Scaling up Bayesian Inference

David Dunson

Departments of Statistical Science & Mathematics, Duke University

July 2, 2018

Outline

Motivation & background

EP-MCMC

aMCMC

Designer MCMC

Generalized Bayes

Complex & high-dimensional data

- Focus: new methods for analyzing & interpreting complex, high-dimensional data

Complex & high-dimensional data

- Focus: new methods for analyzing & interpreting complex, high-dimensional data
- Arise routinely in broad fields of **sciences**, engineering & even arts & humanities

Complex & high-dimensional data

- <u>Focus</u>: new methods for analyzing & interpreting complex, high-dimensional data
- Arise routinely in broad fields of **sciences**, engineering & even arts & humanities
- Statistical, computational & mathematical methods to solve real problems in broad areas

Complex & high-dimensional data

- <u>Focus</u>: new methods for analyzing & interpreting complex, high-dimensional data
- Arise routinely in broad fields of **sciences**, engineering & even arts & humanities
- Statistical, computational & mathematical methods to solve real problems in broad areas
- Despite huge interest in big data, there are <u>vast</u> gaps that have fundamentally limited progress in many fields

Typical approaches to big data

- There is an increasingly immense literature focused on big data

Typical approaches to big data

- There is an increasingly immense literature focused on big data
- Most of the focus has been on penalized optimization methods

Typical approaches to big data

- There is an increasingly immense literature focused on big data
- Most of the focus has been on penalized optimization methods
- Rapidly obtaining a point estimate even when sample size n & overall 'size' of data is immense

Typical approaches to big data

- There is an increasingly immense literature focused on big data
- Most of the focus has been on penalized optimization methods
- Rapidly obtaining a point estimate even when sample size n & overall 'size' of data is immense
- Huge focus on specific settings - e.g., linear regression, identifying cats in images, etc

Typical approaches to big data

- There is an increasingly immense literature focused on big data
- Most of the focus has been on penalized optimization methods
- Rapidly obtaining a point estimate even when sample size n & overall 'size' of data is immense
- Huge focus on specific settings - e.g., linear regression, identifying cats in images, etc
- Bandwagons: most people work on very similar problems, while critical open problems remain untouched

My focus - probability models

"I wish we hadn't learned probability
'cause I don't think our odds are good."

My focus - probability models

"I wish we hadn't learned probability 'cause I don't think our odds are good."

- General probabilistic inference algorithms for complex data

My focus - probability models

"I wish we hadn't learned probability
'cause I don't think our odds are good."

- General probabilistic inference algorithms for complex data
- We would like to be able to handle arbitrarily complex probability models

My focus - probability models

"I wish we hadn't learned probability 'cause I don't think our odds are good."

- General probabilistic inference algorithms for complex data
- We would like to be able to handle arbitrarily complex probability models
- Algorithms scalable to huge data - potentially using many computers

My focus - probability models

"I wish we hadn't learned probability 'cause I don't think our odds are good."

- General probabilistic inference algorithms for complex data
- We would like to be able to handle arbitrarily complex probability models
- Algorithms scalable to huge data - potentially using many computers

My focus - probability models

"I wish we hadn't learned probability 'cause I don't think our odds are good."

- General probabilistic inference algorithms for complex data
- We would like to be able to handle arbitrarily complex probability models
- Algorithms scalable to huge data - potentially using many computers

- Accurate uncertainty quantification (UQ) is a critical issue

My focus - probability models

"I wish we hadn't learned probability
'cause I don't think our odds are good."

- General probabilistic inference algorithms for complex data
- We would like to be able to handle arbitrarily complex probability models
- Algorithms scalable to huge data - potentially using many computers
- Accurate uncertainty quantification (UQ) is a critical issue
- Robustness of inferences also crucial

Bayes approaches

- Bayesian methods offer an attractive general approach for modeling complex data

Bayes approaches

- Bayesian methods offer an attractive general approach for modeling complex data
- Choosing a prior $\pi(\theta)$ & likelihood $L(Y^{(n)}|\theta)$, the posterior is

$$\pi_n(\theta|Y^{(n)}) = \frac{\pi(\theta)L(Y^{(n)}|\theta)}{\int \pi(\theta)L(Y^{(n)}|\theta)d\theta} = \frac{\pi(\theta)L(Y^{(n)}|\theta)}{L(Y^{(n)})}.$$

Bayes approaches

- Bayesian methods offer an attractive general approach for modeling complex data
- Choosing a prior $\pi(\theta)$ & likelihood $L(Y^{(n)}|\theta)$, the posterior is

$$\pi_n(\theta|Y^{(n)}) = \frac{\pi(\theta)L(Y^{(n)}|\theta)}{\int \pi(\theta)L(Y^{(n)}|\theta)d\theta} = \frac{\pi(\theta)L(Y^{(n)}|\theta)}{L(Y^{(n)})}.$$

- Often θ is moderate to high-dimensional & the integral in the denominator is intractable

Bayes approaches

- Bayesian methods offer an attractive general approach for modeling complex data
- Choosing a prior $\pi(\theta)$ & likelihood $L(Y^{(n)}|\theta)$, the posterior is

$$\pi_n(\theta|Y^{(n)}) = \frac{\pi(\theta)L(Y^{(n)}|\theta)}{\int \pi(\theta)L(Y^{(n)}|\theta)d\theta} = \frac{\pi(\theta)L(Y^{(n)}|\theta)}{L(Y^{(n)})}.$$

- Often θ is moderate to high-dimensional & the integral in the denominator is intractable
- Accurate analytic approximations to the posterior have proven elusive outside of narrow settings

Bayes approaches

- Bayesian methods offer an attractive general approach for modeling complex data
- Choosing a prior $\pi(\theta)$ & likelihood $L(Y^{(n)}|\theta)$, the posterior is

$$\pi_n(\theta|Y^{(n)}) = \frac{\pi(\theta)L(Y^{(n)}|\theta)}{\int \pi(\theta)L(Y^{(n)}|\theta)d\theta} = \frac{\pi(\theta)L(Y^{(n)}|\theta)}{L(Y^{(n)})}.$$

