Scalable probabilistic inference

David Dunson

Departments of Statistical Science, Mathematics & ECE, Duke University

September 21, 2016
Outline

Motivation

Hybrid Algorithms

EP-MCMC

aMCMC
Scalable inference with UQ

For massive datasets, it is necessarily to rely on distributed/parallel computing
Scalable inference with UQ

- For massive datasets, it is necessarily to rely on distributed/parallel computing
- Literature focuses on optimization algorithms for point estimation

Motivation
Scalable inference with UQ

- For massive datasets, it is necessarily to rely on distributed/parallel computing
- Literature focuses on optimization algorithms for point estimation
- Such approaches have seen great success in certain areas - high tech industry, web data, etc
Scalable inference with UQ

- For massive datasets, it is necessarily to rely on distributed/parallel computing
- Literature focuses on optimization algorithms for point estimation
- Such approaches have seen great success in certain areas - high tech industry, web data, etc
- But in many settings accurate uncertainty quantification (UQ) is critical
Scalable inference with UQ

- For massive datasets, it is necessarily to rely on distributed/parallel computing
- Literature focuses on optimization algorithms for point estimation
- Such approaches have seen great success in certain areas - high tech industry, web data, etc
- But in many settings accurate uncertainty quantification (UQ) is critical
- Need for scalable algorithms using distributed systems for efficient statistical inferences including UQ
Bayes in practice

- Big & high-dimensional data are now commonplace

Not just in industry!

My collaborations - neuroscience, genomics, ecology, arts, etc

Bayesian models are very well motivated in these applications

Provide substantial advantages over penalization methods

Substantial practical issue is computational time & stability
Bayes in practice

- Big & high-dimensional data are now commonplace
- Not just in industry!
Bayes in practice

- Big & high-dimensional data are now commonplace
- Not just in industry!
- My collaborations - neuroscience, genomics, ecology, arts, etc
Big & high-dimensional data are now commonplace

Not just in industry!

My collaborations - neuroscience, genomics, ecology, arts, etc

Bayesian models are very well motivated in these applications
Bayes in practice

- Big & high-dimensional data are now commonplace
- Not just in industry!
- My collaborations - neuroscience, genomics, ecology, arts, etc
- Bayesian models are very well motivated in these applications
- Provide substantial advantages over penalization methods
Bayes in practice

- Big & high-dimensional data are now commonplace
- Not just in industry!
- My collaborations - neuroscience, genomics, ecology, arts, etc
- Bayesian models are very well motivated in these applications
- Provide substantial advantages over penalization methods
- Substantial practical issue is computational time & stability
Bayes in high-dimensional problems

Models tend to be necessarily heavily parameterized
Bayes in high-dimensional problems

- Models tend to be necessarily heavily parameterized
- Posterior involves many parameters & hence marginal likelihood involves a high-dimensional integral
Bayes in high-dimensional problems

- Models tend to be necessarily heavily parameterized
- Posterior involves many parameters & hence marginal likelihood involves a high-dimensional integral
- MCMC & related sampling algorithms are routinely used to estimate posterior summaries
Bayes in high-dimensional problems

Models tend to be necessarily heavily parameterized.

Posterior involves many parameters & hence marginal likelihood involves a high-dimensional integral.

MCMC & related sampling algorithms are routinely used to estimate posterior summaries.

Multiple vexing computational bottlenecks arise.
Computational bottlenecks in MCMC

- Time per MCMC iteration increases with the number of parameters/unknowns
Computational bottlenecks in MCMC

- Time per MCMC iteration increases with the number of parameters/unknowns
- Often mixing rates get worse as the dimension of the data increases
Computational bottlenecks in MCMC

- Time per MCMC iteration increases with the number of parameters/unknowns.
- Often mixing rates get worse as the dimension of the data increases.
- True for large samples & high-dimensional low sample size data.
Computational bottlenecks in MCMC

- Time per MCMC iteration increases with the number of parameters/unknowns
- Often mixing rates get worse as the dimension of the data increases
- True for large samples & high-dimensional low sample size data
- Storing & doing even basic processing on big data sets is problematic
Computational bottlenecks in MCMC

- Time per MCMC iteration increases with the number of parameters/unknowns.
- Often mixing rates get worse as the dimension of the data increases.
- True for large samples & high-dimensional low sample size data.
- Storing & doing even basic processing on big data sets is problematic.
- Usually MCMC requires multiple likelihood and/or gradient evaluations at each iteration.
Due to the above issues, I routinely run into problems implementing “old school” MCMC algorithms in applications.
Due to the above issues, I routinely run into problems implementing “old school” MCMC algorithms in applications.

Not so easy to solve the bottlenecks even with a substantial toolbox of clever tricks.
Due to the above issues, I routinely run into problems implementing “old school” MCMC algorithms in applications. Not so easy to solve the bottlenecks even with a substantial toolbox of clever tricks. Many applied researchers have abandoned MCMC in favor of alternatives.
Due to the above issues, I routinely run into problems implementing “old school” MCMC algorithms in applications.

