuition to its limit, we demonstrate that there exist models for which previous algorithms output coresets with arbitrarily large relative posterior approximation error at any coreset size (Proposition 2.1). We address this issue by developing a novel coreset construction algorithm, *greedy iterative geodesic ascent (GIGA)*, that optimally scales the coreset log-likelihood to best fit the full dataset log-likelihood. GIGA has the same computational complexity as the current state of the art, but its optimal log-likelihood scaling leads to uniformly bounded relative error for *all* models, as well as asymptotic exponential error decay (Theorem 3.1 and Corollary 3.2). The paper concludes with experimental validation of GIGA on a synthetic vector approximation problem as well as regression models applied to multiple real and synthetic datasets.

---

[1]Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, United States. Correspondence to: Trevor Campbell <tdjc@mit.edu>.
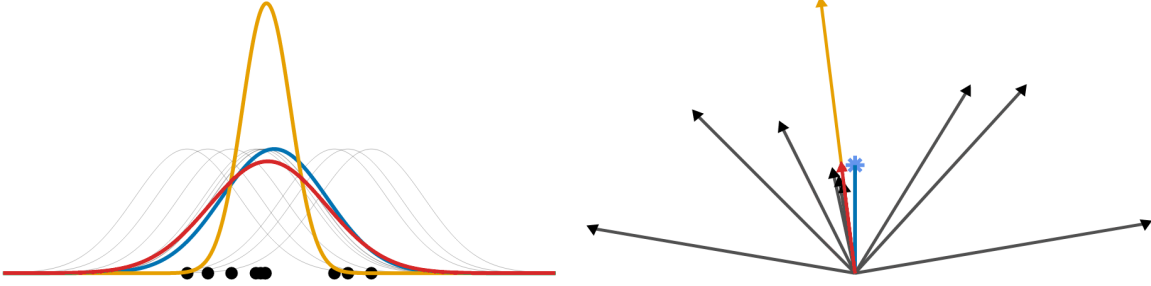
Figure 1. (Left) Gaussian inference for an unknown mean, showing data (black points and likelihood densities), exact posterior (blue), and optimal coreset posterior approximations of size 1 from solving the original coreset construction problem Eq. (3) (red) and the modified problem Eq. (5) (orange). The orange coreset posterior has artificially low uncertainty. The exact and approximate log-posteriors are scaled down (by the same amount) for visualization. (Right) The vector formulation of log-likelihoods using the same color scheme.

2. Bayesian Coresets

2.1. Background

In Bayesian statistical modeling, we are given a dataset $(y_n)_{n=1}^N$ of N observations, a likelihood $p(y_n|\theta)$ for each observation given a parameter $\theta \in \Theta$, and a prior density $\pi_0(\theta)$ on Θ. We assume that the data are conditionally independent given θ. The Bayesian posterior is given by

$$\pi(\theta) := \frac{1}{Z} \exp(\mathcal{L}(\theta))\pi_0(\theta), \tag{1}$$

where the log-likelihood $\mathcal{L}(\theta)$ is defined by

$$\mathcal{L}_n(\theta) := \log p(y_n \mid \theta), \quad \mathcal{L}(\theta) := \sum_{n=1}^N \mathcal{L}_n(\theta), \tag{2}$$

and Z is the marginal likelihood. MCMC returns approximate samples from the posterior, which can be used to construct an empirical approximation to the posterior distribution. Since each sample requires at least one full likelihood evaluation—typically an $\Theta(N)$ operation—MCMC has $\Theta(NT)$ complexity for T posterior samples.

To reduce the complexity, we can instead run MCMC on a *Bayesian coreset* (Huggins et al., 2016), a small, weighted subset of the data. Let $w \in \mathbb{R}^N$ be the vector of non-negative weights, with weight w_n for data point y_n, and $\|w\|_0 := \sum_{n=1}^N \mathbb{1}[w_n > 0] \ll N$. Then we approximate the full log-likelihood with $\mathcal{L}(w, \theta) := \sum_{n=1}^N w_n \mathcal{L}_n(\theta)$ and run MCMC with the approximated likelihood. By viewing the log-likelihood functions $\mathcal{L}_n(\theta), \mathcal{L}(\theta), \mathcal{L}(w, \theta)$ as vectors $\mathcal{L}_n, \mathcal{L}, \mathcal{L}(w)$ in a normed vector space, Campbell & Broderick (2017) pose the problem of constructing a coreset of size M as cardinality-constrained vector approximation,

$$\min_{w \in \mathbb{R}^N} \quad \|\mathcal{L}(w) - \mathcal{L}\|^2 \tag{3}$$
$$\text{s.t.} \quad w \geq 0, \quad \|w\|_0 \leq M.$$

Solving Eq. (3) exactly is not tractable for large N due to the cardinality constraint; approximation is required. Given a norm induced by an inner product, and defining

$$\sigma_n := \|\mathcal{L}_n\| \quad \text{and} \quad \sigma := \sum_{n=1}^N \sigma_n, \tag{4}$$

Campbell & Broderick (2017) replace the cardinality constraint in Eq. (3) with a simplex constraint,

$$\min_{w \in \mathbb{R}^N} \quad \|\mathcal{L}(w) - \mathcal{L}\|^2 \tag{5}$$
$$\text{s.t.} \quad w \geq 0, \quad \sum_{n=1}^N \sigma_n w_n = \sigma.$$

Eq. (5) can be solved while ensuring $\|w\|_0 \leq M$ using either importance sampling (IS) or Frank–Wolfe (FW) (Frank & Wolfe, 1956). Both procedures add one data point to the linear combination $\mathcal{L}(w)$ at each iteration; IS chooses the new data point i.i.d. with probability σ_n/σ, while FW chooses the point most aligned with the residual error. This difference in how the coreset is built results in different convergence behavior: FW exhibits geometric convergence $\|\mathcal{L}(w) - \mathcal{L}\|^2 = O(\nu^M)$ for some $0 < \nu < 1$, while IS is limited by the Monte Carlo rate $\|\mathcal{L}(w) - \mathcal{L}\|^2 = O(M^{-1})$ with high probability (Campbell & Broderick, 2017, Theorems 4.1, 4.4). For this reason, FW is the preferred method for coreset construction.