- Often θ is moderate to high-dimensional & the integral in the denominator is intractable
- Accurate analytic approximations to the posterior have proven elusive outside of narrow settings
- Markov chain Monte Carlo (MCMC) & other posterior sampling algorithms remain the standard

Bayes approaches

- Bayesian methods offer an attractive general approach for modeling complex data
- Choosing a prior $\pi(\theta)$ & likelihood $L(Y^{(n)}|\theta)$, the posterior is

$$\pi_n(\theta|Y^{(n)}) = \frac{\pi(\theta)L(Y^{(n)}|\theta)}{\int \pi(\theta)L(Y^{(n)}|\theta)d\theta} = \frac{\pi(\theta)L(Y^{(n)}|\theta)}{L(Y^{(n)})}.$$

- Often θ is moderate to high-dimensional & the integral in the denominator is intractable
- Accurate analytic approximations to the posterior have proven elusive outside of narrow settings
- Markov chain Monte Carlo (MCMC) & other posterior sampling algorithms remain the standard
- Scaling MCMC to big & complex settings challenging

MCMC & Computational bottlenecks

- MCMC constructs Markov chain with stationary distribution $\pi_n(\theta|Y^{(n)})$

MCMC & Computational bottlenecks

- MCMC constructs Markov chain with stationary distribution $\pi_n(\theta|Y^{(n)})$
- A *transition kernel* is carefully chosen & iterative sampling proceeds

MCMC & Computational bottlenecks

- MCMC constructs Markov chain with stationary distribution $\pi_n(\theta | Y^{(n)})$
- A *transition kernel* is carefully chosen & iterative sampling proceeds
- Time per iteration increases with # of parameters/unknowns

MCMC & Computational bottlenecks

- MCMC constructs Markov chain with stationary distribution $\pi_n(\theta|Y^{(n)})$
- A *transition kernel* is carefully chosen & iterative sampling proceeds
- Time per iteration increases with # of parameters/unknowns
- Mixing worse as dimension of data increases

MCMC & Computational bottlenecks

- MCMC constructs Markov chain with stationary distribution $\pi_n(\theta|Y^{(n)})$
- A *transition kernel* is carefully chosen & iterative sampling proceeds
- Time per iteration increases with # of parameters/unknowns
- Mixing worse as dimension of data increases
- Storing & basic processing on big data sets is problematic

MCMC & Computational bottlenecks

- MCMC constructs Markov chain with stationary distribution $\pi_n(\theta|Y^{(n)})$
- A *transition kernel* is carefully chosen & iterative sampling proceeds
- Time per iteration increases with # of parameters/unknowns
- Mixing worse as dimension of data increases
- Storing & basic processing on big data sets is problematic
- Usually multiple likelihood and/or gradient evaluations at each iteration

Some Solutions

- **Embarrassingly parallel (EP) MCMC**: run MCMC in parallel for different subsets of data & combine.

Some Solutions

- **Embarrassingly parallel (EP) MCMC**: run MCMC in parallel for different subsets of data & combine.
- **Approximate MCMC**: Approximate expensive to evaluate transition kernels.

Some Solutions

- **Embarrassingly parallel (EP) MCMC**: run MCMC in parallel for different subsets of data & combine.
- **Approximate MCMC**: Approximate expensive to evaluate transition kernels.
- **Designer MCMC**: Carefully design MCMC transition kernels to be scalable

Some Solutions

- **Embarrassingly parallel (EP) MCMC**: run MCMC in parallel for different subsets of data & combine.
- **Approximate MCMC**: Approximate expensive to evaluate transition kernels.
- **Designer MCMC**: Carefully design MCMC transition kernels to be scalable
- **Generalized Bayes**: Take a step away from full Bayes inferences for scalability & robustness

Outline

Motivation & background

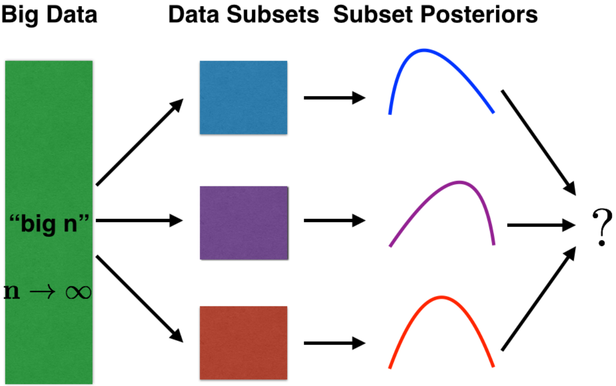
EP-MCMC

aMCMC

Designer MCMC

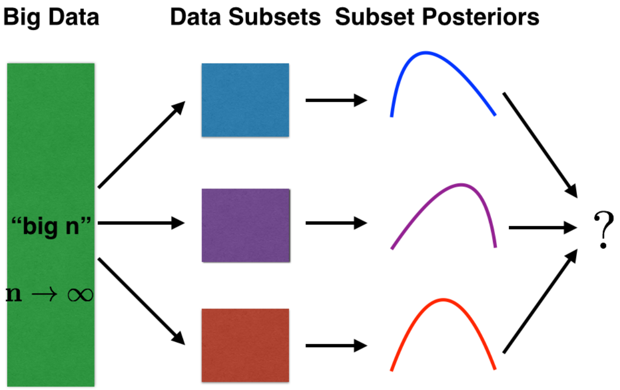
Generalized Bayes

Embarrassingly parallel MCMC



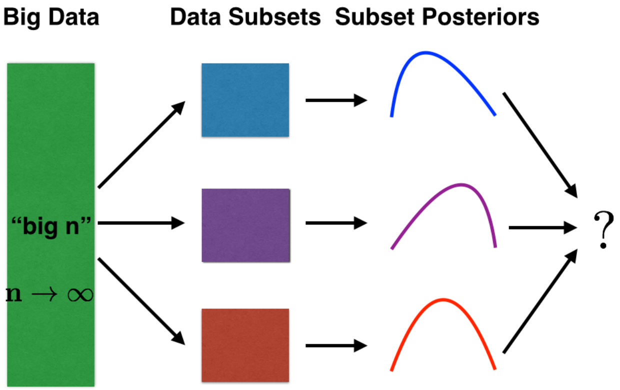
- Divide large sample size n data set into many smaller data sets stored on different machines