Not so easy to solve the bottlenecks even with a substantial toolbox of clever tricks.

Many applied researchers have abandoned MCMC in favor of alternatives.

There are many: (i) point estimation (e.g., MAP); (ii) Laplace; (iii) expectation-propagation; (iv) variational Bayes (VB); etc.
Due to the above issues, I routinely run into problems implementing “old school” MCMC algorithms in applications. Not so easy to solve the bottlenecks even with a substantial toolbox of clever tricks. Many applied researchers have abandoned MCMC in favor of alternatives. There are many: (i) point estimation (e.g., MAP); (ii) Laplace; (iii) expectation-propagation; (iv) variational Bayes (VB); etc.

**Issue:** alternatives lack general accuracy particularly in UQ.
It is appealing to develop simple, new modifications to MCMC.
It is appealing to develop simple, new modifications to MCMC.

Goal is to make computation faster and more robust, while maintaining simplicity, generality & theoretical guarantees.
It is appealing to develop simple, new modifications to MCMC.

Goal is to make computation faster and more robust, while maintaining simplicity, generality & theoretical guarantees.

We would like to accomplish this not just for large sample size problems.
Reviving MCMC

It is appealing to develop simple, new modifications to MCMC. Goal is to make computation faster and more robust, while maintaining simplicity, generality & theoretical guarantees.

We would like to accomplish this not just for large sample size problems.

Most practical problems I run into involve small to moderate samples but HUGE dimensional data.
Possible solutions

- **Hybrid algorithms**: run MCMC for a subset of the parameters & use a fast estimate for the others.
Possible solutions

- **Hybrid algorithms**: run MCMC for a subset of the parameters & use a fast estimate for the others.
- **Embarrassingly parallel (EP) MCMC**: run MCMC in parallel for different subsets of data & combine
Possible solutions

- **Hybrid algorithms**: run MCMC for a subset of the parameters & use a fast estimate for the others.
- **Embarrassingly parallel (EP) MCMC**: run MCMC in parallel for different subsets of data & combine
- **Approximate MCMC**: Approximate expensive to evaluate transition kernels
Outline

Motivation

Hybrid Algorithms

EP-MCMC

aMCMC
Hybrid algorithms

- MCMC is often intractable in high-dimensional data problems
Hybrid algorithms

- MCMC is often intractable in high-dimensional data problems
- Example 1: predict response variable $y_i$ from $x_i = (x_{i1}, \ldots, x_{ip})^T$ with $p \geq 1,000,000$
Hybrid algorithms

- MCMC is often intractable in high-dimensional data problems.
- Example 1: predict response variable $y_i$ from $x_i = (x_{i1}, \ldots, x_{ip})^T$ with $p \geq 1,000,000$.
- Example 2: estimate density of $y_i = (y_{i1}, \ldots, y_{ip})^T$ with $p \geq 1,000$. 

We can define parametric & nonparametric Bayes models but there are too many parameters to update & mixing is horrendous.

Solution: run MCMC only for some of the parameters.
Hybrid algorithms

- MCMC is often intractable in high-dimensional data problems
- Example 1: predict response variable $y_i$ from $x_i = (x_{i1}, \ldots, x_{ip})^T$ with $p \geq 1,000,000$
- Example 2: estimate density of $y_i = (y_{i1}, \ldots, y_{ip})^T$ with $p \geq 1,000$.
- We can define parametric & nonparametric Bayes models but there are too many parameters to update & mixing is horrendous.
Hybrid algorithms

- MCMC is often intractable in high-dimensional data problems
- Example 1: predict response variable $y_i$ from $x_i = (x_{i1}, \ldots, x_{ip})^T$ with $p \geq 1,000,000$
- Example 2: estimate density of $y_i = (y_{i1}, \ldots, y_{ip})^T$ with $p \geq 1,000$.
- We can define parametric & nonparametric Bayes models but there are too many parameters to update & mixing is horrendous.
- **Solution**: run MCMC only for some of the parameters
Flexible Bayesian models for high-dimensional data are very richly parameterized.
Motivation

- Flexible Bayesian models for high-dimensional data are very richly parameterized.
- Such models face issues with “weak identifiability.”
Motivation

- Flexible Bayesian models for high-dimensional data are very richly parameterized.
- Such models face issues with “weak identifiability”.
- Hard to tie down all the parameters based on the observed data.
Motivation

- Flexible Bayesian models for high-dimensional data are very richly parameterized
- Such models face issues with “weak identifiability”
- Hard to tie down all the parameters based on the observed data
- Address problem by fixing key parameters at point estimate
Flexible Bayesian models for high-dimensional data are very richly parameterized.

Such models face issues with “weak identifiability”.

Hard to tie down all the parameters based on the observed data.

Address problem by fixing key parameters at point estimate.

