# Variance Reduction and Quasi-Newton for Particle-Based Variational Inference

Michael H. Zhu[1] Chang Liu[2] Jun Zhu[3]

[1]Stanford University

[2]Microsoft Research

[3]Tsinghua University

ICML 2020

# Bayesian inference

- A central problem in Bayesian inference is approximating an intractable posterior distribution $p$ and estimating intractable expectations $\mathbb{E}_p\left[f(X)\right] = \int f(x)p(x)\,\mathrm{d}x$ with respect to $p$.

# Bayesian inference

- A central problem in Bayesian inference is approximating an intractable posterior distribution $p$ and estimating intractable expectations $\mathbb{E}_p[f(X)] = \int f(x)p(x)\,\mathrm{d}x$ with respect to $p$.
- Bayesian inference uses an entire distribution over the parameters for estimation.

# Bayesian inference

- A central problem in Bayesian inference is approximating an intractable posterior distribution $p$ and estimating intractable expectations $\mathbb{E}_p\left[f(X)\right] = \int f(x)p(x)\,\mathrm{d}x$ with respect to $p$.
- Bayesian inference uses an entire distribution over the parameters for estimation.
- For example, we can compute:

# Bayesian inference

- A central problem in Bayesian inference is approximating an intractable posterior distribution $p$ and estimating intractable expectations $\mathbb{E}_p\left[f(X)\right] = \int f(x)p(x)\,\mathrm{d}x$ with respect to $p$.
- Bayesian inference uses an entire distribution over the parameters for estimation.
- For example, we can compute:
  - posterior mean and covariance

# Bayesian inference

- A central problem in Bayesian inference is approximating an intractable posterior distribution $p$ and estimating intractable expectations $\mathbb{E}_p\left[f(X)\right] = \int f(x)p(x)\,\mathrm{d}x$ with respect to $p$.
- Bayesian inference uses an entire distribution over the parameters for estimation.
- For example, we can compute:
  - posterior mean and covariance
  - quantiles

# Bayesian inference

- A central problem in Bayesian inference is approximating an intractable posterior distribution $p$ and estimating intractable expectations $\mathbb{E}_p\left[f(X)\right] = \int f(x)p(x)\,\mathrm{d}x$ with respect to $p$.
- Bayesian inference uses an entire distribution over the parameters for estimation.
- For example, we can compute:
  - posterior mean and covariance
  - quantiles
  - marginal distributions

# Bayesian inference

- A central problem in Bayesian inference is approximating an intractable posterior distribution $p$ and estimating intractable expectations $\mathbb{E}_p\left[f(X)\right] = \int f(x)p(x)\,\mathrm{d}x$ with respect to $p$.
- Bayesian inference uses an entire distribution over the parameters for estimation.
- For example, we can compute:
    - posterior mean and covariance
    - quantiles
    - marginal distributions
- However, Bayesian inference is generally more computationally challenging.

# Bayesian inference

- A central problem in Bayesian inference is approximating an intractable posterior distribution $p$ and estimating intractable expectations $\mathbb{E}_p\left[f(X)\right] = \int f(x)p(x)\,\mathrm{d}x$ with respect to $p$.
- Bayesian inference uses an entire distribution over the parameters for estimation.
- For example, we can compute:
    - posterior mean and covariance
    - quantiles
    - marginal distributions
- However, Bayesian inference is generally more computationally challenging.
- The goal is to do inference accurately and efficiently.

# MCMC and VI

- Markov Chain Monte Carlo (MCMC) methods are a large class of sampling-based algorithms.

# MCMC and VI

- Markov Chain Monte Carlo (MCMC) methods are a large class of sampling-based algorithms.
  - samples $x_1, x_2, \ldots$ represent $p$

# MCMC and VI

- Markov Chain Monte Carlo (MCMC) methods are a large class of sampling-based algorithms.
  - samples $x_1, x_2, \ldots$ represent $p$
  - the sample average $\frac{1}{M}\sum_{i=1}^{M} f(x_i)$ is an asymptotically exact estimator of $\mathbb{E}_p\left[f(X)\right]$ as $M \to \infty$.

# MCMC and VI

- Markov Chain Monte Carlo (MCMC) methods are a large class of sampling-based algorithms.
    - samples $x_1, x_2, \ldots$ represent $p$
    - the sample average $\frac{1}{M}\sum_{i=1}^{M} f(x_i)$ is an asymptotically exact estimator of $\mathbb{E}_p\left[f(X)\right]$ as $M \to \infty$.
- Variational inference (VI) methods recast the inference problem as a parametric optimization problem.