Embarrassingly parallel MCMC



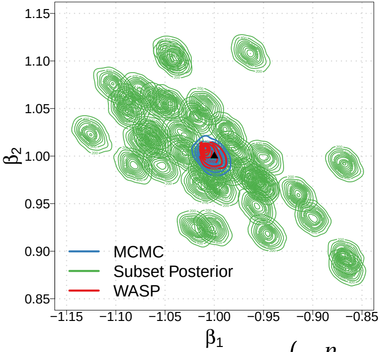
- Divide large sample size n data set into many smaller data sets stored on different machines
- Draw posterior samples for each subset posterior in parallel

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

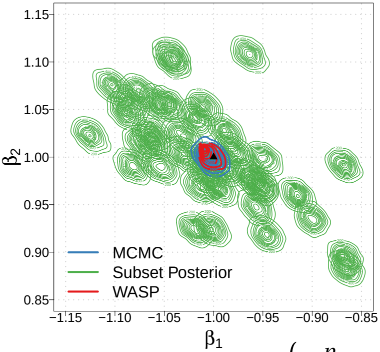
Toy Example: Logistic Regression



$$\mathrm{pr}(y_i = 1 | x_{i1}, \ldots, x_{ip}, \theta) = \frac{\exp\left(\sum_{j=1}^{p} x_{ij}\beta_j\right)}{1 + \exp\left(\sum_{j=1}^{p} x_{ij}\beta_j\right)}.$$

- Subset posteriors: ‘noisy’ approximations of full data posterior.

Toy Example: Logistic Regression



$$\text{pr}(y_i = 1 | x_{i1}, \ldots, x_{ip}, \theta) = \frac{\exp\left(\sum_{j=1}^{p} x_{ij}\beta_j\right)}{1 + \exp\left(\sum_{j=1}^{p} x_{ij}\beta_j\right)}.$$

- Subset posteriors: ‘noisy’ approximations of full data posterior.
- ‘Averaging’ of subset posteriors reduces this ‘noise’ & leads to an accurate posterior approximation.

Stochastic Approximation

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

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

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

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 jth data subset

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

Stochastic Approximation

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

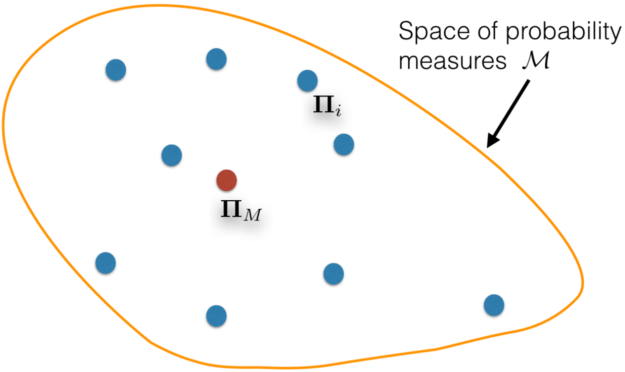
$$\pi_n(\theta \mid Y^{(n)}) = \frac{\prod_{i=1}^n p_i(y_i \mid \theta)\pi(\theta)}{\int_\Theta \prod_{i=1}^n p_i(y_i \mid \theta)\pi(\theta)d\theta}.$$

- 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 jth data subset

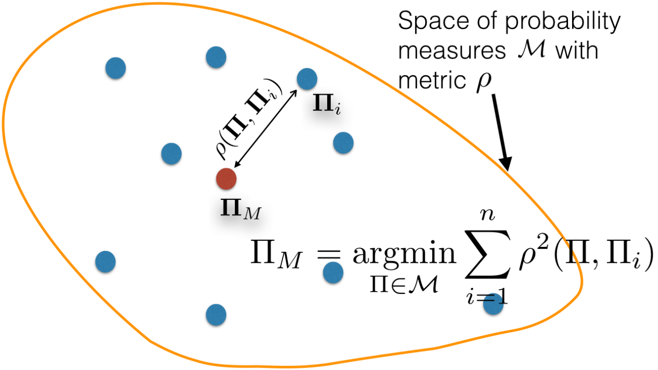
$$\pi_m^\gamma(\theta \mid Y_{[j]}) = \frac{\prod_{i \in [j]} (p_i(y_i \mid \theta))^\gamma \pi(\theta)}{\int_\Theta \prod_{i \in [j]} (p_i(y_i \mid \theta))^\gamma \pi(\theta)d\theta}.$$

- $\gamma = O(k)$ - chosen to minimize approximation error

Barycenter in Metric Spaces



Barycenter in Metric Spaces



$$\Pi_M = \operatorname*{argmin}_{\Pi \in \mathcal{M}} \sum_{i=1}^{n} \rho^2(\Pi, \Pi_i)$$

WAsserstein barycenter of Subset Posteriors (WASP)

Srivastava et al (2015)

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

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

WAsserstein barycenter of Subset Posteriors (WASP)

Srivastava et al (2015)

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

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

- $\Pi_m^{\gamma}(\cdot \mid Y_{[j]})$ for $j = 1, \ldots, k$ are combined through WASP

$$\overline{\Pi}_n^{\gamma}(\cdot \mid Y^{(n)}) = \underset{\Pi \in \mathscr{P}_2(\Theta)}{\text{argmin}} \frac{1}{k} \sum_{j=1}^{k} W_2^2(\Pi, \Pi_m^{\gamma}(\cdot \mid Y_{[j]})).$$ [Agueh & Carlier (2011)]

WAsserstein barycenter of Subset Posteriors (WASP)

Srivastava et al (2015)