Solves weak identifiability & over-parameterization problem, leading to dramatic gains in MCMC mixing + time/iteration.
Motivation

- Flexible Bayesian models for high-dimensional data are very richly parameterized.
- Such models face issues with “weak identifiability”.
- Hard to tie down all the parameters based on the observed data.
- Address problem by fixing key parameters at point estimate.
- Solves weak identifiability & over-parameterization problem, leading to dramatic gains in MCMC mixing + time/iteration.
- Uncertainty quantification is often good.
Example 1: conditional density estimation

- Assume $x_i = (x_{i1}, \ldots, x_{ip})^T \in \mathcal{X} \subset \mathbb{R}^p$, with $x_i$s concentrated near $\mathcal{M}$ – lower dimensional subspace

Petralia, Vogelstein & Dunson (2013, NIPS)
Example 1: conditional density estimation

Petralia, Vogelstein & Dunson (2013, NIPS)

Assume \( x_i = (x_{i1}, \ldots, x_{ip})^T \in \mathcal{X} \subset \mathbb{R}^p \), with \( x_i \)'s concentrated near \( \mathcal{M} \) – lower dimensional subspace

Fast estimation of \( \mathcal{M} \) in a multiscale manner - \( z_i = \) binary sequence encoding location of \( x_i \)
Example 1: conditional density estimation

Petralia, Vogelstein & Dunson (2013, NIPS)

Assume \( x_i = (x_{i1}, \ldots, x_{ip})^T \in \mathcal{X} \subset \mathbb{R}^p \), with \( x_i \)'s concentrated near \( \mathcal{M} \) – lower dimensional subspace

Fast estimation of \( \mathcal{M} \) in a multiscale manner - \( z_i \) = binary sequence encoding location of \( x_i \)

Define a nonparametric Bayes model for \( f(y_i | x_i) = f(y_i | z_i) \) & run MCMC for unknowns in this model assuming \( z_i \) known
Example 1: conditional density estimation

Assume $x_i = (x_{i1}, \ldots, x_{ip})^T \in \mathcal{X} \subset \mathbb{R}^p$, with $x_i$'s concentrated near $\mathcal{M}$ – lower dimensional subspace

Fast estimation of $\mathcal{M}$ in a multiscale manner - $z_i$ = binary sequence encoding location of $x_i$

Define a nonparametric Bayes model for $f(y_i|x_i) = f(y_i|z_i)$ & run MCMC for unknowns in this model assuming $z_i$ known

Great performance in estimating $f(y|x)$ for $x$ varying across $\mathcal{M}$ & in prediction + computationally very efficient
Example 2: high-dimensional density estimation

\[ y_i = (y_{i1}, \ldots, y_{ip})^T \sim f \text{ with } p \text{ large & } f \text{ an unknown density} \]

Ye, Canale & Dunson (2016, AISTATS)
Example 2: high-dimensional density estimation

\[ y_i = (y_{i1}, \ldots, y_{ip})^T \sim f \text{ with } p \text{ large & } f \text{ an unknown density} \]

Potentially use Dirichlet process mixtures of factor models
Example 2: high-dimensional density estimation

Ye, Canale & Dunson (2016, AISTATS)

\[ y_i = (y_{i1}, \ldots, y_{ip})^T \sim f \] with \( p \) large & \( f \) an unknown density

- Potentially use Dirichlet process mixtures of factor models
- Approach doesn’t scale well at all with \( p \)
Example 2: high-dimensional density estimation

\[ y_i = (y_{i1}, \ldots, y_{ip})^T \sim f \] with \( p \) large & \( f \) an unknown density

- Potentially use Dirichlet process mixtures of factor models
- Approach doesn’t scale well at all with \( p \)
- Instead use hybrid of Gibbs sampling & fast multiscale SVD
Example 2: high-dimensional density estimation

\[ y_i = (y_{i1}, \ldots, y_{ip})^T \sim f \] with \( p \) large & \( f \) an unknown density

- Potentially use Dirichlet process mixtures of factor models
- Approach doesn’t scale well at all with \( p \)
- Instead use hybrid of Gibbs sampling & fast multiscale SVD
- Scalable, excellent mixing & empirical/predictive performance
Outline

Motivation

Hybrid Algorithms

EP-MCMC

aMCMC
Embarrassingly parallel MCMC

Divide large sample size \( n \) data set into many smaller data sets stored on different machines

\[ n \rightarrow \infty \]
Embarrassingly parallel MCMC

- Divide large sample size \( n \) data set into many smaller data sets stored on different machines.
- Draw posterior samples for each subset posterior in parallel.