# MCMC and VI

- Markov Chain Monte Carlo (MCMC) methods are a large class of sampling-based algorithms.
  - samples $x_1, x_2, \ldots$ represent $p$
  - the sample average $\frac{1}{M}\sum_{i=1}^{M} f(x_i)$ is an asymptotically exact estimator of $\mathbb{E}_p\left[f(X)\right]$ as $M \to \infty$.
- Variational inference (VI) methods recast the inference problem as a parametric optimization problem.
- MCMC methods are asymptotically exact but can be slow; VI methods can be fast but are generally biased.

# Particle-based Variational Inference

- Particle-based Variational Inference methods (ParVIs) are nonparametric variational inference methods that optimize a set of particles $\{x^{(1)}, x^{(2)}, \ldots, x^{(M)}\}$ to best represent $p$.

# Particle-based Variational Inference

- Particle-based Variational Inference methods (ParVIs) are nonparametric variational inference methods that optimize a set of particles $\{x^{(1)}, x^{(2)}, \ldots, x^{(M)}\}$ to best represent $p$.
- Stein Variational Gradient Descent (SVGD)

$$x_{k+1}^{(i)} \leftarrow x_k^{(i)} + \frac{\epsilon}{M} \sum_{j=1}^{M} \left( K(x_k^{(i)}, x_k^{(j)}) \nabla_{x_k^{(j)}} \log p(x_k^{(j)}) + \nabla_{x_k^{(j)}} K(x_k^{(i)}, x_k^{(j)}) \right)$$

# Particle-based Variational Inference

- Particle-based Variational Inference methods (ParVIs) are nonparametric variational inference methods that optimize a set of particles $\{x^{(1)}, x^{(2)}, \ldots, x^{(M)}\}$ to best represent $p$.
- Stein Variational Gradient Descent (SVGD)

$$x_{k+1}^{(i)} \leftarrow x_k^{(i)} + \frac{\epsilon}{M} \sum_{j=1}^{M} \left( K(x_k^{(i)}, x_k^{(j)}) \nabla_{x_k^{(j)}} \log p(x_k^{(j)}) + \nabla_{x_k^{(j)}} K(x_k^{(i)}, x_k^{(j)}) \right)$$

- Blob, GFSD, GFSF

# Particle-based Variational Inference

- Particle-based Variational Inference methods (ParVIs) are nonparametric variational inference methods that optimize a set of particles $\{x^{(1)}, x^{(2)}, \ldots, x^{(M)}\}$ to best represent $p$.
- Stein Variational Gradient Descent (SVGD)

$$x_{k+1}^{(i)} \leftarrow x_k^{(i)} + \frac{\epsilon}{M} \sum_{j=1}^{M} \left( K(x_k^{(i)}, x_k^{(j)}) \nabla_{x_k^{(j)}} \log p(x_k^{(j)}) + \nabla_{x_k^{(j)}} K(x_k^{(i)}, x_k^{(j)}) \right)$$

- Blob, GFSD, GFSF
- An important question for ParVIs is the quality of the posterior inference for a given posterior distribution $p$.

# Particle-based Variational Inference

- Particle-based Variational Inference methods (ParVIs) are nonparametric variational inference methods that optimize a set of particles $\{x^{(1)}, x^{(2)}, \ldots, x^{(M)}\}$ to best represent $p$.
- Stein Variational Gradient Descent (SVGD)

$$x_{k+1}^{(i)} \leftarrow x_k^{(i)} + \frac{\epsilon}{M} \sum_{j=1}^{M} \left( K(x_k^{(i)}, x_k^{(j)}) \nabla_{x_k^{(j)}} \log p(x_k^{(j)}) + \nabla_{x_k^{(j)}} K(x_k^{(i)}, x_k^{(j)}) \right)$$

- Blob, GFSD, GFSF
- An important question for ParVIs is the quality of the posterior inference for a given posterior distribution $p$.
  - How well do the particles $\{x^{(1)}, x^{(2)}, \ldots, x^{(M)}\}$ represent $p$ in practice?