Eq. (3) is a special case of the sparse vector approximation problem, which has been studied extensively in past literature. Convex optimization formulations—e.g. basis pursuit (Chen et al., 1999), LASSO (Tibshirani, 1996), the Dantzig selector (Candès & Tao, 2007), and compressed sensing (Candès & Tao, 2005; Donoho, 2006; Boche et al., 2015)—are expensive to solve compared to our greedy approach, and often require tuning regularization coefficients and thresholding to ensure cardinality constraint feasibility. Previous greedy iterative algorithms—e.g. (orthogonal)

matching pursuit (Mallat & Zhang, 1993; Chen et al., 1989; Tropp, 2004), Frank–Wolfe and its variants (Frank & Wolfe, 1956; Guélat & Marcotte, 1986; Jaggi, 2013; Lacoste-Julien & Jaggi, 2015; Locatello et al., 2017), Hilbert space vector approximation methods (Barron et al., 2008), kernel herding (Chen et al., 2010), and AdaBoost (Freund & Schapire, 1997)—have sublinear error convergence unless computationally expensive correction steps are included. In contrast, the algorithm developed in this paper has no correction steps, no tuning parameters, and geometric error convergence.

2.2. Posterior Uncertainty Underestimation

The new $\sum_n \sigma_n w_n = \sigma$ constraint in Eq. (5) has an unfortunate practical consequence: both IS and FW must scale the coreset log-likelihood suboptimally—roughly, by σ rather than $\|\mathcal{L}\|$ as they should—in order to maintain feasibility. Since $\sigma \geq \|\mathcal{L}\|$, intuitively the coreset construction algorithms are adding too much "artificial data" via the coreset weights, resulting in an overly certain posterior approximation. It is worth noting that this effect is apparent in the error bounds developed by Campbell & Broderick (2017), which are all proportional to σ rather than $\|\mathcal{L}\|$ as one might hope for when approximating $\mathcal{L}$.

Fig. 1 provides intuition in the setting of Gaussian inference for an unknown mean. In this example, we construct the optimal coreset of size 1 for the modified problem in Eq. (5) (orange) and for the original coreset construction problem that we would ideally like to solve in Eq. (3) (red). Compared with the exact posterior (blue), the orange coreset approximation from Eq. (5) has artificially low uncertainty, since it must place weight σ/σ_n on its chosen point. Building on this intuition, Proposition 2.1 shows that there are problems[1] for which both FW and IS perform arbitrarily poorly for any number of iterations M.

Proposition 2.1. *For any $M \in \mathbb{N}$, there exists $(\mathcal{L}_n)_{n=1}^N$ for which both the FW and IS coresets after M iterations have arbitrarily large error relative to $\|\mathcal{L}\|$.*

Proof. Let $\mathcal{L}_n = 1/N 1_n$ where 1_n is the indicator for the nth component of $\mathbb{R}^N$, and let $\mathcal{L} = \sum_{n=1}^N \mathcal{L}_n$. Then in the 2-norm, $\sigma_n = 1/N$, $\sigma = 1$, and $\|\mathcal{L}\| = 1/\sqrt{N}$. By symmetry, the optimal w for Eq. (5) satisfying $\|w\|_0 \leq M$ has uniform nonzero weights N/M. Substituting yields

$$\frac{\|\mathcal{L}(w) - \mathcal{L}\|}{\|\mathcal{L}\|} = \sqrt{\frac{N}{M} - 1}, \tag{6}$$

which can be made as large as desired by increasing N. The result follows since both FW and IS generate a feasible solution w for Eq. (5) satisfying $\|w\|_0 \leq M$. $\square$

---

[1]While the proof uses orthogonal vectors in $\mathbb{R}^N$ for simplicity, a similar construction arises from the model $\theta \sim \mathcal{N}(0, I)$, $x_n \sim \mathcal{N}(\theta_n, 1)$, $n \in [N]$ given the norm $\|\mathcal{L}\| = \sqrt{\mathbb{E}_\pi [\|\nabla \mathcal{L}\|^2]}$.

In contrast, the red coreset approximation obtained by solving Eq. (3) scales its weight to minimize $\|\mathcal{L}(w) - \mathcal{L}\|$, resulting in a significantly better approximation of posterior uncertainty. Note that we can scale the weight vector w by any $\alpha \geq 0$ without affecting feasibility in the cardinality constraint, i.e., $\|\alpha w\|_0 \leq \|w\|_0$. In the following section, we use this property to develop a greedy coreset construction algorithm that, unlike FW and IS, maintains optimal log-likelihood scaling in each iteration.

3. Greedy Iterative Geodesic Ascent (GIGA)

In this section, we provide a new algorithm for Bayesian coreset construction and demonstrate that it yields improved approximation error guarantees proportional to $\|\mathcal{L}\|$ (rather than $\sigma \geq \|\mathcal{L}\|$). We begin in Section 3.1 by solving for the optimal log-likelihood scaling analytically. After solving this "radial optimization problem," we are left with a new optimization problem on the unit hyperspherical manifold. In Sections 3.2 and 3.3, we demonstrate how to solve this new problem by iteratively building the coreset one point at a time. Since this procedure selects the point greedily based on a geodesic alignment criterion, we call it *greedy iterative geodesic ascent (GIGA)*, detailed in Algorithm 1. In Section 3.4 we scale the resulting coreset optimally using the procedure developed in Section 3.1. Finally, in Section 3.5, we prove that Algorithm 1 provides approximation error that is proportional to $\|\mathcal{L}\|$ and geometrically decaying in M, as shown by Theorem 3.1.

Theorem 3.1. *The output w of Algorithm 1 satisfies*

$$\|\mathcal{L}(w) - \mathcal{L}\| \leq \eta \|\mathcal{L}\| \nu_M, \tag{7}$$

where ν_M is decreasing and ≤ 1 for all $M \in \mathbb{N}$, $\nu_M = O(\nu^M)$ for some $0 < \nu < 1$, and η is defined by

$$0 \leq \eta := \sqrt{1 - \left(\max_{n \in [N]} \left\langle \frac{\mathcal{L}_n}{\|\mathcal{L}_n\|}, \frac{\mathcal{L}}{\|\mathcal{L}\|} \right\rangle \right)^2} \leq 1. \tag{8}$$

The particulars of the sequence $(\nu_M)_{M=1}^\infty$ are somewhat involved; see Section 3.5 for the detailed development. A straightforward consequence of Theorem 3.1 is that the issue described in Proposition 2.1 has been resolved; the solution $\mathcal{L}(w)$ is always scaled optimally, leading to decaying relative error. Corollary 3.2 makes this notion precise.