Diagram:

- Big Data
  - "big n"
  - \( n \to \infty \)
- Data Subsets
- Subset Posteriors
- 'Magically' combine the results quickly & simply
Embarrassingly parallel MCMC

- Divide large sample size $n$ data set into many smaller data sets stored on different machines
- Draw posterior samples for each subset posterior in parallel
- ‘Magically’ combine the results quickly & simply
Toy Example: Logistic Regression

Subset posteriors are ‘noisy’ approximations of full data posterior.
‘Averaging’ of subset posteriors reduces this ‘noise’ & leads to an accurate posterior approximation.
Full data posterior density of \emph{inid} data $Y^{(n)}$

$$
\pi_n(\theta \mid Y^{(n)}) = \frac{\prod_{i=1}^{n} p_i(y_i \mid \theta) \pi(\theta)}{\int_{\Theta} \prod_{i=1}^{n} p_i(y_i \mid \theta) \pi(\theta) d\theta}.
$$
**Stochastic Approximation**

- Full data posterior density of *inid* data $Y^{(n)}$

$$\pi_n(\theta \mid Y^{(n)}) = \frac{\prod_{i=1}^{n} p_i(y_i \mid \theta)\pi(\theta)}{\int_{\Theta} \prod_{i=1}^{n} p_i(y_i \mid \theta)\pi(\theta)d\theta}.$$ 

- Divide full data $Y^{(n)}$ into $k$ subsets of size $m$:

$Y^{(n)} = (Y_{[1]}, \ldots, Y_{[j]}, \ldots, Y_{[k]})$. 

EP-MCMC 15
Stochastic Approximation

- Full data posterior density of \( \text{inid data } Y^{(n)} \)

\[
\pi_n(\theta \mid Y^{(n)}) = \frac{\prod_{i=1}^{n} p_i(y_i \mid \theta)\pi(\theta)}{\int_{\Theta} \prod_{i=1}^{n} p_i(y_i \mid \theta)\pi(\theta) \, d\theta}.
\]

- Divide full data \( Y^{(n)} \) into \( k \) subsets of size \( m \):
  \( Y^{(n)} = (Y_{[1]}, \ldots, Y_{[j]}, \ldots, Y_{[k]}) \).

- Subset posterior density for \( j \)th data subset

\[
\pi_{\gamma_{mn}}^{(n)}(\theta \mid Y_{[j]}) = \frac{\prod_{i \in [j]} (p_i(y_i \mid \theta))^{\gamma_n} \pi(\theta)}{\int_{\Theta} \prod_{i \in [j]} (p_i(y_i \mid \theta))^{\gamma_n} \pi(\theta) \, d\theta}.
\]
Stochastic Approximation

Full data posterior density of *inid* data $Y^{(n)}$

$$\pi_n(\theta \mid Y^{(n)}) = \frac{\prod_{i=1}^{n} p_i(y_i \mid \theta) \pi(\theta)}{\int_{\Theta} \prod_{i=1}^{n} p_i(y_i \mid \theta) \pi(\theta) d\theta}.$$ 

Divide full data $Y^{(n)}$ into $k$ subsets of size $m$:

$Y^{(n)} = (Y_{[1]}, \ldots, Y_{[j]}, \ldots, Y_{[k]})$.

Subset posterior density for $j$th data subset

$$\pi_{m_n}^{\gamma_n}(\theta \mid Y_{[j]}) = \frac{\prod_{i \in [j]} (p_i(y_i \mid \theta))^{\gamma_n} \pi(\theta)}{\int_{\Theta} \prod_{i \in [j]} (p_i(y_i \mid \theta))^{\gamma_n} \pi(\theta) d\theta}.$$ 

In most cases, $\gamma_n = O(k)$. 

---

EP-MCMC
Barycenter in Metric Spaces

Space of probability measures $\mathcal{M}$
Barycenter in Metric Spaces

Space of probability measures $\mathcal{M}$ with metric $\rho$

$$\Pi_M = \operatorname{argmin}_{\Pi \in \mathcal{M}} \sum_{i=1}^{n} \rho^2(\Pi, \Pi_i)$$
Wasserstein barycenter of Subset Posteriors (WASP)

Srivastava, Li & Dunson (2015)

2-Wasserstein distance between $\mu, \nu \in \mathcal{P}_2(\Theta)$

$$W_2(\mu, \nu) = \inf \left\{ \left( \mathbb{E}[d^2(X, Y)] \right)^{\frac{1}{2}} : \text{law}(X) = \mu, \text{law}(Y) = \nu \right\}.$$
2-Wasserstein distance between $\mu, \nu \in \mathcal{P}_2(\Theta)$

$$W_2(\mu, \nu) = \inf \left\{ \left( \mathbb{E} \left[ d^2(X, Y) \right] \right)^{\frac{1}{2}} : \text{law}(X) = \mu, \text{law}(Y) = \nu \right\}.$$ 

$\Pi_{\gamma_n m_j} (\cdot | Y[j])$ for $j = 1, \ldots, k$ are combined through WASP

$$\overline{\Pi}_n (\cdot | Y^{(n)}) = \arg\min_{\Pi \in \mathcal{P}_2(\Theta)} \frac{1}{k} \sum_{j=1}^{k} W_2^2 (\Pi, \Pi_{\gamma_n m_j} (\cdot | Y[j])).$$ [Agueh & Carlier (2011)]
WAsserstein barycenter of Subset Posteriors (WASP)