# Particle-based Variational Inference

- Particle-based Variational Inference methods (ParVIs) are nonparametric variational inference methods that optimize a set of particles $\{x^{(1)}, x^{(2)}, \ldots, x^{(M)}\}$ to best represent $p$.
- Stein Variational Gradient Descent (SVGD)

$$x_{k+1}^{(i)} \leftarrow x_k^{(i)} + \frac{\epsilon}{M} \sum_{j=1}^{M} \left( K(x_k^{(i)}, x_k^{(j)}) \nabla_{x_k^{(j)}} \log p(x_k^{(j)}) + \nabla_{x_k^{(j)}} K(x_k^{(i)}, x_k^{(j)}) \right)$$

- Blob, GFSD, GFSF
- An important question for ParVIs is the quality of the posterior inference for a given posterior distribution $p$.
  - How well do the particles $\{x^{(1)}, x^{(2)}, \ldots, x^{(M)}\}$ represent $p$ in practice?
  - How accurate is the estimator $\frac{1}{M}\sum_{i=1}^{M} f(x^{(i)})$ for $\mathbb{E}_p\left[f(X)\right]$ in practice?

# Motivation

- For accurate posterior inference, highly accurate solutions to the ParVI optimization problem are needed.

# Motivation

- For accurate posterior inference, highly accurate solutions to the ParVI optimization problem are needed.
- We leverage ideas from large-scale optimization.

# Motivation

- For accurate posterior inference, highly accurate solutions to the ParVI optimization problem are needed.
- We leverage ideas from large-scale optimization.
- Stochastic gradient descent (SGD)

# Motivation

- For accurate posterior inference, highly accurate solutions to the ParVI optimization problem are needed.
- We leverage ideas from large-scale optimization.
- Stochastic gradient descent (SGD)
  - can reach an approximate solution relatively quickly

# Motivation

- For accurate posterior inference, highly accurate solutions to the ParVI optimization problem are needed.
- We leverage ideas from large-scale optimization.
- Stochastic gradient descent (SGD)
  - can reach an approximate solution relatively quickly
  - but has slow asymptotic convergence

# Motivation

- For accurate posterior inference, highly accurate solutions to the ParVI optimization problem are needed.
- We leverage ideas from large-scale optimization.
- Stochastic gradient descent (SGD)
  - can reach an approximate solution relatively quickly
  - but has slow asymptotic convergence
- Variance reduction methods for SGD, like SVRG

# Motivation

- For accurate posterior inference, highly accurate solutions to the ParVI optimization problem are needed.
- We leverage ideas from large-scale optimization.
- Stochastic gradient descent (SGD)
  - can reach an approximate solution relatively quickly
  - but has slow asymptotic convergence
- Variance reduction methods for SGD, like SVRG
  - accelerate convergence for strongly convex problems when highly accurate solutions are needed

# Motivation

- For accurate posterior inference, highly accurate solutions to the ParVI optimization problem are needed.
- We leverage ideas from large-scale optimization.
- Stochastic gradient descent (SGD)
  - can reach an approximate solution relatively quickly
  - but has slow asymptotic convergence
- Variance reduction methods for SGD, like SVRG
  - accelerate convergence for strongly convex problems when highly accurate solutions are needed
- Quasi-Newton methods, like L-BFGS

# Motivation

- For accurate posterior inference, highly accurate solutions to the ParVI optimization problem are needed.
- We leverage ideas from large-scale optimization.
- Stochastic gradient descent (SGD)
  - can reach an approximate solution relatively quickly
  - but has slow asymptotic convergence
- Variance reduction methods for SGD, like SVRG
  - accelerate convergence for strongly convex problems when highly accurate solutions are needed
- Quasi-Newton methods, like L-BFGS
  - speed up convergence for ill-conditioned problems by approximating the inverse Hessian

# Motivation

- For accurate posterior inference, highly accurate solutions to the ParVI optimization problem are needed.
- We leverage ideas from large-scale optimization.
- Stochastic gradient descent (SGD)
  - can reach an approximate solution relatively quickly
  - but has slow asymptotic convergence
- Variance reduction methods for SGD, like SVRG
  - accelerate convergence for strongly convex problems when highly accurate solutions are needed
- Quasi-Newton methods, like L-BFGS
  - speed up convergence for ill-conditioned problems by approximating the inverse Hessian
  - but traditionally are full-dataset methods

# Motivation

- For accurate posterior inference, highly accurate solutions to the ParVI optimization problem are needed.
- We leverage ideas from large-scale optimization.
- Stochastic gradient descent (SGD)
  - can reach an approximate solution relatively quickly
  - but has slow asymptotic convergence
- Variance reduction methods for SGD, like SVRG
  - accelerate convergence for strongly convex problems when highly accurate solutions are needed
- Quasi-Newton methods, like L-BFGS
  - speed up convergence for ill-conditioned problems by approximating the inverse Hessian
  - but traditionally are full-dataset methods
- Combining stochastic quasi-Newton methods and variance reduction

# Wasserstein optimization perspective of ParVIs

- However, ParVIs are not optimizing some function on the particle space, so directly applying the optimization techniques to the ParVI update rule of each particle is not technically sound.