Corollary 3.2. *For any set of vectors $(\mathcal{L}_n)_{n=1}^N$, Algorithm 1 provides a solution to Eq. (3) with error ≤ 1 relative to $\|\mathcal{L}\|$.*

3.1. Solving the Radial Optimization

We begin again with the problem of coreset construction for a collection of vectors $(\mathcal{L}_n)_{n=1}^N$ given by Eq. (3). Without loss of generality, we assume $\|\mathcal{L}\| > 0$ and $\forall n \in [N]$,

Algorithm 1 GIGA: Greedy Iterative Geodesic Ascent

Require: $(\mathcal{L}_n)_{n=1}^N$, M, $\langle \cdot, \cdot \rangle$
   ▷ Normalize vectors and initialize weights to 0
1: $\mathcal{L} \leftarrow \sum_{n=1}^N \mathcal{L}_n$
2: $\forall n \in [N]$ $\ell_n \leftarrow \frac{\mathcal{L}_n}{\|\mathcal{L}_n\|}, \ell \leftarrow \frac{\mathcal{L}}{\|\mathcal{L}\|}$
3: $w_0 \leftarrow 0, \ell(w_0) \leftarrow 0$
4: **for** $t \in \{0, \ldots, M-1\}$ **do**
   ▷ Compute the geodesic direction for each data point
5:     $d_t \leftarrow \frac{\ell - \langle \ell, \ell(w_t) \rangle \ell(w_t)}{\|\ell - \langle \ell, \ell(w_t) \rangle \ell(w_t)\|}$
6:     $\forall n \in [N], d_{tn} \leftarrow \frac{\ell_n - \langle \ell_n, \ell(w_t) \rangle \ell(w_t)}{\|\ell_n - \langle \ell_n, \ell(w_t) \rangle \ell(w_t)\|}$
   ▷ Choose the best geodesic
7:     $n_t \leftarrow \arg\max_{n \in [N]} \langle d_t, d_{tn} \rangle$
8:     $\zeta_0 \leftarrow \langle \ell, \ell_{n_t} \rangle, \zeta_1 \leftarrow \langle \ell, \ell(w_t) \rangle, \zeta_2 \leftarrow \langle \ell_{n_t}, \ell(w_t) \rangle$
   ▷ Compute the step size
9:     $\gamma_t \leftarrow \frac{\zeta_0 - \zeta_1 \zeta_2}{(\zeta_0 - \zeta_1 \zeta_2) + (\zeta_1 - \zeta_0 \zeta_2)}$
   ▷ Update the coreset
10:    $\ell(w_{t+1}) \leftarrow \frac{(1-\gamma_t)\ell(w_t) + \gamma_t \ell_{n_t}}{\|(1-\gamma_t)\ell(w_t) + \gamma_t \ell_{n_t}\|}$
11:    $w_{t+1} \leftarrow \frac{(1-\gamma_t)w_t + \gamma_t 1_{n_t}}{\|(1-\gamma_t)\ell(w_t) + \gamma_t \ell_{n_t}\|}$
12: **end for**
   ▷ Scale the weights optimally
13: $\forall n \in [N], w_{Mn} \leftarrow w_{Mn} \frac{\|\mathcal{L}\|}{\|\mathcal{L}_n\|} \langle \ell(w_M), \ell \rangle$
14: **return** w

$\|\mathcal{L}_n\| > 0$; if $\|\mathcal{L}\| = 0$ then $w = 0$ is a trivial optimal solution, and any $\mathcal{L}_n$ for which $\|\mathcal{L}_n\| = 0$ can be removed from the coreset without affecting the objective in Eq. (3). Recalling that the weights of a coreset can be scaled by an arbitrary constant $\alpha \geq 0$ without affecting feasibility, we rewrite Eq. (3) as

$$\min_{\alpha \in \mathbb{R}, w \in \mathbb{R}^N} \quad \|\alpha \mathcal{L}(w) - \mathcal{L}\|^2 \tag{9}$$
$$\text{s.t.} \quad \alpha \geq 0, w \geq 0, \|w\|_0 \leq M.$$

Taking advantage of the fact that $\| \cdot \|$ is induced by an inner product, we can expand the objective in Eq. (9) as a quadratic function of α, and analytically solve the optimization in α to yield

$$\alpha^\star = \frac{\|\mathcal{L}\|}{\|\mathcal{L}(w)\|} \max \left\{ 0, \left\langle \frac{\mathcal{L}(w)}{\|\mathcal{L}(w)\|}, \frac{\mathcal{L}}{\|\mathcal{L}\|} \right\rangle \right\}. \tag{10}$$

In other words, $\mathcal{L}(w)$ should be rescaled to have norm $\|\mathcal{L}\|$, and then scaled further depending on its directional alignment with $\mathcal{L}$. We substitute this result back into Eq. (9) to find that coreset construction is equivalent to solving

$$\min_{w \in \mathbb{R}^N} \quad \|\mathcal{L}\|^2 \left(1 - \max \left\{ 0, \left\langle \frac{\mathcal{L}(w)}{\|\mathcal{L}(w)\|}, \frac{\mathcal{L}}{\|\mathcal{L}\|} \right\rangle \right\}^2 \right) \tag{11}$$
$$\text{s.t.} \quad w \geq 0, \|w\|_0 \leq M.$$

Note that by selecting the optimal scaling via Eq. (10), the cost now depends only on the alignment of $\mathcal{L}(w)$ and $\mathcal{L}$

irrespective of their norms. Finally, defining the normalized vectors $\ell_n := \frac{\mathcal{L}_n}{\|\mathcal{L}_n\|}$, $\ell := \frac{\mathcal{L}}{\|\mathcal{L}\|}$, and $\ell(w) := \sum_{n=1}^N w_n \ell_n$, solving Eq. (11) is equivalent to maximizing the alignment of ℓ and $\ell(w)$ over $\ell(w)$ lying on the unit hypersphere:

$$\max_{w \in \mathbb{R}^N} \quad \langle \ell(w), \ell \rangle \tag{12}$$
$$\text{s.t.} \quad w \geq 0, \|w\|_0 \leq M, \|\ell(w)\| = 1.$$

Compare this optimization problem to Eq. (5); Eq. (12) shows that the natural choice of manifold for optimization is the *unit hypersphere* rather than the simplex. Therefore, in Sections 3.2 and 3.3, we develop a greedy optimization algorithm that operates on the hyperspherical manifold. At each iteration, the algorithm picks the point indexed by n for which the geodesic between $\ell(w)$ and ℓ_n is most aligned with the geodesic between $\ell(w)$ and ℓ. Once the point has been added, it reweights the coreset and iterates.