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

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

- $\Pi_m^{\gamma}(\cdot \mid Y_{[j]})$ for $j = 1, \ldots, k$ are combined through WASP

$$\overline{\Pi}_n^{\gamma}(\cdot \mid Y^{(n)}) = \underset{\Pi \in \mathscr{P}_2(\Theta)}{\operatorname{argmin}} \frac{1}{k} \sum_{j=1}^{k} W_2^2(\Pi, \Pi_m^{\gamma}(\cdot \mid Y_{[j]})).$$ [Agueh & Carlier (2011)]

- Plugging in $\widehat{\Pi}_m^{\gamma}(\cdot \mid Y_{[j]})$ 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 (2017)

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

Simple & Fast Posterior Interval Estimation (PIE)

Li, Srivastava & Dunson (2017)

- 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)

Li, Srivastava & Dunson (2017)

- 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 (2017)

- 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)

Li, Srivastava & Dunson (2017)

- 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

Simple & Fast Posterior Interval Estimation (PIE)

Li, Srivastava & Dunson (2017)

- 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
- Can implement in *STAN*, which allows powered likelihoods

Theory on PIE/1-d WASP

- We show 1-d WASP $\overline{\Pi}_n(\xi|Y^{(n)})$ is highly accurate approximation to exact posterior $\Pi_n(\xi|Y^{(n)})$

Theory on PIE/1-d WASP

- We show 1-d WASP $\overline{\Pi}_n(\xi | Y^{(n)})$ is highly accurate approximation to exact posterior $\Pi_n(\xi | Y^{(n)})$
- As subset sample size m increases, W_2 distance between them decreases at faster than parametric rate $o_p(n^{-1/2})$

Theory on PIE/1-d WASP

- We show 1-d WASP $\overline{\Pi}_n(\xi|Y^{(n)})$ is highly accurate approximation to exact posterior $\Pi_n(\xi|Y^{(n)})$
- As subset sample size m increases, W_2 distance between them decreases at faster than parametric rate $o_p(n^{-1/2})$
- Theorem allows $k = O(n^c)$ and $m = O(n^{1-c})$ for any $c \in (0, 1)$, so m can increase very slowly relative to k (*recall* $n = mk$)

Theory on PIE/1-d WASP

- We show 1-d WASP $\overline{\Pi}_n(\xi|Y^{(n)})$ is highly accurate approximation to exact posterior $\Pi_n(\xi|Y^{(n)})$
- As subset sample size m increases, W_2 distance between them decreases at faster than parametric rate $o_p(n^{-1/2})$
- Theorem allows $k = O(n^c)$ and $m = O(n^{1-c})$ for any $c \in (0, 1)$, so m can increase very slowly relative to k (*recall* $n = mk$)
- Their biases, variances, quantiles only differ in high orders of the total sample size

Theory on PIE/1-d WASP

- We show 1-d WASP $\overline{\Pi}_n(\xi|Y^{(n)})$ is highly accurate approximation to exact posterior $\Pi_n(\xi|Y^{(n)})$
- As subset sample size m increases, W_2 distance between them decreases at faster than parametric rate $o_p(n^{-1/2})$
- Theorem allows $k = O(n^c)$ and $m = O(n^{1-c})$ for any $c \in (0, 1)$, so m can increase very slowly relative to k (*recall* $n = mk$)
- Their biases, variances, quantiles only differ in high orders of the total sample size
- Conditions: standard, mild conditions on likelihood + prior finite 2nd moment & uniform integrabiity of subset posteriors

Results

- We have implemented for rich variety of data & models

Results

- We have implemented for rich variety of data & models
- Logistic & linear random effects models, mixture models, matrix & tensor factorizations, Gaussian process regression

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.

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

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 <u>much</u> 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 <u>much</u> faster than MCMC & highly accurate
- Carefully designed VB implementations often do very well

Outline

Motivation & background

EP-MCMC

aMCMC

Designer MCMC

Generalized Bayes

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

aMCMC *Johndrow, Mattingly, Mukherjee & Dunson*

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

aMCMC *Johndrow, Mattingly, Mukherjee & Dunson*

- Different way to speed up MCMC - replace expensive transition kernels with approximations
- For example, approximate a conditional distribution in Gibbs sampler with a Gaussian or using a subsample of data
- Can potentially vastly speed up MCMC sampling in high-dimensional settings

aMCMC *Johndrow, Mattingly, Mukherjee & Dunson*

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

aMCMC *Johndrow, Mattingly, Mukherjee & Dunson*

- Different way to speed up MCMC - replace expensive transition kernels with approximations
- For example, approximate a conditional distribution in Gibbs sampler with a Gaussian or using a subsample of data
- Can potentially 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

aMCMC Overview

- 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

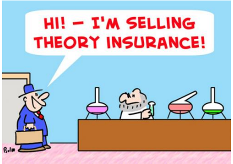
aMCMC Overview

- 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
- Approximate kernel in exact chain with more computationally tractable alternative

aMCMC Overview

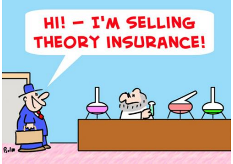
- aMCMC is used routinely in an essentially *ad hoc* manner
- Our goal: obtain theory guarantees & use these to target design of algorithms
- Define ‘exact’ MCMC algorithm, which is computationally intractable but has good mixing
- ‘exact’ chain converges to stationary distribution corresponding to exact posterior
- Approximate kernel in exact chain with more computationally tractable alternative
- ‘Comp-minimax’ = optimal approx level conditional on computational time

Sketch of theory



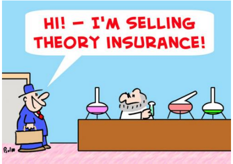
- Define $s_\epsilon = \tau_1(\mathscr{P})/\tau_1(\mathscr{P}_\epsilon)$ = *computational speed-up*, $\tau_1(\mathscr{P})$ = time for one step with transition kernel $\mathscr{P}$