# Wasserstein optimization perspective of ParVIs

- However, ParVIs are not optimizing some function on the particle space, so directly applying the optimization techniques to the ParVI update rule of each particle is not technically sound.
- Fortunately, ParVIs have been understood as minimizing the KL divergence $\mathrm{KL}_p(q) := \mathbb{E}_q[\log q/p]$ between the variational distribution $q$ and the target distribution $p$ on a general distribution space, the Wasserstein space.

# Wasserstein optimization perspective of ParVIs

- However, ParVIs are not optimizing some function on the particle space, so directly applying the optimization techniques to the ParVI update rule of each particle is not technically sound.
- Fortunately, ParVIs have been understood as minimizing the KL divergence $\mathrm{KL}_p(q) := \mathbb{E}_q[\log q/p]$ between the variational distribution $q$ and the target distribution $p$ on a general distribution space, the Wasserstein space.
- The Wasserstein space has a Riemannian structure, so we can leverage Riemannian optimization techniques.

# ParVIs on the Wasserstein space

- ParVIs can be formulated as optimizing $\mathrm{KL}_p(q)$ on the Wasserstein space by simulating the gradient flow of $\mathrm{KL}_p$.

# ParVIs on the Wasserstein space

- ParVIs can be formulated as optimizing $\mathrm{KL}_p(q)$ on the Wasserstein space by simulating the gradient flow of $\mathrm{KL}_p$.
- With the Riemannian structure of the Wasserstein space, the gradient can be expressed as:

$$\operatorname{grad}\mathrm{KL}_p(q) = \nabla \log(q/p).$$

# ParVIs on the Wasserstein space

- ParVIs can be formulated as optimizing $\mathrm{KL}_p(q)$ on the Wasserstein space by simulating the gradient flow of $\mathrm{KL}_p$.
- With the Riemannian structure of the Wasserstein space, the gradient can be expressed as:

$$\operatorname{grad} \mathrm{KL}_p(q) = \nabla \log(q/p).$$

- Let $\{x^{(i)}\}_{i=1}^M$ be a set of particles of $q$. The gradient flow simulation can be carried out by successively updating particles using a particle-based numerical approximation of $\operatorname{grad} \mathrm{KL}_p(q)$:

$$x_{k+1}^{(i)} \leftarrow x_k^{(i)} - \epsilon \hat{G}(\{x_k^{(j)}\}_j)^{(i)}$$

# SGD for ParVIs

- Let $p_0(x)$ be the prior and $p_n(x) := p(D_n|x)$ be the likelihood term for data point $D_n$. The KL divergence can be decomposed as the sum:

$$\mathrm{KL}_p(q) = \mathbb{E}_q[\log q] - \mathbb{E}_q[\log p_0] - \sum_{n=1}^{N} \mathbb{E}_q[\log p_n] + \log Z$$

# SGD for ParVIs

- Let $p_0(x)$ be the prior and $p_n(x) := p(D_n|x)$ be the likelihood term for data point $D_n$. The KL divergence can be decomposed as the sum:

$$\mathrm{KL}_p(q) = \mathbb{E}_q[\log q] - \mathbb{E}_q[\log p_0] - \sum_{n=1}^{N} \mathbb{E}_q[\log p_n] + \log Z$$

- The gradient over the full dataset is:

$$-\operatorname{grad} \mathrm{KL}_p(q) = \overbrace{\nabla \log p_0 - \nabla \log q}^{U(q)} + \overbrace{\sum_{n=1}^{N} \underbrace{\nabla \log p_n}_{V_n(q)}}^{V(q)}$$

# SGD for ParVIs

- Let $p_0(x)$ be the prior and $p_n(x) := p(D_n|x)$ be the likelihood term for data point $D_n$. The KL divergence can be decomposed as the sum:

$$\mathrm{KL}_p(q) = \mathbb{E}_q[\log q] - \mathbb{E}_q[\log p_0] - \sum_{n=1}^{N} \mathbb{E}_q[\log p_n] + \log Z$$