Srivastava, Li & Dunson (2015)

悬挂 2-Wasserstein distance between $\mu, \nu \in \mathcal{P}_2(\Theta)$

\[
W_2(\mu, \nu) = \inf \left\{ \left( \mathbb{E}[d^2(X, Y)] \right)^{\frac{1}{2}} : \text{law}(X) = \mu, \text{law}(Y) = \nu \right\}.
\]

悬挂 $\Pi_{\gamma_n}^{m_j}(\cdot \mid Y[j])$ for $j = 1, \ldots, k$ are combined through WASP

\[
\Pi_{\gamma_n}^{m_j}(\cdot \mid Y[j]) = \arg\min_{\Pi \in \mathcal{P}_2(\Theta)} \frac{1}{k} \sum_{j=1}^{k} W_2^2(\Pi, \Pi_{\gamma_n}^{m_j}(\cdot \mid Y[j])).
\]

[Agueh & Carlier (2011)]

悬挂 Plugging in $\Pi_{\gamma_n}^{m_j}(\cdot \mid Y[j])$ for $j = 1, \ldots, k$, a linear program (LP) can be used for fast estimation of an atomic approximation
Simple & fast Posterior Interval Estimation (PIE)

Li, Srivastava & Dunson (2015)

Usually report point & interval estimates for different 1-d functionals - multidimensional posterior difficult to interpret
Simple & fast Posterior Interval Estimation (PIE)

Li, Srivastava & Dunson (2015)

- Usually report point & interval estimates for different 1-d functionals - multidimensional posterior difficult to interpret
- WASP has explicit relationship with subset posteriors in 1-d
Simple & fast Posterior Interval Estimation (PIE)

Usually report point & interval estimates for different 1-d functionals - multidimensional posterior difficult to interpret

WASP has explicit relationship with subset posteriors in 1-d

Quantiles of WASP are simple averages of quantiles of subset posteriors
Simple & fast Posterior Interval Estimation (PIE)

Li, Srivastava & Dunson (2015)

- Usually report point & interval estimates for different 1-d functionals - multidimensional posterior difficult to interpret
- WASP has explicit relationship with subset posteriors in 1-d
- Quantiles of WASP are simple averages of quantiles of subset posteriors
- Leads to a super trivial algorithm - run MCMC for each subset & average quantiles - reminiscent of bag of little bootstraps
Simple & fast Posterior Interval Estimation (PIE)

- Usually report point & interval estimates for different 1-d functionals - multidimensional posterior difficult to interpret
- WASP has explicit relationship with subset posteriors in 1-d
- Quantiles of WASP are simple averages of quantiles of subset posteriors
- Leads to a super trivial algorithm - run MCMC for each subset & average quantiles - reminiscent of bag of little bootstraps
- Strong theory showing accuracy of the resulting approximation

Li, Srivastava & Dunson (2015)
Simple & fast Posterior Interval Estimation (PIE)

Li, Srivastava & Dunson (2015)

- Usually report point & interval estimates for different 1-d functionals - multidimensional posterior difficult to interpret
- WASP has explicit relationship with subset posteriors in 1-d
- Quantiles of WASP are simple averages of quantiles of subset posteriors
- Leads to a super trivial algorithm - run MCMC for each subset & average quantiles - reminiscent of bag of little bootstraps
- Strong theory showing accuracy of the resulting approximation
- Trivial to implement in STAN, which allows powered likelihoods
Results

We have implemented for rich variety of data & models

“Data don’t make any sense, we will have to resort to statistics.”
We have implemented for a rich variety of data & models:

- Logistic & linear random effects models
- Mixture models
- Matrix & tensor factorizations
- Gaussian process regression

Nonparametric models, dependence, hierarchical models, etc.

We compare to long runs of MCMC (when feasible) & VB.

WASP/PIE is much faster than MCMC & highly accurate.

Carefully designed VB implementations often do very well & are typically faster.
Results

- We have implemented for a rich variety of data & models
- Logistic & linear random effects models, mixture models, matrix & tensor factorizations, Gaussian process regression
- Nonparametric models, dependence, hierarchical models, etc.
Results

- We have implemented for rich variety of data & models
- Logistic & linear random effects models, mixture models, matrix & tensor factorizations, Gaussian process regression
- Nonparametric models, dependence, hierarchical models, etc.
- We compare to long runs of MCMC (when feasible) & VB
We have implemented for rich variety of data & models:

- Logistic & linear random effects models, mixture models, matrix & tensor factorizations, Gaussian process regression
- Nonparametric models, dependence, hierarchical models, etc.
- We compare to long runs of MCMC (when feasible) & VB
- WASP/PIE is much faster than MCMC & highly accurate
Results