Sketch of theory

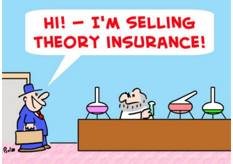


- Define $s_\epsilon = \tau_1(\mathscr{P})/\tau_1(\mathscr{P}_\epsilon)$ = *computational speed-up*, $\tau_1(\mathscr{P})$ = time for one step with transition kernel $\mathscr{P}$
- Interest: optimizing computational time-accuracy tradeoff for estimators of $\Pi f = \int_\Theta f(\theta)\Pi(d\theta|x)$

Sketch of theory



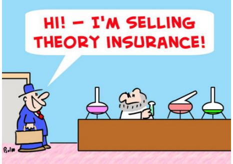
- Define $s_\epsilon = \tau_1(\mathscr{P})/\tau_1(\mathscr{P}_\epsilon)$ = *computational speed-up*, $\tau_1(\mathscr{P})$ = time for one step with transition kernel $\mathscr{P}$
- Interest: optimizing computational time-accuracy tradeoff for estimators of $\Pi f = \int_\Theta f(\theta)\Pi(d\theta|x)$
- We provide *tight, finite sample* bounds on L_2 error



Sketch of theory

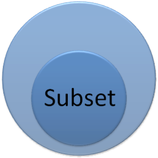
- Define $s_\epsilon = \tau_1(\mathscr{P})/\tau_1(\mathscr{P}_\epsilon)$ = *computational speed-up*, $\tau_1(\mathscr{P})$ = time for one step with transition kernel $\mathscr{P}$
- Interest: optimizing computational time-accuracy tradeoff for estimators of $\Pi f = \int_\Theta f(\theta)\Pi(d\theta|x)$
- We provide *tight, finite sample* bounds on L_2 error
- aMCMC estimators win for low computational budgets but have asymptotic bias

Sketch of theory



- Define $s_\epsilon = \tau_1(\mathscr{P})/\tau_1(\mathscr{P}_\epsilon)$ = *computational speed-up*, $\tau_1(\mathscr{P})$ = time for one step with transition kernel $\mathscr{P}$
- 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 → larger s_ϵ & rougher approximations are better when speed super important

Ex 1: Approximations using subsets

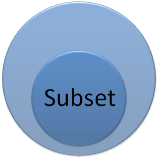


- Replace the full data likelihood with

$$L_\epsilon(x \mid \theta) = \left(\prod_{i \in V} L(x_i \mid \theta) \right)^{N/|V|},$$

for randomly chosen subset $V \subset \{1, \ldots, n\}$.

Ex 1: Approximations using subsets

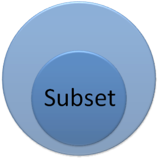


- Replace the full data likelihood with

$$L_\epsilon(x \mid \theta) = \left(\prod_{i \in V} L(x_i \mid \theta) \right)^{N/|V|},$$

  for randomly chosen subset $V \subset \{1, \ldots, n\}$.
- Applied to Pólya-Gamma data augmentation for logistic regression

Ex 1: Approximations using subsets

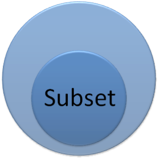


- Replace the full data likelihood with

$$L_\epsilon(x \mid \theta) = \left(\prod_{i \in V} L(x_i \mid \theta) \right)^{N/|V|},$$

  for randomly chosen subset $V \subset \{1, \ldots, n\}$.
- Applied to Pólya-Gamma data augmentation for logistic regression
- Different V at each iteration – trivial modification to Gibbs

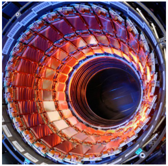
Ex 1: Approximations using subsets



- Replace the full data likelihood with

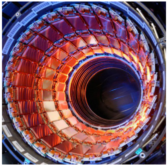
$$L_\epsilon(x \mid \theta) = \left(\prod_{i \in V} L(x_i \mid \theta) \right)^{N/|V|},$$

  for randomly chosen subset $V \subset \{1, \ldots, n\}$.
- Applied to Pólya-Gamma data augmentation for logistic regression
- Different V at each iteration – trivial modification to Gibbs
- Assumptions hold with high probability for subsets > minimal size (wrt distribution of subsets, data & kernel).



Application to SUSY dataset

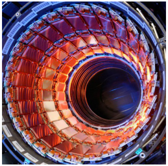
- $n = 5,000,000$ (0.5 million test), binary outcome & 18 continuous covariates



Application to SUSY dataset

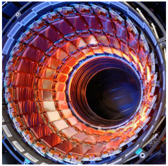
- $n = 5,000,000$ (0.5 million test), binary outcome & 18 continuous covariates
- Considered subsets sizes ranging from $|V| = 1,000$ to $4,500,000$

Application to SUSY dataset

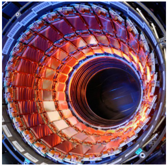


- $n = 5,000,000$ (0.5 million test), binary outcome & 18 continuous covariates
- Considered subsets sizes ranging from $|V| = 1,000$ to $4,500,000$
- Considered different losses as function of $|V|$

Application to SUSY dataset

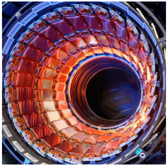


- $n = 5,000,000$ (0.5 million test), binary outcome & 18 continuous covariates
- Considered subsets sizes ranging from $|V| = 1,000$ to $4,500,000$
- Considered different losses as function of $|V|$
- Rate at which loss $\rightarrow 0$ with ϵ heavily dependent on loss



Application to SUSY dataset

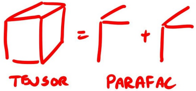
- $n = 5,000,000$ (0.5 million test), binary outcome & 18 continuous covariates
- Considered subsets sizes ranging from $|V| = 1,000$ to $4,500,000$
- Considered different losses as function of $|V|$
- Rate at which loss $\to 0$ with ϵ heavily dependent on loss
- For small computational budget & focus 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 $\to 0$ with ϵ 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 2: Mixture models & tensor factorizations