3.2. Coreset Initialization

For the remainder of this section, denote the value of the weights at iteration t by $w_t \in \mathbb{R}_+^N$, and write the nth component of this vector as w_{tn}. To initialize the optimization, we select the vector ℓ_n most aligned with ℓ, i.e.

$$n_1 = \arg\max_{n \in [N]} \langle \ell_n, \ell \rangle, \quad w_1 = 1_{n_1}. \tag{13}$$

This initialization provides two benefits: for all iterations t, there is no point ℓ_n in the hyperspherical cap $\{y : \|y\| = 1, \langle y, \ell \rangle \geq r\}$ for any $r > \langle \ell(w_1), \ell \rangle$, as well as $\langle \ell(w_1), \ell \rangle > 0$ by Lemma 3.3. We use these properties to develop the error bound in Theorem 3.1 and guarantee that the iteration step developed below in Section 3.3 is always feasible. Note that Algorithm 1 does not explicitly run this initialization and simply sets $w_0 = 0$, since the initialization for w_1 is equivalent to the usual iteration step described in Section 3.3 after setting $w_0 = 0$.

Lemma 3.3. *The initialization satisfies*

$$\langle \ell(w_1), \ell \rangle \geq \frac{\|\mathcal{L}\|}{\sigma} > 0. \tag{14}$$

Proof. For any $\xi \in \Delta^{N-1}$, we have that

$$\langle \ell(w_1), \ell \rangle = \max_{n \in [N]} \langle \ell_n, \ell \rangle \geq \sum_n \xi_n \langle \ell_n, \ell \rangle. \tag{15}$$

So choosing $\xi_n = \sigma_n / \sigma$,

$$\langle \ell(w_1), \ell \rangle \geq \frac{1}{\sigma} \left\langle \sum_n \mathcal{L}_n, \ell \right\rangle = \frac{\|\mathcal{L}\|}{\sigma} \langle \ell, \ell \rangle = \frac{\|\mathcal{L}\|}{\sigma}. \tag{16}$$

By assumption (see Section 3.1), we have $\|\mathcal{L}\| > 0$. $\qquad\square$
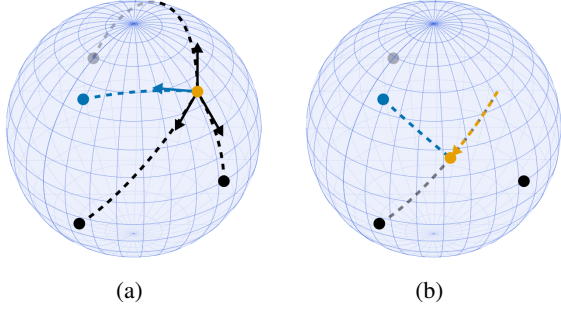
(a)   (b)

Figure 2. Fig. 2a shows the greedy geodesic selection procedure, with normalized total log-likelihood ℓ in blue, normalized data log-likelihoods ℓ_n in black, and the current iterate $\ell(w)$ in orange. Fig. 2b shows the subsequent line search procedure.

3.3. Greedy Point Selection and Reweighting

At each iteration t, we have a set of weights $w_t \in \mathbb{R}_+^N$ for which $\|\ell(w_t)\| = 1$ and $\|w_t\|_0 \leq t$. The next point to add to the coreset is the one for which the geodesic direction is most aligned with the direction of the geodesic from $\ell(w_t)$ to ℓ; Fig. 2a illustrates this selection. Precisely, we define

$$d_t := \frac{\ell - \langle \ell, \ell(w_t) \rangle \ell(w_t)}{\|\ell - \langle \ell, \ell(w_t) \rangle \ell(w_t)\|}$$
$$\forall n \in [N], \quad d_{tn} := \frac{\ell_n - \langle \ell_n, \ell(w_t) \rangle \ell(w_t)}{\|\ell_n - \langle \ell_n, \ell(w_t) \rangle \ell(w_t)\|}, \quad (17)$$

and we select the data point at index n_t where

$$n_t = \arg\max_{n \in [N]} \langle d_t, d_{tn} \rangle. \quad (18)$$

This selection leads to the largest decrease in approximation error, as shown later in Eq. (28). For any vector u with $\|u\| = 0$, we define $\frac{u}{\|u\|} := 0$ to avoid any issues with Eq. (18). Once the vector ℓ_{n_t} has been selected, the next task is to redistribute the weights between $\ell(w_t)$ and ℓ_{n_t} to compute w_{t+1}. We perform a search along the geodesic from $\ell(w_t)$ to ℓ_{n_t} to maximize $\langle \ell(w_{t+1}), \ell \rangle$, as shown in Fig. 2b:

$$\gamma_t = \arg\max_{\gamma \in [0,1]} \left\langle \ell, \frac{\ell(w_t) + \gamma(\ell_{n_t} - \ell(w_t))}{\|\ell(w_t) + \gamma(\ell_{n_t} - \ell(w_t))\|} \right\rangle. \quad (19)$$

Constraining $\gamma \in [0, 1]$ ensures that the resulting w_{t+1} is feasible for Eq. (12). Taking the derivative and setting it to 0 yields the unconstrained optimum of Eq. (19),

$$\zeta_0 := \langle \ell, \ell_{n_t} \rangle \;\; \zeta_1 := \langle \ell, \ell(w_t) \rangle \;\; \zeta_2 := \langle \ell_{n_t}, \ell(w_t) \rangle \quad (20)$$
$$\gamma_t = \frac{(\zeta_0 - \zeta_1\zeta_2)}{(\zeta_0 - \zeta_1\zeta_2) + (\zeta_1 - \zeta_0\zeta_2)}. \quad (21)$$

Lemma 3.4 shows that γ_t is also the solution to the constrained line search problem. The proof of Lemma 3.4 is

based on the fact that we initialized the procedure with the vector most aligned with ℓ (and so ℓ_{n_t} must lie on the "other side" of ℓ from $\ell(w_t)$, i.e. $\gamma_t \leq 1$), along with the fact that the optimal objective in Eq. (18) is guaranteed to be positive (so moving towards ℓ_{n_t} is guaranteed to improve the objective, i.e. $\gamma_t \geq 0$).