We have implemented for rich variety of data & models

- Logistic & linear random effects models, mixture models, matrix & tensor factorizations, Gaussian process regression
- Nonparametric models, dependence, hierarchical models, etc.
- We compare to long runs of MCMC (when feasible) & VB
- WASP/PIE is much faster than MCMC & highly accurate
- Carefully designed VB implementations often do very well & are typically faster
Outline

Motivation

Hybrid Algorithms

EP-MCMC

aMCMC
Different way to speed up MCMC - replace expensive transition kernels with approximations
Different way to speed up MCMC - replace expensive transition kernels with approximations

e.g, approximate conditional distribution in Gibbs sampler with Gaussian or using a subsample of data
Different way to speed up MCMC - replace expensive transition kernels with approximations

e.g, approximate conditional distribution in Gibbs sampler with Gaussian or using a subsample of data

Can vastly speed up MCMC sampling in high-dimensional settings
Different way to speed up MCMC - replace expensive transition kernels with approximations

- e.g, approximate conditional distribution in Gibbs sampler with Gaussian or using a subsample of data
- Can vastly speed up MCMC sampling in high-dimensional settings
- Original MCMC sampler converges to a stationary distribution corresponding to the exact posterior
Different way to speed up MCMC - replace expensive transition kernels with approximations

e.g, approximate conditional distribution in Gibbs sampler with Gaussian or using a subsample of data

Can vastly speed up MCMC sampling in high-dimensional settings

Original MCMC sampler converges to a stationary distribution corresponding to the exact posterior

Not clear what happens when we start substituting in approximations - may diverge etc
Different way to speed up MCMC - replace expensive transition kernels with approximations

- e.g, approximate conditional distribution in Gibbs sampler with Gaussian or using a subsample of data

- Can vastly speed up MCMC sampling in high-dimensional settings

- Original MCMC sampler converges to a stationary distribution corresponding to the exact posterior

- Not clear what happens when we start substituting in approximations - may diverge etc

- Introducing a small amount of bias may give a great gain in reduction in MSE per computational time
aMCMC is used routinely in an essentially *ad hoc* manner
aMCMC Overview

- aMCMC is used routinely in an essentially *ad hoc* manner
- Our goal: obtain theory guarantees & use these to target design of algorithms
aMCMC Overview

- aMCMC is used routinely in an essentially *ad hoc* manner
- Our goal: obtain theory guarantees & use these to target design of algorithms
- Define ‘exact’ MCMC algorithm, which is computationally intractable but has good mixing
aMCMC Overview

- aMCMC is used routinely in an essentially *ad hoc* manner
- Our goal: obtain theory guarantees & use these to target design of algorithms
- Define ‘exact’ MCMC algorithm, which is computationally intractable but has good mixing
- ‘exact’ chain converges to stationary distribution corresponding to exact posterior
aMCMC is used routinely in an essentially *ad hoc* manner

Our goal: obtain theory guarantees & use these to target design of algorithms

Define ‘exact’ MCMC algorithm, which is computationally intractable but has good mixing

‘exact’ chain converges to stationary distribution corresponding to exact posterior

Approximate kernel in exact chain with more computationally tractable alternative
Define $s_\epsilon = \tau_1(\mathcal{P}) / \tau_1(\mathcal{P}_\epsilon) = \text{computational speed-up}$, $\tau_1(\mathcal{P}) =$ time for one step with transition kernel $\mathcal{P}$.
Sketch of theory

Define $s_c = \tau_1(\mathcal{P}) / \tau_1(\mathcal{P}_c) = \textit{computational speed-up}$, $\tau_1(\mathcal{P})$ = time for one step with transition kernel $\mathcal{P}$

**Interest**: optimizing computational time-accuracy tradeoff for estimators of $\Pi f = \int_\Theta f(\theta) \Pi(d\theta|x)$
Define $s_{\epsilon} = \tau_1(\mathcal{P}) / \tau_1(\mathcal{P}_\epsilon) = \textit{computational speed-up}$, $\tau_1(\mathcal{P}) =$ time for one step with transition kernel $\mathcal{P}$

Interest: optimizing computational time-accuracy tradeoff for estimators of $\Pi f = \int_{\Theta} f(\theta) \Pi(d\theta|x)$

We provide \textit{tight, finite sample} bounds on $L_2$ error
Sketch of theory

Define $s_\epsilon = \tau_1(\mathcal{P}) / \tau_1(\mathcal{P}_\epsilon) = \textit{computational speed-up}$, $\tau_1(\mathcal{P}) =$ time for one step with transition kernel $\mathcal{P}$

Interest: optimizing computational time-accuracy tradeoff for estimators of $\Pi f = \int_{\Theta} f(\theta) \Pi(d\theta|x)$

We provide \textit{tight, finite sample} bounds on $L_2$ error

aMCMC estimators win for low computational budgets but have asymptotic bias
Sketch of theory