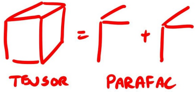


- We also considered a nonparametric Bayes model:

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

a very useful model for multivariate categorical data

Application 2: Mixture models & tensor factorizations

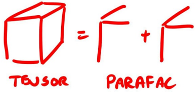


- We also considered a nonparametric Bayes model:

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

  a very useful model for multivariate categorical data
- Dunson & Xing (2009) - a data augmentation Gibbs sampler

Application 2: Mixture models & tensor factorizations

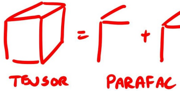


- We also considered a nonparametric Bayes model:

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

  a very useful model for multivariate categorical data
- Dunson & Xing (2009) - a data augmentation Gibbs sampler
- Sampling latent classes computationally prohibitive for huge n

Application 2: Mixture models & tensor factorizations

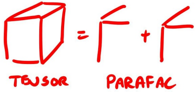


- We also considered a nonparametric Bayes model:

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

  a very useful model for multivariate categorical data
- Dunson & Xing (2009) - a data augmentation Gibbs sampler
- Sampling latent classes computationally prohibitive for huge n
- Use adaptive Gaussian approximation - avoid sampling individual latent classes

Application 2: Mixture models & tensor factorizations

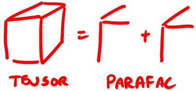


- We also considered a nonparametric Bayes model:

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

  a very useful model for multivariate categorical data
- Dunson & Xing (2009) - a data augmentation Gibbs sampler
- 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

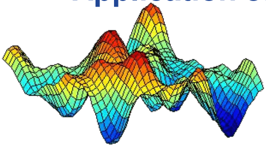


- We also considered a nonparametric Bayes model:

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

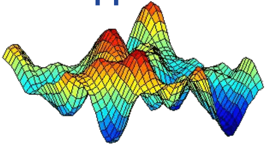
  a very useful model for multivariate categorical data
- Dunson & Xing (2009) - a data augmentation Gibbs sampler
- Sampling latent classes computationally prohibitive for huge n
- Use adaptive Gaussian approximation - avoid sampling individual latent classes
- We have shown Assumptions 1-2, Assumption 2 result more general than this setting
- Improved computation performance for large n

Application 3: Low rank approximation to GP



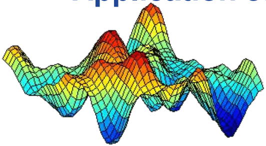
- Gaussian process regression, $y_i = f(x_i) + \eta_i,\, \eta_i \sim N(0, \sigma^2)$

Application 3: Low rank approximation to GP



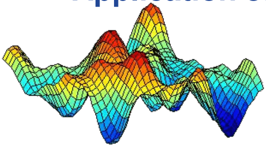
- Gaussian process regression, $y_i = f(x_i) + \eta_i,\ \eta_i \sim N(0, \sigma^2)$
- $f \sim GP$ prior with covariance $\tau^2 \exp(-\phi||x_1 - x_2||^2)$

Application 3: Low rank approximation to GP



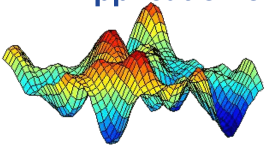
- Gaussian process regression, $y_i = f(x_i) + \eta_i$, $\eta_i \sim N(0, \sigma^2)$
- $f \sim GP$ prior with covariance $\tau^2 \exp(-\phi||x_1 - x_2||^2)$
- Discrete-uniform on ϕ & gamma priors on τ^{-2}, σ^{-2}

Application 3: Low rank approximation to GP



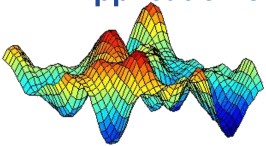
- Gaussian process regression, $y_i = f(x_i) + \eta_i,\ \eta_i \sim N(0, \sigma^2)$
- $f \sim GP$ prior with covariance $\tau^2 \exp(-\phi||x_1 - x_2||^2)$
- Discrete-uniform on ϕ & gamma priors on τ^{-2}, σ^{-2}
- Marginal MCMC sampler updates $\phi, \tau^{-2}, \sigma^{-2}$

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 ϕ & gamma priors on τ^{-2}, σ^{-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 Σ

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 ϕ & gamma priors on τ^{-2}, σ^{-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 Σ
- Less accurate approximations clearly superior in practice for small computational budget

Outline

Motivation & background

EP-MCMC

aMCMC

Designer MCMC

Generalized Bayes

Designer MCMC

- In designing MCMC for large datasets, we need to be careful & clever about the transition kernel

Designer MCMC

- In designing MCMC for large datasets, we need to be careful & clever about the transition kernel
- Try to exploit structure in the model to accelerate computation

Designer MCMC

- In designing MCMC for large datasets, we need to be careful & clever about the transition kernel
- Try to exploit structure in the model to accelerate computation
- Increasing rich literature - relying on (biased) subsampling, new classes of MCMC algorithms, etc

Designer MCMC

- In designing MCMC for large datasets, we need to be careful & clever about the transition kernel
- Try to exploit structure in the model to accelerate computation
- Increasing rich literature - relying on (biased) subsampling, new classes of MCMC algorithms, etc
- I'll illustrate briefly with a new class of multiscale MCMC algorithms

Multiscale Metropolis-Hastings *Young, Mattingly & Dunson*

- Exploit a multiscale characterization the log-likelihood to choose a truncation approximation

Multiscale Metropolis-Hastings *Young, Mattingly & Dunson*

- Exploit a multiscale characterization the log-likelihood to choose a truncation approximation
- Run two Markov chains in parallel targeting the true & approximate posteriors

Multiscale Metropolis-Hastings *Young, Mattingly & Dunson*

- Exploit a multiscale characterization the log-likelihood to choose a truncation approximation
- Run two Markov chains in parallel targeting the true & approximate posteriors
- Algorithm 1: use approximating chain as proposals for true chain