Lemma 3.4. *For all $t \in \mathbb{N}$, $\gamma_t \in [0, 1]$.*

Proof. It is sufficient to show that both $\zeta_0 - \zeta_1\zeta_2$ and $\zeta_1 - \zeta_0\zeta_2$ are nonnegative and that at least one is strictly positive. First, examining Eq. (18), note that for any $\xi \in \Delta^{N-1}$,

$$\langle d_t, d_{tn_t} \rangle = \max_{n \in [N]} \langle d_t, d_{tn} \rangle \geq \sum_{n=1}^{N} \xi_n \langle d_t, d_{tn} \rangle, \quad (22)$$

so choosing $\xi_n = C \|\mathcal{L}_n\| \|\ell_n - \langle \ell_n, \ell(w_t) \rangle \ell(w_t)\|$ where $C > 0$ is chosen appropriately to normalize ξ, we have that

$$\langle d_t, d_{tn_t} \rangle \geq C \|\mathcal{L}\| \|\ell - \langle \ell, \ell(w_t) \rangle \ell(w_t)\| > 0, \quad (23)$$

since $\|\mathcal{L}\| > 0$ by assumption (see Section 3.1), and $\ell \neq \ell(w_t)$ since otherwise we could terminate the optimization after iteration $t - 1$. By expanding the formulae for d_t and d_{tn_t}, we have that for some $C' > 0$,

$$\zeta_0 - \zeta_1\zeta_2 = C' \langle d_t, d_{tn_t} \rangle > 0. \quad (24)$$

For the other term, first note that $\langle \ell, \ell(w_t) \rangle$ is a monotonically increasing function in t since $\gamma = 0$ is feasible in Eq. (19). Combined with the choice of initialization for w_1, we have that $\zeta_1 \geq \zeta_0$. Further, since $\zeta_2 \in [-1, 1]$ and $\zeta_1 \geq 0$ by its initialization, Eq. (24) implies $\zeta_1 \geq \zeta_1(-\zeta_2) \geq -\zeta_0$. Therefore $\zeta_1 \geq |\zeta_0| \geq |\zeta_0\zeta_2| \geq \zeta_0\zeta_2$, and the proof is complete. □

Given the line search step size γ_t, we can now update the weights to compute $\ell(w_{t+1})$ and w_{t+1}:

$$\ell(w_{t+1}) = \frac{\ell(w_t) + \gamma_t(\ell_{n_t} - \ell(w_t))}{\|\ell(w_t) + \gamma_t(\ell_{n_t} - \ell(w_t))\|} \quad (25)$$
$$w_{t+1} = \frac{w_t + \gamma_t(1_{n_t} - w_t)}{\|\ell(w_t) + \gamma_t(\ell_{n_t} - \ell(w_t))\|}. \quad (26)$$

There are two options for practical implementation: either (A) we keep track of only w_t, recomputing and normalizing $\ell(w_t)$ at each iteration to obtain w_{t+1}; or (B) we keep track of and store both w_t and $\ell(w_t)$. In theory both are equivalent, but in practice (B) will cause numerical errors to accumulate, making $\ell(w_t)$ not equal to the sum of normalized ℓ_n weighted by w_{tn}. However, in terms of vector sum and inner-product operations, (A) incurs an additional cost of $O(t)$ at each iteration t—resulting in a total of $O(M^2)$ for M iterations—while (B) has constant cost for each iteration with a total cost of $O(M)$. We found empirically that option (B) is preferable in practice, since the reduced numerical error of (A) does not justify the $O(M^2)$ cost.

3.4. Output

After running M iterations, the algorithm must output a set of weights that correspond to unnormalized vectors $\mathcal{L}_n$ rather than the normalized counterparts ℓ_n used within the algorithm. In addition, weights need to be rescaled optimally by $\alpha^\star$ from Eq. (10). The following formula adjusts the weights w_M accordingly to generate the output:

$$\forall n \in [N], \quad w_{Mn} \leftarrow w_{Mn} \frac{\|\mathcal{L}\|}{\|\mathcal{L}_n\|} \langle \ell(w_M), \ell \rangle. \quad (27)$$

3.5. Convergence Guarantee

We now prove Theorem 3.1 by bounding the objective of Eq. (11) as a function of the number of iterations M of greedy geodesic ascent. Writing $J_t := 1 - \langle \ell(w_t), \ell \rangle^2$ for brevity, we have that $\|\mathcal{L}\|^2 J_t = \|\alpha^\star \mathcal{L}(w_t) - \mathcal{L}\|^2$ for $\alpha^\star$ from Eq. (10), so providing an upper bound on J_t provides a relative bound on the coreset error. Substituting the optimal line search value γ_t into the objective of Eq. (19) yields

$$J_{t+1} = J_t \left(1 - \langle d_t, d_{tn_t} \rangle^2 \right), \quad (28)$$

where d_t, d_{tn}, and n_t are given in Eqs. (17) and (18). In other words, the cost J_t decreases corresponding to how well aligned the $\ell(w_t)$-to-ℓ and $\ell(w_t)$-to-ℓ_{n_t} geodesics are. We define for each $n \in [N]$ the geodesic direction from ℓ to ℓ_n to be $d_{\infty n} := \frac{\ell_n - \langle \ell_n, \ell \rangle \ell}{\|\ell_n - \langle \ell_n, \ell \rangle \ell\|}$; this slight abuse of notation from Eq. (17) is justified since we expect $\ell(w_t) \to \ell$ as $t \to \infty$. Then the worst-case alignment is governed by two constants, τ and ϵ:

$$\tau^{-1} := \min_{s \in \mathbb{R}^N} \|s\|_1 \quad \text{s.t.} \quad \ell = \sum_{m=1}^N s_m \ell_m \quad s \geq 0 \quad (29)$$

$$\epsilon := \min_{s \in \mathbb{R}^N} \max_{n \in [N]} \left\langle -\sum_m s_m d_{\infty m}, d_{\infty n} \right\rangle \quad (30)$$

$$\text{s.t.} \quad \|\sum_m s_m d_{\infty m}\| = 1, \quad s \geq 0.$$

Both are guaranteed to be positive by Lemma 3.5 below. The constant τ captures how fundamentally hard it is to approximate ℓ using $(\ell_n)_{n=1}^N$, and governs the worst-case behavior of the large steps taken in early iterations. In contrast, ϵ captures the worst-case behavior of the smaller steps in the $t \to \infty$ asymptotic regime when $\ell(w_t) \approx \ell$. These notions are quantified precisely in Lemma 3.6. The proofs of both Lemmas 3.5 and 3.6 may be found in Appendix B.

Lemma 3.5. *The constants τ and ϵ satisfy*

$$\tau \geq \frac{\|\mathcal{L}\|}{\sigma} > 0 \qquad and \qquad \epsilon > 0. \quad (31)$$

Lemma 3.6. *The geodesic alignment $\langle d_t, d_{tn_t} \rangle$ satisfies*

$$\langle d_t, d_{tn_t} \rangle \geq \tau \sqrt{J_t} \vee f(J_t), \quad (32)$$

where $f : [0, 1] \to \mathbb{R}$ is defined by

$$f(x) := \frac{\sqrt{1-x}\sqrt{1-\beta^2}\epsilon + \sqrt{x}\beta}{\sqrt{1 - \left(\sqrt{x}\sqrt{1-\beta^2}\epsilon - \sqrt{1-x}\beta\right)^2}} \quad (33)$$

$$\beta := 0 \wedge \left(\min_{n \in [N]} \langle \ell_n, \ell \rangle \ s.t. \ \langle \ell_n, \ell \rangle > -1 \right). \quad (34)$$

The proof of Theorem 3.1 below follows by using the $\tau \sqrt{J_t}$ bound from Lemma 3.6 to obtain an $O(1/t)$ bound on J_t, and then combining that result with the $f(J_t)$ bound from Lemma 3.6 to obtain the desired geometric decay.