Define \( s_\epsilon = \frac{\tau_1(\mathcal{P})}{\tau_1(\mathcal{P}_\epsilon)} \) = computational speed-up, \( \tau_1(\mathcal{P}) \) = time for one step with transition kernel \( \mathcal{P} \)

Interest: optimizing computational time-accuracy tradeoff for estimators of \( \Pi f = \int_\Theta f(\theta) \Pi(d\theta|x) \)

We provide tight, finite sample bounds on \( L_2 \) error

aMCMC estimators win for low computational budgets but have asymptotic bias

Often larger approximation error \( \rightarrow \) larger \( s_\epsilon \) & rougher approximations are better when speed super important
Define \( s_\epsilon = \frac{\tau_1(\mathcal{P})}{\tau_1(\mathcal{P}_\epsilon)} = \textit{computational speed-up}, \tau_1(\mathcal{P}) = \text{time for one step with transition kernel } \mathcal{P} \)

\( \textbf{Interest:} \) optimizing computational time-accuracy tradeoff for estimators of \( \Pi f = \int_\Theta f(\theta)\Pi(d\theta|\mathcal{x}) \)

We provide \textit{tight, finite sample} bounds on \( L_2 \) error

\( \text{aMCMC estimators win for low computational budgets but have asymptotic bias} \)

Often larger approximation error \( \rightarrow \) larger \( s_\epsilon \) & rougher approximations are better when speed super important

\( \textbf{We define a notion called} \textit{‘comp-minimax’} to formalize optimality with a computational budget \)
Application to SUSY dataset

$n = 5,000,000$ (0.5 million test), binary outcome & 18 continuous covariates
Application to SUSY dataset

$n = 5,000,000$ (0.5 million test), binary outcome & 18 continuous covariates

- Considered subsets sizes ranging from $|V| = 1,000$ to $4,500,000$

- Rate at which loss $\to 0$ with $\epsilon$ heavily dependent on loss

- For small computational budget & focus on posterior mean estimation, small subsets preferred

- As budget increases & loss focused more on tails (e.g., for interval estimation), optimal $|V|$ increases
Application to SUSY dataset

- \( n = 5,000,000 \) (0.5 million test), binary outcome & 18 continuous covariates
- Considered subsets sizes ranging from \(|V| = 1,000\) to \(4,500,000\)
- Considered different losses as function of \(|V|\)
Application to SUSY dataset

- $n = 5,000,000$ (0.5 million test), binary outcome & 18 continuous covariates
- Considered subsets sizes ranging from $|V| = 1,000$ to $4,500,000$
- Considered different losses as function of $|V|$
- Rate at which loss $\rightarrow 0$ with $\epsilon$ heavily dependent on loss
Application to SUSY dataset

- $n = 5,000,000$ (0.5 million test), binary outcome & 18 continuous covariates
- Considered subsets sizes ranging from $|V| = 1,000$ to 4,500,000
- Considered different losses as function of $|V|$
- Rate at which loss $\rightarrow 0$ with $\epsilon$ heavily dependent on loss
- For small computational budget & focus on posterior mean estimation, small subsets preferred
Application to SUSY dataset

- $n = 5,000,000$ (0.5 million test), binary outcome & 18 continuous covariates
- Considered subsets sizes ranging from $|V| = 1,000$ to $4,500,000$
- Considered different losses as function of $|V|$
- Rate at which loss $\rightarrow 0$ with $\epsilon$ heavily dependent on loss
- For small computational budget & focus on posterior mean estimation, small subsets preferred
- As budget increases & loss focused more on tails (e.g., for interval estimation), optimal $|V|$ increases
We also considered a nonparametric Bayes model:

$$\text{pr}(y_{i1} = c_1, \ldots, y_{ip} = c_p) = \sum_{h=1}^{k} \lambda_h \prod_{j=1}^{p} \psi_{hc_j}^{(j)},$$

a very useful model for multivariate categorical data
We also considered a nonparametric Bayes model:

\[
\text{pr}(y_{i1} = c_1, \ldots, y_{ip} = c_p) = \sum_{h=1}^{k} \lambda_h \prod_{j=1}^{p} \psi_{hc_j}^{(j)},
\]

a very useful model for multivariate categorical data

Sampling latent classes computationally prohibitive for huge \( n \)
We also considered a nonparametric Bayes model:

\[
\text{pr}(y_{i1} = c_1, \ldots, y_{ip} = c_p) = \sum_{h=1}^{k} \lambda_h \prod_{j=1}^{p} \psi_{hc_j}^{(j)},
\]

a very useful model for multivariate categorical data.