- The gradient over the full dataset is:

$$-\operatorname{grad}\mathrm{KL}_p(q) = \overbrace{\nabla \log p_0 - \nabla \log q}^{U(q)} + \overbrace{\sum_{n=1}^{N} \underbrace{\nabla \log p_n}_{V_n(q)}}^{V(q)}$$

- Let $\hat{U}(\{x^{(j)}\}_j)^{(i)}$ and $\hat{V}_n(\{x^{(j)}\}_j)^{(i)}$ be the particle-based numerical approximations of $U(q)$ and $V_n(q)$ respectively.

# SGD for ParVIs

- Let $p_0(x)$ be the prior and $p_n(x) := p(D_n|x)$ be the likelihood term for data point $D_n$. The KL divergence can be decomposed as the sum:

$$\mathrm{KL}_p(q) = \mathbb{E}_q[\log q] - \mathbb{E}_q[\log p_0] - \sum_{n=1}^{N} \mathbb{E}_q[\log p_n] + \log Z$$

- The gradient over the full dataset is:

$$-\operatorname{grad} \mathrm{KL}_p(q) = \overbrace{\nabla \log p_0 - \nabla \log q}^{U(q)} + \overbrace{\sum_{n=1}^{N} \underbrace{\nabla \log p_n}_{V_n(q)}}^{V(q)}$$

- Let $\hat{U}(\{x^{(j)}\}_j)^{(i)}$ and $\hat{V}_n(\{x^{(j)}\}_j)^{(i)}$ be the particle-based numerical approximations of $U(q)$ and $V_n(q)$ respectively.
- The SGD update step is first sample a data point $n_k \in \{1, \cdots, N\}$ uniformly at random and then update

$$x_{k+1}^{(i)} \leftarrow x_k^{(i)} + \varepsilon \left( \hat{U}(\{x_k^{(j)}\}_j)^{(i)} + N \hat{V}_{n_k}(\{x_k^{(j)}\}_j)^{(i)} \right)$$

# Riemannian variance reduction and quasi-Newton

- We want to derive variance reduction and quasi-Newton methods for ParVIs based on Riemannian variance reduction and quasi-Newton algorithms.

# Riemannian variance reduction and quasi-Newton

- We want to derive variance reduction and quasi-Newton methods for ParVIs based on Riemannian variance reduction and quasi-Newton algorithms.
- Riemannian variance reduction and quasi-Newton algorithms require geometric structures of the Riemannian manifold like (inverse) exponential map and parallel transport.

# Riemannian variance reduction and quasi-Newton

- We want to derive variance reduction and quasi-Newton methods for ParVIs based on Riemannian variance reduction and quasi-Newton algorithms.
- Riemannian variance reduction and quasi-Newton algorithms require geometric structures of the Riemannian manifold like (inverse) exponential map and parallel transport.
    - e.g., for transporting a cached direction at a snapshot position to the current position

# Riemannian variance reduction and quasi-Newton

- We want to derive variance reduction and quasi-Newton methods for ParVIs based on Riemannian variance reduction and quasi-Newton algorithms.
- Riemannian variance reduction and quasi-Newton algorithms require geometric structures of the Riemannian manifold like (inverse) exponential map and parallel transport.
    - e.g., for transporting a cached direction at a snapshot position to the current position
- We derive particle realizations of these operations under the pairwise-close approximation, which are stable and do not increase the order of computation cost.

# Riemannian variance reduction and quasi-Newton

- We want to derive variance reduction and quasi-Newton methods for ParVIs based on Riemannian variance reduction and quasi-Newton algorithms.
- Riemannian variance reduction and quasi-Newton algorithms require geometric structures of the Riemannian manifold like (inverse) exponential map and parallel transport.
  - e.g., for transporting a cached direction at a snapshot position to the current position
- We derive particle realizations of these operations under the pairwise-close approximation, which are stable and do not increase the order of computation cost.
- Note that the algorithms we present in this talk are simplified under the pairwise-close approximation.

# Variance Reduction for ParVIs

- Based on Riemannian SVRG, we can derive SVRG for ParVIs.