Proof of Theorem 3.1. Substituting the $\tau \sqrt{J_t}$ bound from Lemma 3.6 into Eq. (28) and applying a standard inductive argument (e.g. the proof of Campbell & Broderick (2017, Lemma A.6)) yields

$$J_t \leq B(t) := \frac{J_1}{1 + \tau^2 J_1 (t-1)}. \quad (35)$$

Since $B(t) \to 0$ and $f(B(t)) \to \epsilon > 0$ as $t \to \infty$, there exists a $t^\star \in \mathbb{N}$ for which $t > t^\star$ implies $f(B(t)) \geq \tau \sqrt{B(t)}$. Furthermore, since $f(x)$ is monotonically decreasing, we have that $f(J_t) \geq f(B(t))$. Using the bound $\langle d_t, d_{tn_t} \rangle \geq f(B(t))$ in Eq. (28) and combining with Eq. (35) yields

$$J_t \leq B(t \wedge t^\star) \prod_{s=t^\star+1}^t \left(1 - f^2(B(s)) \right). \quad (36)$$

Multiplying by $\|\mathcal{L}\|^2$, taking the square root, and noting that η from Eq. (8) is equal to $\sqrt{J_1}$ gives the final result. The convergence of $f(B(t)) \to \epsilon$ as $t \to \infty$ shows that the rate of decay in the theorem statement is $\nu = \sqrt{1 - \epsilon^2}$. $\square$

4. Experiments

In this section we evaluate the performance of GIGA coreset construction compared with both uniformly random sub-sampling and the Frank–Wolfe-based method of Campbell & Broderick (2017). We first test the algorithms on simple synthetic examples, and then test them on logistic and Poisson regression models applied to numerous real and synthetic datasets. Code for these experiments is available at https://github.com/trevorcampbell/bayesian-coresets.

4.1. Synthetic Gaussian Inference

To generate Fig. 1, we generated $\mu \sim \mathcal{N}(0, 1)$ followed by $(y_n)_{n=1}^{10} \sim \mathcal{N}(\mu, 1)$. We then constructed Bayesian coresets via FW and GIGA using the norm specified in (Campbell & Broderick, 2017, Section 6). Across 1000 replications of this experiment, the median relative error in posterior variance approximation is 3% for GIGA and 48% for FW.

4.2. Synthetic Vector Sum Approximation

In this experiment, we generated 20 independent datasets consisting of 10^6 50-dimensional vectors from the multivariate normal distribution with mean 0 and identity covariance. We then constructed coresets for each of the datasets via uniformly random subsampling (RND), Frank–Wolfe (FW), and GIGA. We compared the algorithms on two metrics: reconstruction error, as measured by the 2-norm between $\mathcal{L}(w)$ and $\mathcal{L}$; and representation efficiency, as measured by the size of the coreset.

Fig. 3 shows the results of the experiment, with reconstruction error in Fig. 3a and coreset size in Fig. 3b. Across all construction iterations, GIGA provides a 2–4 order-of-magnitude reduction in error as compared with FW, and significantly outperforms RND. The exponential convergence of GIGA is evident. The flat section of FW/GIGA in Fig. 3a for iterations beyond 10^2 is due to the algorithm reaching the limits of numerical precision. In addition, Fig. 3b shows that GIGA can improve representational efficiency over FW, ceasing to grow the coreset once it reaches a size of 120, while FW continues to add points until it is over twice as large. Note that this experiment is designed to highlight the strengths of GIGA: by the law of large numbers, as $N \to \infty$ the sum $\mathcal{L}$ of the N i.i.d. standard multivariate normal data vectors satisfies $\|\mathcal{L}\| \to 0$, while the sum of their norms $\sigma \to \infty$ a.s. But Appendix A.1 shows that even in pathological cases, GIGA outperforms FW due to its optimal log-likelihood scaling.

4.3. Bayesian Posterior Approximation

In this experiment, we used GIGA to generate Bayesian coresets for logistic and Poisson regression. For both models, we used the standard multivariate normal prior $\theta \sim \mathcal{N}(0, I)$. In the **logistic regression** setting, we have a set of data points $(x_n, y_n)_{n=1}^N$ each consisting of a feature $x_n \in \mathbb{R}^D$ and a label $y_n \in \{-1, 1\}$, and the goal is to predict the label of a new point given its feature. We thus seek to infer the posterior distribution of the parameter $\theta \in \mathbb{R}^{D+1}$ governing the generation of y_n given x_n via

$$y_n \,|\, x_n, \theta \overset{\text{indep}}{\sim} \text{Bern}\left(\frac{1}{1 + e^{-z_n^T \theta}}\right) \quad z_n := \begin{bmatrix} x_n \\ 1 \end{bmatrix}. \quad (37)$$

In the **Poisson regression** setting, we have a set of data points $(x_n, y_n)_{n=1}^N$ each consisting of a feature $x_n \in \mathbb{R}^D$ and a count $y_n \in \mathbb{N}$, and the goal is to learn a relationship between features x_n and the associated mean count. We thus seek to infer the posterior distribution of the parameter $\theta \in \mathbb{R}^{D+1}$ governing the generation of y_n given x_n via

$$y_n \,|\, x_n, \theta \overset{\text{indep}}{\sim} \text{Poiss}\left(\log\left(1 + e^{z_n^T \theta}\right)\right) \quad z_n := \begin{bmatrix} x_n \\ 1 \end{bmatrix}. \quad (38)$$

We tested the coreset construction methods for each model

on a number of datasets; see Appendix D for references. For **logistic regression**, the Synthetic dataset consisted of $N = 10,000$ data points with covariate $x_n \in \mathbb{R}^2$ sampled i.i.d. from $\mathcal{N}(0, I)$, and label $y_n \in \{-1, 1\}$ generated from the logistic likelihood with parameter $\theta = [3, 3, 0]^T$. The Phishing dataset consisted of $N = 11,055$ data points each with $D = 68$ features. The DS1 dataset consisted of $N = 26,733$ data points each with $D = 10$ features.