Sampling latent classes computationally prohibitive for huge \(n\)

Use adaptive Gaussian approximation - avoid sampling individual latent classes
We also considered a nonparametric Bayes model:

\[
\text{pr}(y_{i1} = c_1, \ldots, y_{ip} = c_p) = \sum_{h=1}^{k} \lambda_h \prod_{j=1}^{p} \psi_{hcj}^{(j)},
\]

a very useful model for multivariate categorical data

- Sampling latent classes computationally prohibitive for huge \( n \)
- Use adaptive Gaussian approximation - avoid sampling individual latent classes
- We have shown Assumptions 1-2, Assumption 2 result more general than this setting
Application 2: Mixture models & tensor factorizations

- We also considered a nonparametric Bayes model:

\[ \text{pr}(y_{i1} = c_1, \ldots, y_{ip} = c_p) = \sum_{h=1}^{k} \lambda_h \prod_{j=1}^{p} \psi_{hc_j}^{(j)}, \]

a very useful model for multivariate categorical data

- Sampling latent classes computationally prohibitive for huge \( n \)

- Use adaptive Gaussian approximation - avoid sampling individual latent classes

- We have shown Assumptions 1-2, Assumption 2 result more general than this setting

- Improved computation performance for large \( n \)
Application 3: Low rank approximation to GP

Gaussian process regression, $y_i = f(x_i) + \eta_i$, $\eta_i \sim N(0, \sigma^2)$
Application 3: Low rank approximation to GP

Gaussian process regression, $y_i = f(x_i) + \eta_i$, $\eta_i \sim N(0, \sigma^2)$

$f \sim GP$ prior with covariance $\tau^2 \exp(-\phi \|x_1 - x_2\|^2)$
Application 3: Low rank approximation to GP

- Gaussian process regression, $y_i = f(x_i) + \eta_i$, $\eta_i \sim N(0, \sigma^2)$
- $f \sim GP$ prior with covariance $\tau^2 \exp(-\phi ||x_1 - x_2||^2)$
- Discrete-uniform on $\phi$ & gamma priors on $\tau^{-2}, \sigma^{-2}$
Application 3: Low rank approximation to GP

- Gaussian process regression, $y_i = f(x_i) + \eta_i$, $\eta_i \sim N(0, \sigma^2)$
- $f \sim \text{GP}$ prior with covariance $\tau^2 \exp(-\phi||x_1 - x_2||^2)$
- Discrete-uniform on $\phi$ & gamma priors on $\tau^{-2}, \sigma^{-2}$
- Marginal MCMC sampler updates $\phi, \tau^{-2}, \sigma^{-2}$
Application 3: Low rank approximation to GP

Gaussian process regression, \( y_i = f(x_i) + \eta_i, \eta_i \sim N(0, \sigma^2) \)

\( f \sim GP \) prior with covariance \( \tau^2 \exp(-\phi ||x_1 - x_2||^2) \)

Discrete-uniform on \( \phi \) & gamma priors on \( \tau^{-2}, \sigma^{-2} \)

Marginal MCMC sampler updates \( \phi, \tau^{-2}, \sigma^{-2} \)

We show Assumption 1 holds under mild regularity conditions on “truth”, Assumption 2 holds for partial rank-\( r \) eigen approximation to \( \Sigma \)
Application 3: Low rank approximation to GP

- Gaussian process regression, $y_i = f(x_i) + \eta_i$, $\eta_i \sim N(0, \sigma^2)$
- $f \sim GP$ prior with covariance $\tau^2 \exp(-\phi||x_1 - x_2||^2)$
- Discrete-uniform on $\phi$ & gamma priors on $\tau^{-2}, \sigma^{-2}$
- Marginal MCMC sampler updates $\phi, \tau^{-2}, \sigma^{-2}$
- We show Assumption 1 holds under mild regularity conditions on “truth”, Assumption 2 holds for partial rank-$r$ eigen approximation to $\Sigma$
- Less accurate approximations clearly superior in practice for small computational budget
Discussion

Discussed three very general classes of algorithms for scalable Bayes inference
Discussion

Discussed three very general classes of algorithms for scalable Bayes inference

Hybrid, EP-MCMC & aMCMC - later two have ‘strong’ theoretical support
Discussion

Discussed three very general classes of algorithms for scalable Bayes inference

Hybrid, EP-MCMC & aMCMC - later two have ‘strong’ theoretical support

Interest in improving theory - avoid reliance on asymptotics in EP-MCMC & weaken assumptions in aMCMC
Discussed three very general classes of algorithms for scalable Bayes inference
- Hybrid, EP-MCMC & aMCMC - later two have ‘strong’ theoretical support
- Interest in improving theory - avoid reliance on asymptotics in EP-MCMC & weaken assumptions in aMCMC
- Useful to combine algorithms - e.g., run aMCMC for each subset
Discussion

- Discussed three very general classes of algorithms for scalable Bayes inference
- Hybrid, EP-MCMC & aMCMC - later two have ‘strong’ theoretical support
- Interest in improving theory - avoid reliance on asymptotics in EP-MCMC & weaken assumptions in aMCMC
- Useful to combine algorithms - e.g., run aMCMC for each subset
- By looking at aMCMC algorithms through our theory lens, suggests new & improved algorithms
Acknowledgment

This work was supported by grants from NSF, NIH, and NIEHS.