Scaling Bayesian inference by constructing approximating exponential families

Jonathan H. Huggins
Postdoctoral Research Fellow
Department of Biostatistics, Harvard

With: Ryan Adams, Tamara Broderick, James Zou
Bayesian Inference
Bayesian Inference

Microcredit
Bayesian Inference

Cancer genomics

Figure 4

The contributions of mutational signatures to individual cancers are shown. The horizontal axis represents the number of mutations per sample, and the vertical axis denotes the number of mutations attributed to a specific signature. Different cancer types are represented by different colors. The figure shows how the mutational signatures vary across different cancer types and samples, highlighting the role of Bayesian inference in understanding these variations.

Supplementary Figs 29–58. Summary of the total contributions for all operative signatures across all cancer samples. Contributions across all cancer samples could be found in Tables 1–3. The table shows the total number of mutations for each signature and cancer type, providing a comprehensive overview of the mutational landscapes.

No. mutations per Mb

Sample

Breast

Cervix

Colorectum

Cervix

Lung adenocarcinoma

Melanoma
Bayesian Inference

Microcredit

Cancer genomics

Fuel consumption
• Challenge: existing methods slow (and/or tedious, unreliable)
Our proposal: use Bayesian Inference to create efficient summaries of data. This approach can be particularly useful in cancer genomics, where the efficiency of DNA damage and DNA maintenance processes can change as a consequence of differing carcinogen exposures or after neoplastic transformation with 192 mutation subclasses.

Several signatures showed substantial differences in mutation profiles and many were characterized predominantly by C mutations at ApT. For example, signature 4 displays transcriptional strand bias for C mutations at ApT, which is observed in lung adenocarcinoma and small cell carcinomas, head and neck squamous, and liver cancers. Signature 4 also shows a strong transcriptional strand bias with T mutations on the untranscribed compared to the transcribed strand on which each mutation has taken place.

Because a mutation in a transcribed genomic region may be either on the transcriptional strand on which each mutation has taken place. We re-extracted substitution mutational signatures incorporating the substitution type, the sequence context immediately 5' to the mutated base and whether the mutated pyrimidine is on the transcribed or untranscribed strand. The mutation types are displayed on the horizontal axis, and the mutation counts are shown on the vertical axis.

Other signatures show strong transcriptional strand bias. For instance, signature 7, mainly found in malignant melanoma, shows a similar pattern. Similarly, signature 12, which features TpT trinucleotides, also found in hepatocellular carcinomas, contributes very large numbers of substitutions from signature 3. A subset of cancer cases of these three classes is characterized by TpG and ApT trinucleotides, also found in hepatocellular carcinomas.

Signature 15, found in several samples of lung and stomach cancer, contributes very large numbers of substitutions. Signature 15 also contributes very large numbers of substitutions for breast cancer and its origin is currently unknown. On the assumption that the transcriptional strand biases in signatures and other causes of transcriptional strand bias may exist. Beyond these known examples of DNA damage processed by transcription-coupled NER, other signatures show strong transcriptional strand bias.

The higher prevalence of C mutations at ApT, CpG mutations, but is distinct from signature 3. A higher resolution version of all mutational signatures and other causes of transcriptional strand bias may exist. The mutation types are displayed on the horizontal axis, and the mutation counts are shown on the vertical axis.

The efficiency of DNA damage and DNA maintenance processes can change as a consequence of differing carcinogen exposures or after neoplastic transformation with 192 mutation subclasses. Because a mutation in a transcribed genomic region may be either on the transcriptional strand on which each mutation has taken place.

We re-extracted substitution mutational signatures incorporating the substitution type, the sequence context immediately 5' to the mutated base and whether the mutated pyrimidine is on the transcribed or untranscribed strand. The mutation types are displayed on the horizontal axis, and the mutation counts are shown on the vertical axis.

Other signatures show strong transcriptional strand bias. For example, signature 4 displays transcriptional strand bias for C mutations at ApT, which is observed in lung adenocarcinoma and small cell carcinomas, head and neck squamous, and liver cancers. Signature 4 also shows a strong transcriptional strand bias with T mutations on the untranscribed compared to the transcribed strand on which each mutation has taken place.

Because a mutation in a transcribed genomic region may be either on the transcriptional strand on which each mutation has taken place. We re-extracted substitution mutational signatures incorporating the substitution type, the sequence context immediately 5' to the mutated base and whether the mutated pyrimidine is on the transcribed or untranscribed strand. The mutation types are displayed on the horizontal axis, and the mutation counts are shown on the vertical axis.

Other signatures show strong transcriptional strand bias. For instance, signature 7, mainly found in malignant melanoma, shows a similar pattern. Similarly, signature 12, which features TpT trinucleotides, also found in hepatocellular carcinomas, contributes very large numbers of substitutions from signature 3. A subset of cancer cases of these three classes is characterized by TpG and ApT trinucleotides, also found in hepatocellular carcinomas.