For **Poisson regression**, the Synthetic dataset consisted of $N = 10,000$ data points with covariate $x_n \in \mathbb{R}$ sampled i.i.d. from $\mathcal{N}(0, 1)$, and count $y_n \in \mathbb{N}$ generated from the Poisson likelihood with $\theta = [1, 0]^T$. The BikeTrips dataset consisted of $N = 17,386$ data points each with $D = 8$ features. The AirportDelays dataset consisted of $N = 7,580$ data points each with $D = 15$ features.

We constructed coresets for each of the datasets via uniformly random subsampling (RND), Frank–Wolfe (FW), and GIGA using the weighted Fisher information distance,

$$\|\mathcal{L}_n\|^2 := \mathbb{E}_{\hat{\pi}}\left[\|\nabla \mathcal{L}_n(\theta)\|_2^2\right], \quad (39)$$

where $\mathcal{L}_n(\theta)$ is the log-likelihood for data point n, and $\hat{\pi}$ is obtained via the Laplace approximation. In order to do so, we approximated all vectors $\mathcal{L}_n$ using a 500-dimensional random feature projection (following Campbell & Broderick, 2017). For posterior inference, we used Hamiltonian Monte Carlo (Neal, 2011) with 15 leapfrog steps per sample. We simulated a total of 6,000 steps, with 1,000 warmup steps for step size adaptation with a target acceptance rate of 0.8, and 5,000 posterior sampling steps. Appendix A shows results for similar experiments using random-walk Metropolis–Hastings and the No-U-Turn Sampler (Hoffman & Gelman, 2014). We ran 20 trials of projection / coreset construction / MCMC for each combination of dataset, model, and algorithm. We evaluated the coresets at logarithmically-spaced construction iterations between $M = 1$ and 1000 by their median posterior Fisher information distance, estimated using samples obtained by running posterior inference on the full dataset in each trial. We compared these results versus computation time as well as coreset size. The latter serves as an implementation-independent measure of total cost, since the cost of running MCMC is much greater than running coreset construction and depends linearly on coreset size.

The results of this experiment are shown in Fig. 4. In Fig. 4a and Fig. 4b, the Fisher information distance is normalized by the median distance of RND for comparison. In Fig. 4b, the computation time is normalized by the median time to run MCMC on the full dataset. The suboptimal coreset log-likelihood scaling of FW can be seen in Fig. 4a for small coresets, resulting in similar performance to RND. In contrast, GIGA correctly scales posterior uncertainty across all coreset sizes, resulting in a major (3–4 orders of magnitude) reduction in error. Fig. 4b shows the same
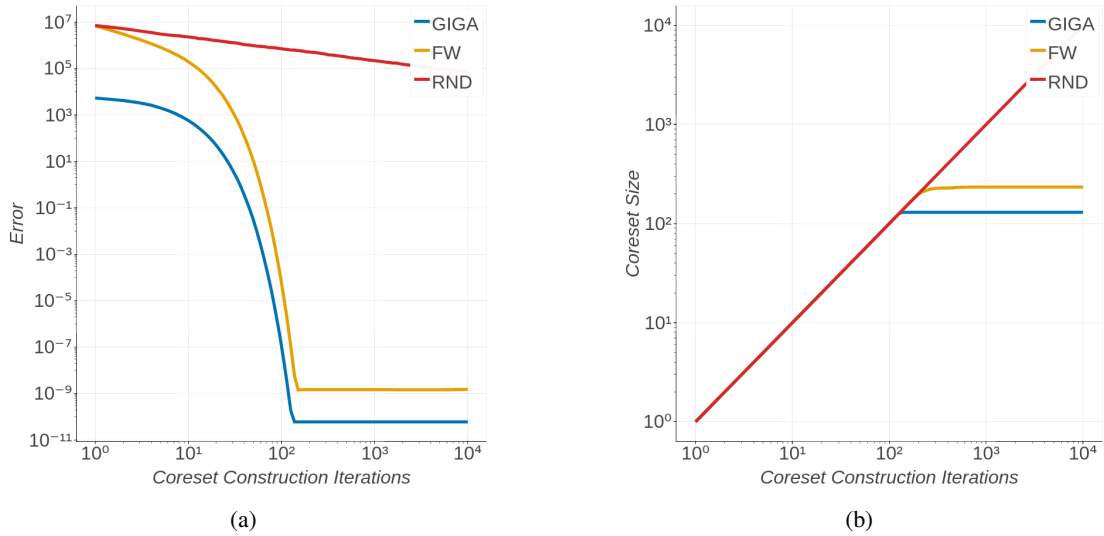
(a)

(b)

Figure 3. Comparison of different coreset constructions on the synthetic $\mathbb{R}^{50}$ vector dataset. Fig. 3a shows a comparison of 2-norm error between the coreset $\mathcal{L}(w)$ and the true sum $\mathcal{L}$ as a function of construction iterations, demonstrating the significant improvement in quality when using GIGA. Fig. 3b shows a similar comparison of coreset size, showing that GIGA produces smaller coresets for the same computational effort. All plots show the median curve over 20 independent trials.
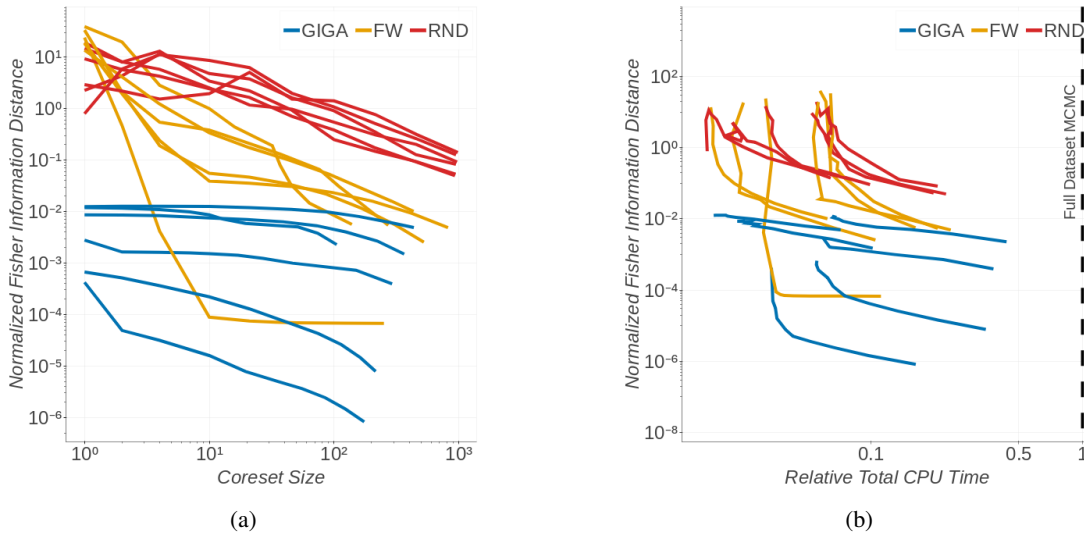

(a)

(b)

Figure 4. Comparison of the median Fisher information distance to the true posterior for GIGA, FW, and RND on the logistic and Poisson regression models over 20 random trials. Distances are normalized by the median value of RND for comparison. In Fig. 4b, computation time is normalized by the median value required to run MCMC on the full dataset. GIGA consistently outperforms FW and RND.

results plotted versus total computation time. This confirms that across a variety of models and datasets, GIGA provides significant improvements in posterior error over the state of the art.