# Variance Reduction for ParVIs

- Based on Riemannian SVRG, we can derive SVRG for ParVIs.
- At the start of every outer loop,

# Variance Reduction for ParVIs

- Based on Riemannian SVRG, we can derive SVRG for ParVIs.
- At the start of every outer loop,
  - the positions of the particles are recorded as a reference snapshot position $\{\tilde{x}^{(i)}\}_{i=1}^{M}$, and

# Variance Reduction for ParVIs

- Based on Riemannian SVRG, we can derive SVRG for ParVIs.
- At the start of every outer loop,
  - the positions of the particles are recorded as a reference snapshot position $\{\tilde{x}^{(i)}\}_{i=1}^{M}$, and
  - the corresponding full-summation over the entire dataset is computed and stored: $\tilde{V}^{(i)} \leftarrow \sum_{n=1}^{N} \hat{V}_n(\{\tilde{x}^{(j)}\}_j)^{(i)}$.

# Variance Reduction for ParVIs

- Based on Riemannian SVRG, we can derive SVRG for ParVIs.
- At the start of every outer loop,
  - the positions of the particles are recorded as a reference snapshot position $\{\tilde{x}^{(i)}\}_{i=1}^{M}$, and
  - the corresponding full-summation over the entire dataset is computed and stored: $\tilde{V}^{(i)} \leftarrow \sum_{n=1}^{N} \hat{V}_n(\{\tilde{x}^{(j)}\}_j)^{(i)}$.
- In each subsequent iteration $k$, for a random data point $n_k$,

# Variance Reduction for ParVIs

- Based on Riemannian SVRG, we can derive SVRG for ParVIs.
- At the start of every outer loop,
  - the positions of the particles are recorded as a reference snapshot position $\{\tilde{x}^{(i)}\}_{i=1}^{M}$, and
  - the corresponding full-summation over the entire dataset is computed and stored: $\tilde{V}^{(i)} \leftarrow \sum_{n=1}^{N} \hat{V}_n(\{\tilde{x}^{(j)}\}_j)^{(i)}$.
- In each subsequent iteration $k$, for a random data point $n_k$,
  - the stochastic gradient at the current position $\{x_k^{(i)}\}_{i=1}^{M}$ is combined with the stochastic gradient at the snapshot position $\{\tilde{x}^{(i)}\}_{i=1}^{M}$ and the stored full gradient $\{\tilde{V}^{(i)}\}_{i=1}^{M}$ to get the variance-reduced gradient.

$$x_{k+1}^{(i)} \leftarrow x_k^{(i)} + \varepsilon \quad \Big( \hat{U}(\{x_k^{(j)}\}_j)^{(i)} + N\hat{V}_{n_k}(\{x_k^{(j)}\}_j)^{(i)} - \left( \left\{ N\hat{V}_{n_k}(\{\tilde{x}^{(j')}\}_{j'})^{(j)} - \tilde{V}^{(j)} \right\}_j \right)^{(i)} \Big)$$

# Stochastic Quasi-Newton with Variance Reduction (SQN-VR) for ParVIs

- In addition to variance reduction, we can further incorporate quasi-Newton preconditioning techniques based on Riemannian SQN-VR.

# Stochastic Quasi-Newton with Variance Reduction (SQN-VR) for ParVIs

- In addition to variance reduction, we can further incorporate quasi-Newton preconditioning techniques based on Riemannian SQN-VR.
- At the start of every outer loop,

# Stochastic Quasi-Newton with Variance Reduction (SQN-VR) for ParVIs

- In addition to variance reduction, we can further incorporate quasi-Newton preconditioning techniques based on Riemannian SQN-VR.
- At the start of every outer loop,
  - as in SVRG, the snapshot position $\{\tilde{x}^{(i)}\}_{i=1}^{M}$ and the corresponding full-summation $\{\tilde{V}^{(i)}\}_{i=1}^{M}$ are stored, and

# Stochastic Quasi-Newton with Variance Reduction (SQN-VR) for ParVIs

- In addition to variance reduction, we can further incorporate quasi-Newton preconditioning techniques based on Riemannian SQN-VR.
- At the start of every outer loop,
  - as in SVRG, the snapshot position $\{\tilde{x}^{(i)}\}_{i=1}^{M}$ and the corresponding full-summation $\{\tilde{V}^{(i)}\}_{i=1}^{M}$ are stored, and
  - the L-BFGS curvature pairs are updated, using the difference between the current and previous snapshot position and the difference between their corresponding full-summations.