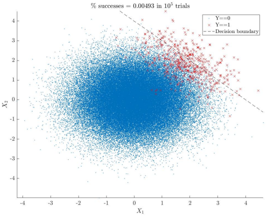
Multiscale Metropolis-Hastings *Young, Mattingly & Dunson*

- Exploit a multiscale characterization the log-likelihood to choose a truncation approximation
- Run two Markov chains in parallel targeting the true & approximate posteriors
- Algorithm 1: use approximating chain as proposals for true chain
- Algorithm 2: swap states of two chains (as in parallel tempering)

Multiscale Metropolis-Hastings *Young, Mattingly & Dunson*

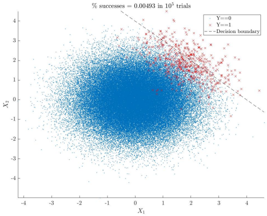
- Exploit a multiscale characterization the log-likelihood to choose a truncation approximation
- Run two Markov chains in parallel targeting the true & approximate posteriors
- <u>Algorithm 1</u>: use approximating chain as proposals for true chain
- <u>Algorithm 2</u>: swap states of two chains (as in parallel tempering)
- Given time, I'll just illustrate briefly with two canonical examples

Selection subsampling for logistic regression



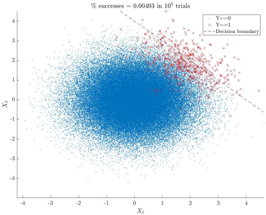
- In big data applications, the proportion of 1s is often very badly imbalanced

Selection subsampling for logistic regression



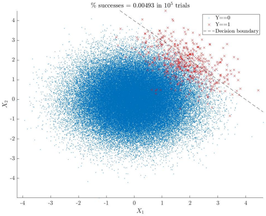
- In big data applications, the proportion of 1s is often very badly imbalanced
- This can lead to horrendous mixing for popular MCMC algorithms (Johndrow et al)

Selection subsampling for logistic regression



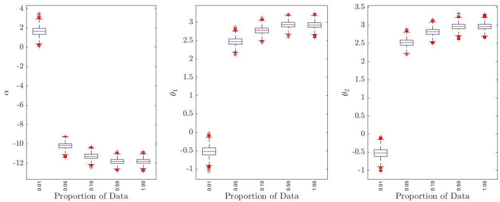
- In big data applications, the proportion of 1s is often very badly imbalanced
- This can lead to horrendous mixing for popular MCMC algorithms (Johndrow et al)
- Scalable algorithms using uniform subsampling (including EP-MCMC) fail - all zeros in subsamples

Selection subsampling for logistic regression



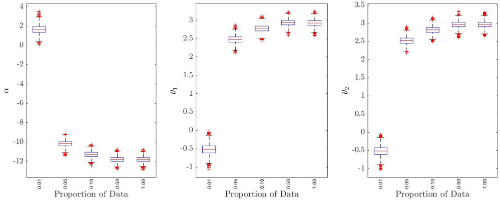
- In big data applications, the proportion of 1s is often very badly imbalanced
- This can lead to horrendous mixing for popular MCMC algorithms (Johndrow et al)
- Scalable algorithms using uniform subsampling (including EP-MCMC) fail - all zeros in subsamples
- Calculate full data MAP θ_{MAP} & select data in subset to maximize information about full data log-likelihood

Results for logistic regression simulation



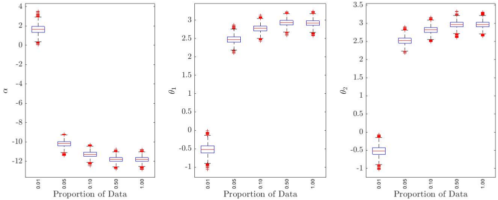
- Generated data from an imbalanced logistic regression model with $N = 10^5$ & $\theta = (-12, 3, 3)$

Results for logistic regression simulation



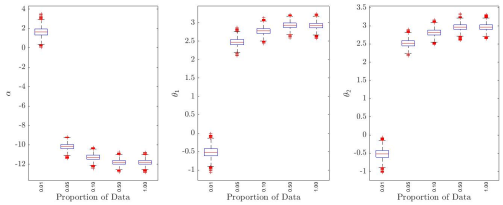
- Generated data from an imbalanced logistic regression model with $N = 10^5$ & $\theta = (-12, 3, 3)$
- Big enough to illustrate the advantages of proposed approach while still being able to run MCMC on full data

Results for logistic regression simulation



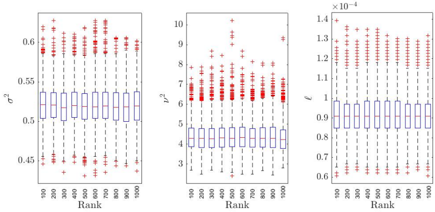
- Generated data from an imbalanced logistic regression model with $N = 10^5$ & $\theta = (-12, 3, 3)$
- Big enough to illustrate the advantages of proposed approach while still being able to run MCMC on full data
- We avoided Polya-Gamma data augmentation due to results in Johndrow et al

Results for logistic regression simulation



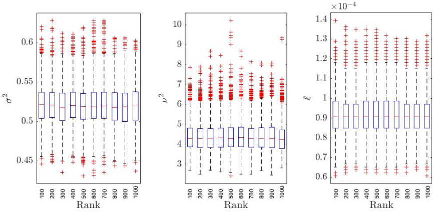
- Generated data from an imbalanced logistic regression model with $N = 10^5$ & $\theta = (-12, 3, 3)$
- Big enough to illustrate the advantages of proposed approach while still being able to run MCMC on full data
- We avoided Polya-Gamma data augmentation due to results in Johndrow et al
- Ran MCMC using $1, 5, 10, 50, 100\%$ of the data with $N(0, 100)$ priors

Gaussian process example



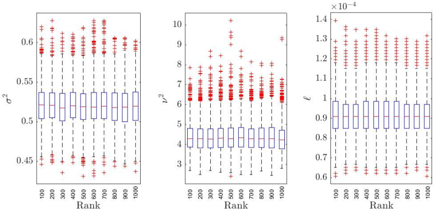
- $Y_i = f(X_i) + \epsilon_i, \quad i = 1, \ldots, N$, with f given a Gaussian process (GP) prior