5. Conclusion

This paper presented *greedy iterative geodesic ascent (GIGA)*, a novel Bayesian coreset construction algorithm.

Like previous algorithms, GIGA is simple to implement, has low computational cost, and has no tuning parameters. But in contrast to past work, GIGA scales the coreset log-likelihood optimally, providing significant improvements in the quality of posterior approximation. The theoretical guarantees and experimental results presented in this work reflect this improvement.

Acknowledgments

References

Angelino, Elaine, Johnson, Matthew, and Adams, Ryan. Patterns of scalable Bayesian inference. *Foundations and Trends in Machine Learning*, 9(1–2):1–129, 2016.

Barron, Andrew, Cohen, Albert, Dahmen, Wolfgang, and DeVore, Ronald. Approximation and learning by greedy algorithms. *The Annals of Statistics*, 36(1):64–94, 2008.

Boche, Holger, Calderbank, Robert, Kutyniok, Gitta, and Vybíral, Jan. A survey of compressed sensing. In Boche, Holger, Calderbank, Robert, Kutyniok, Gitta, and Vybíral, Jan (eds.), *Compressed Sensing and its Applications: MATHEON Workshop 2013*. Birkhäuser, 2015.

Broderick, Tamara, Boyd, Nicholas, Wibisono, Andre, Wilson, Ashia, and Jordan, Michael. Streaming variational Bayes. In *Advances in Neural Information Processing Systems*, 2013.

Campbell, Trevor and Broderick, Tamara. Automated scalable Bayesian inference via Hilbert coresets. *arXiv:1710.05053*, 2017.

Campbell, Trevor and How, Jonathan. Approximate decentralized Bayesian inference. In *Uncertainty in Artificial Intelligence*, 2014.

Campbell, Trevor, Straub, Julian, Fisher III, John W., and How, Jonathan. Streaming, distributed variational inference for Bayesian nonparametrics. In *Advances in Neural Information Processing Systems*, 2015.

Candès, Emmanual and Tao, Terence. The Dantzig selector: statistical estimation when p is much larger than n. *The Annals of Statistics*, 35(6):2313–2351, 2007.

Candès, Emmanuel and Tao, Terence. Decoding by linear programming. *IEEE Transactions on Information Theory*, 51(12): 4203–4215, 2005.

Chen, Scott, Donoho, David, and Saunders, Michael. Atomic decomposition by basis pursuit. *SIAM Review*, 43(1):129–159, 1999.

Chen, Sheng, Billings, Stephen, and Luo, Wan. Orthogonal least squares methods and their application to non-linear system identification. *International Journal of Control*, 50(5):1873–1896, 1989.

Chen, Yutian, Welling, Max, and Smola, Alex. Super-samples from kernel herding. In *Uncertainty in Artificial Intelligence*, 2010.

Dieng, Adji, Tran, Dustin, Rangananth, Rajesh, Paisley, John, and Blei, David. Variational inference via χ upper bound minimization. In *Advances in Neural Information Processing Systems*, 2017.

Donoho, David. Compressed sensing. *IEEE Transactions on Information Theory*, 52(4):1289–1306, 2006.

Frank, Marguerite and Wolfe, Philip. An algorithm for quadratic programming. *Naval Research Logistics Quarterly*, 3:95–110, 1956.

Freund, Yoav and Schapire, Robert. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55:119–139, 1997.

Guélat, Jacques and Marcotte, Patrice. Some comments on Wolfe's 'away step'. *Mathematical Programming*, 35:110–119, 1986.

Hoffman, Matthew and Gelman, Andrew. The No-U-Turn Sampler: adaptively setting path lengths in Hamiltonian Monte Carlo. *Journal of Machine Learning Research*, 15:1351–1381, 2014.

Hoffman, Matthew, Blei, David, Wang, Chong, and Paisley, John. Stochastic variational inference. *Journal of Machine Learning Research*, 14:1303–1347, 2013.

Huggins, Jonathan, Campbell, Trevor, and Broderick, Tamara. Coresets for Bayesian logistic regression. In *Advances in Neural Information Processing Systems*, 2016.

Jaggi, Martin. Revisiting Frank-Wolfe: projection-free sparse convex optimization. In *International Conference on Machine Learning*, 2013.

Jordan, Michael, Ghahramani, Zoubin, Jaakkola, Tommi, and Saul, Lawrence. An introduction to variational methods for graphical models. *Machine Learning*, 37:183–233, 1999.

Lacoste-Julien, Simon and Jaggi, Martin. On the global linear convergence of Frank-Wolfe optimization variants. In *Advances in Neural Information Processing Systems*, 2015.

Locatello, Francesco, Tschannen, Michael, Rätsch, Gunnar, and Jaggi, Martin. Greedy algorithms for cone constrained optimization with convergence guarantees. In *Advances in Neural Information Processing Systems*, 2017.

Mallat, Stéphane and Zhang, Zhifeng. Matching pursuits with time-frequency dictionaries. *IEEE Transactions on Signal Processing*, 41(12):3397–3415, 1993.

Neal, Radford. MCMC using Hamiltonian dynamics. In Brooks, Steve, Gelman, Andrew, Jones, Galin, and Meng, Xiao-Li (eds.), *Handbook of Markov chain Monte Carlo*, chapter 5. CRC Press, 2011.

Ranganath, Rajesh, Gerrish, Sean, and Blei, David. Black box variational inference. In *International Conference on Artificial Intelligence and Statistics*, 2014.

Robert, Christian and Casella, George. *Monte Carlo Statistical Methods*. Springer, 2nd edition, 2004.

Tibshirani, Robert. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society Series B*, 58(1): 267–288, 1996.

Tropp, Joel. Greed is good: algorithmic results for sparse approximation. *IEEE Transactions on Information Theory*, 50(10): 2231–2242, 2004.

Wainwright, Martin and Jordan, Michael. Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, 1(1–2):1–305, 2008.