# Stochastic Quasi-Newton with Variance Reduction (SQN-VR) for ParVIs

- In addition to variance reduction, we can further incorporate quasi-Newton preconditioning techniques based on Riemannian SQN-VR.
- At the start of every outer loop,
  - as in SVRG, the snapshot position $\{\tilde{x}^{(i)}\}_{i=1}^{M}$ and the corresponding full-summation $\{\tilde{V}^{(i)}\}_{i=1}^{M}$ are stored, and
  - the L-BFGS curvature pairs are updated, using the difference between the current and previous snapshot position and the difference between their corresponding full-summations.
- In each subsequent iteration $k$,

# Stochastic Quasi-Newton with Variance Reduction (SQN-VR) for ParVIs

- In addition to variance reduction, we can further incorporate quasi-Newton preconditioning techniques based on Riemannian SQN-VR.
- At the start of every outer loop,
    - as in SVRG, the snapshot position $\{\tilde{x}^{(i)}\}_{i=1}^{M}$ and the corresponding full-summation $\{\tilde{V}^{(i)}\}_{i=1}^{M}$ are stored, and
    - the L-BFGS curvature pairs are updated, using the difference between the current and previous snapshot position and the difference between their corresponding full-summations.
- In each subsequent iteration $k$,
    - first compute the variance-reduced gradient as in SVRG, and

# Stochastic Quasi-Newton with Variance Reduction (SQN-VR) for ParVIs

- In addition to variance reduction, we can further incorporate quasi-Newton preconditioning techniques based on Riemannian SQN-VR.
- At the start of every outer loop,
    - as in SVRG, the snapshot position $\{\tilde{x}^{(i)}\}_{i=1}^{M}$ and the corresponding full-summation $\{\tilde{V}^{(i)}\}_{i=1}^{M}$ are stored, and
    - the L-BFGS curvature pairs are updated, using the difference between the current and previous snapshot position and the difference between their corresponding full-summations.
- In each subsequent iteration $k$,
    - first compute the variance-reduced gradient as in SVRG, and
    - then apply a quasi-Newton update using the L-BFGS two-loop recursion with the variance reduced gradient.

# Experimental setup

- We run experiments on Bayesian linear regression and logistic regression datasets with a batch size of 10.

# Experimental setup

- We run experiments on Bayesian linear regression and logistic regression datasets with a batch size of 10.
- For the choice of ParVI, we use SVGD with 100 particles and the linear kernel.

# Experimental setup

- We run experiments on Bayesian linear regression and logistic regression datasets with a batch size of 10.
- For the choice of ParVI, we use SVGD with 100 particles and the linear kernel.
- We compare the following optimization algorithms: SGD, AdaGrad with momentum, SVRG, SPIDER, and SQN-VR.

# Experimental setup

- We run experiments on Bayesian linear regression and logistic regression datasets with a batch size of 10.
- For the choice of ParVI, we use SVGD with 100 particles and the linear kernel.
- We compare the following optimization algorithms: SGD, AdaGrad with momentum, SVRG, SPIDER, and SQN-VR.
- Evaluation:

# Experimental setup

- We run experiments on Bayesian linear regression and logistic regression datasets with a batch size of 10.
- For the choice of ParVI, we use SVGD with 100 particles and the linear kernel.
- We compare the following optimization algorithms: SGD, AdaGrad with momentum, SVRG, SPIDER, and SQN-VR.
- Evaluation:
  - We obtain a set of 40,000 ground-truth samples using MCMC.

# Experimental setup

- We run experiments on Bayesian linear regression and logistic regression datasets with a batch size of 10.
- For the choice of ParVI, we use SVGD with 100 particles and the linear kernel.
- We compare the following optimization algorithms: SGD, AdaGrad with momentum, SVRG, SPIDER, and SQN-VR.
- Evaluation:
  - We obtain a set of 40,000 ground-truth samples using MCMC.

  1. Maximum Mean Discrepancy (MMD) between the 100 ParVI particles and the 40,000 MCMC samples.

# Experimental setup

- We run experiments on Bayesian linear regression and logistic regression datasets with a batch size of 10.
- For the choice of ParVI, we use SVGD with 100 particles and the linear kernel.
- We compare the following optimization algorithms: SGD, AdaGrad with momentum, SVRG, SPIDER, and SQN-VR.
- Evaluation:
  - We obtain a set of 40,000 ground-truth samples using MCMC.