Gaussian process example



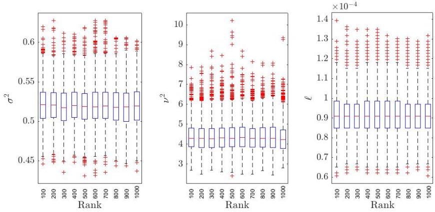
- $Y_i = f(X_i) + \epsilon_i, \quad i = 1, \ldots, N$, with f given a Gaussian process (GP) prior
- Marginalizing out f, obtain $Y|\theta, \sigma^2 \sim N(0, K_\theta + \sigma^2 I)$

Gaussian process example



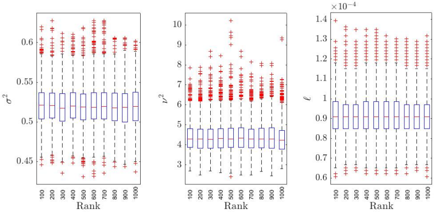
- $Y_i = f(X_i) + \epsilon_i, \quad i = 1, \ldots, N$, with f given a Gaussian process (GP) prior
- Marginalizing out f, obtain $Y|\theta, \sigma^2 \sim N(0, K_\theta + \sigma^2 I)$
- Can run a Metropolis-Hasting algorithm to update covariance parameters but $O(N^3)$ per step

Gaussian process example



- $Y_i = f(X_i) + \epsilon_i, \quad i = 1, \ldots, N$, with f given a Gaussian process (GP) prior
- Marginalizing out f, obtain $Y|\theta, \sigma^2 \sim N(0, K_\theta + \sigma^2 I)$
- Can run a Metropolis-Hasting algorithm to update covariance parameters but $O(N^3)$ per step
- truncated SVD can be used to approximate K_θ & speed this up

Gaussian process example



- $Y_i = f(X_i) + \epsilon_i, \quad i = 1, \ldots, N$, with f given a Gaussian process (GP) prior
- Marginalizing out f, obtain $Y|\theta, \sigma^2 \sim N(0, K_\theta + \sigma^2 I)$
- Can run a Metropolis-Hasting algorithm to update covariance parameters but $O(N^3)$ per step
- truncated SVD can be used to approximate K_θ & speed this up
- To illustrate our approach, we used $N = 1,000$ & ran for ranks of $100, 200, \ldots, 1000$

Outline

Generalized Bayes

- Often it is useful to take a step away from an exactly fully Bayes approach

Generalized Bayes

- Often it is useful to take a step away from an exactly fully Bayes approach
- This can improve robustness to model misspecification & scalability simultaneously

Generalized Bayes

- Often it is useful to take a step away from an exactly fully Bayes approach
- This can improve robustness to model misspecification & scalability simultaneously
- We have found *modularization* particularly useful

Generalized Bayes

- Often it is useful to take a step away from an exactly fully Bayes approach
- This can improve robustness to model misspecification & scalability simultaneously
- We have found *modularization* particularly useful
- Allow the posterior for certain model components to only be informed by part of the data

Generalized Bayes

- Often it is useful to take a step away from an exactly fully Bayes approach
- This can improve robustness to model misspecification & scalability simultaneously
- We have found *modularization* particularly useful
- Allow the posterior for certain model components to only be informed by part of the data
- Example 1: Modular Bayes screening (Chen & Dunson)

Generalized Bayes

- Often it is useful to take a step away from an exactly fully Bayes approach
- This can improve robustness to model misspecification & scalability simultaneously
- We have found *modularization* particularly useful
- Allow the posterior for certain model components to only be informed by part of the data
- Example 1: Modular Bayes screening (Chen & Dunson)
- Example 2: Bayesian mosaic (Wang & Dunson)

Hybrid 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

Hybrid 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

Hybrid 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

Hybrid 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
- Instead use hybrid of Gibbs sampling & fast multiscale SVD

Hybrid 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
- Instead use hybrid of Gibbs sampling & fast multiscale SVD
- Scalable, excellent mixing & empirical/predictive performance

Discussion

- No longer true that MCMC is not scalable

Discussion

- No longer true that MCMC is not scalable
- Often the key computational bottlenecks similar or the same as optimization algorithms

Discussion

- No longer true that MCMC is not scalable
- Often the key computational bottlenecks similar or the same as optimization algorithms
- Vastly smaller community working on innovating MCMC and related sampling algorithms

Discussion

- No longer true that MCMC is not scalable
- Often the key computational bottlenecks similar or the same as optimization algorithms
- Vastly smaller community working on innovating MCMC and related sampling algorithms
- Theory is hard and more work on scaling limits and optimality is needed

Discussion

- No longer true that MCMC is not scalable
- Often the key computational bottlenecks similar or the same as optimization algorithms
- Vastly smaller community working on innovating MCMC and related sampling algorithms
- Theory is hard and more work on scaling limits and optimality is needed
- Certainly MCMC cannot be ruled out & we can can/have applied sampling in huge data problems

Some references

- Chen Y, Dunson DB (2017) Modular Bayes screening for high-dimensional predictors. *arXiv:1703.09906*
- Duan LL, Johndrow JE, Dunson DB (2017) Calibrated data augmentation for scalable Markov chain Monte Carlo. *arXiv:1703.03123*
- Li C, Srivastava S, Dunson D (2017) Simple, scalable and accurate posterior interval estimation. *Biometrika* 104, 665-80.
- Mukhopadhyay M, Dunson D (2017) Targeted random projection for prediction from high-dimensional features. *arXiv:1712.02445*
- Wang Y, Dunson D (2018) Bayesian mosaic: Parallelizable composite posterior. *arXiv:1804.00353*
- Young A, Mattingly J, Dunson D (2018) Accelerating Metropolis-Hastings algorithms through a multiscale view of the posterior distribution. *In progress*.