  1. Maximum Mean Discrepancy (MMD) between the 100 ParVI particles and the 40,000 MCMC samples.
  2. Mean-squared error for estimating posterior mean.

# Experimental setup

- We run experiments on Bayesian linear regression and logistic regression datasets with a batch size of 10.
- For the choice of ParVI, we use SVGD with 100 particles and the linear kernel.
- We compare the following optimization algorithms: SGD, AdaGrad with momentum, SVRG, SPIDER, and SQN-VR.
- Evaluation:
  - We obtain a set of 40,000 ground-truth samples using MCMC.

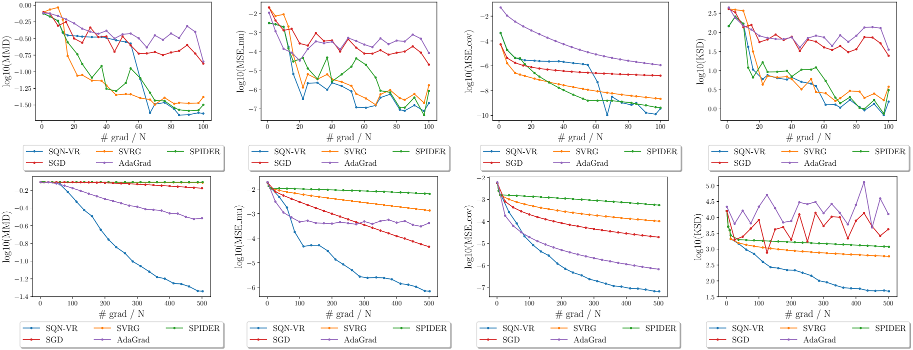
  1. Maximum Mean Discrepancy (MMD) between the 100 ParVI particles and the 40,000 MCMC samples.
  2. Mean-squared error for estimating posterior mean.
  3. Mean-squared error for estimating posterior covariance.

# Experimental setup

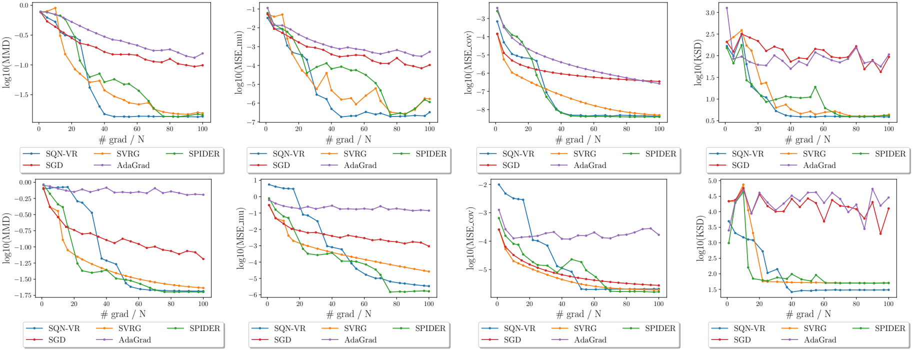
- We run experiments on Bayesian linear regression and logistic regression datasets with a batch size of 10.
- For the choice of ParVI, we use SVGD with 100 particles and the linear kernel.
- We compare the following optimization algorithms: SGD, AdaGrad with momentum, SVRG, SPIDER, and SQN-VR.
- Evaluation:
  - We obtain a set of 40,000 ground-truth samples using MCMC.

  1. Maximum Mean Discrepancy (MMD) between the 100 ParVI particles and the 40,000 MCMC samples.
  2. Mean-squared error for estimating posterior mean.
  3. Mean-squared error for estimating posterior covariance.
  4. Kernel Stein Discrepancy (KSD) for the 100 ParVI particles with respect to the posterior distribution.

# Experimental results for Bayesian linear regression



# Experimental results for Bayesian logistic regression



# Summary

- We propose a variance reduction and quasi-Newton preconditioning framework for ParVIs.

# Summary

- We propose a variance reduction and quasi-Newton preconditioning framework for ParVIs.
- Our methods are derived from the Wasserstein optimization perspective of ParVIs and Riemannian optimization algorithms.

# Summary

- We propose a variance reduction and quasi-Newton preconditioning framework for ParVIs.
- Our methods are derived from the Wasserstein optimization perspective of ParVIs and Riemannian optimization algorithms.
- Our experimental results on Bayesian linear regression and logistic regression show that our new methods result in a set of particles with significantly better sample quality.