Signature 15, found in several samples of lung and stomach cancer, contributes very large numbers of substitutions. Signature 15 also contributes very large numbers of substitutions for breast cancer and its origin is currently unknown. On the assumption that the transcriptional strand biases in signatures and other causes of transcriptional strand bias may exist. Beyond these known examples of DNA damage processed by transcription-coupled NER, other signatures show strong transcriptional strand bias.
Bayesian Inference

• Challenge: existing methods slow (and/or tedious, unreliable)

• Our proposal: use efficient summaries of data

• Approximate sufficient statistics for simple, scalable Bayesian inference with error bounds for finite data
Roadmap

- Approximate Bayes review
- Likelihood approximation and dataset compression
- Approximate sufficient statistics
- Accuracy guarantees
Roadmap

• Approximate Bayes review

• Likelihood approximation and dataset compression

• Approximate sufficient statistics

• Accuracy guarantees
Bayesian inference is challenging...

$$\pi(\theta|Y) = \frac{p(Y|\theta)\pi_0(\theta)}{Z}$$
Bayesian inference is challenging...

\[ \pi(\theta|Y) = \frac{p(Y|\theta)\pi_0(\theta)}{Z} \]
Bayesian inference is challenging...

\[ \pi(\theta|Y) = \frac{p(Y|\theta)\pi_0(\theta)}{Z} \]

- calculate...
Bayesian inference is challenging...

\[
\pi(\theta|Y) = \frac{p(Y|\theta) \pi_0(\theta)}{Z}
\]

- calculate...
  - mean(\(\theta_{20}|Y\))
Bayesian inference is challenging...

\[
\pi(\theta|Y) = \frac{p(Y|\theta)\pi_0(\theta)}{Z}
\]

• calculate...
  • mean(\(\theta_{20} | Y\))
  • var(\(\theta_{20} | Y\))
Bayesian inference is challenging…

\[
\pi(\theta | Y) = \frac{p(Y | \theta) \pi_0(\theta)}{Z}
\]

- calculate…
  - \text{mean}(\theta_{20} | Y)
  - \text{var}(\theta_{20} | Y)
  - \Pr[\theta_{20} < .1 | Y]
Bayesian inference is challenging...

\[ \pi(\theta|Y) = \frac{p(Y|\theta)p_0(\theta)}{Z} \]

- calculate...
  - mean(\(\theta_20 \ | \ Y\))
  - var(\(\theta_20 \ | \ Y\))
  - Pr[\(\theta_20 < .1 \ | \ Y\)]

- **Goal:** compute expectations wrt \(\pi\)
Bayesian inference is challenging... 

\[ \pi(\theta|Y) = \frac{p(Y|\theta)\pi_0(\theta)}{Z} \]

- calculate...
  - \( \text{mean}(\theta_{20} | Y) \)
  - \( \text{var}(\theta_{20} | Y) \)
  - \( \text{Pr}[\theta_{20} < .1 | Y] \)

- **Goal:** compute expectations wrt \( \pi \)

- A hard problem!
Bayesian inference is challenging...

\[
\pi(\theta | Y) = \frac{p(Y|\theta)\pi_0(\theta)}{Z}
\]

• calculate...
  • mean(\(\theta_20 | Y\))
  • var(\(\theta_20 | Y\))
  • Pr[\(\theta_20 < .1 | Y\)]

• Goal: compute expectations wrt \(\pi\)

• A hard problem!
Bayesian inference is challenging…

\[ \pi(\theta|Y) = \frac{p(Y|\theta)\pi_0(\theta)}{Z} \]

- calculate…
  - mean(\(\theta_{20} | Y\))
  - var(\(\theta_{20} | Y\))
  - Pr[\(\theta_{20} < .1 | Y\)]

- **Goal:** compute expectations wrt \(\pi\)

- A hard problem!
Bayesian inference is challenging...

\[
\pi(\theta|Y) = \frac{p(Y|\theta)\pi_0(\theta)}{Z}
\]

- calculate...
  - mean(\(\theta_{20} | Y\))
  - var(\(\theta_{20} | Y\))
  - Pr[\(\theta_{20} < .1 | Y\)]
- **Goal**: compute expectations wrt \(\pi\)
- A hard problem!
Bayesian inference is challenging…

\[ \pi(\theta|Y) = \frac{p(Y|\theta)\pi_0(\theta)}{Z} \]

- calculate…
  - mean(\(\theta_{20} \mid Y\))
  - var(\(\theta_{20} \mid Y\))
  - Pr[\(\theta_{20} < .1 \mid Y\)]

- **Goal:** compute expectations wrt \(\pi\)

- **A hard problem!**

- **Solution:** approximate \(\pi\)
What do we want from an approximate inference algorithm?
What do we want from an approximate inference algorithm?

1. Scalability
What do we want from an approximate inference algorithm?

1. Scalability

   • large datasets
What do we want from an approximate inference algorithm?

1. Scalability
   - large datasets
   - streaming and distributed data
What do we want from an approximate inference algorithm?

1. Scalability
   - large datasets
   - streaming and distributed data
   - moderate-sized data with complex models
What do we want from an approximate inference algorithm?

1. Scalability
   - large datasets
   - streaming and distributed data
   - moderate-sized data with complex models

2. Arbitrary accuracy: $d(\pi, \pi_{\text{approx}}) \to 0$
What do we want from an approximate inference algorithm?

1. Scalability
   - large datasets
   - streaming and distributed data
   - moderate-sized data with complex models

2. Arbitrary accuracy:  \( d(\pi, \pi_{\text{approx}}) \rightarrow 0 \)

3. Validation of approximation quality:  \( d(\pi, \pi_{\text{approx}}) < c \)
Modern Bayesian inference

1. Scalability

2. Arbitrary accuracy

3. Validation of approximation quality
Modern Bayesian inference

1. Scalability

2. Arbitrary accuracy

3. Validation of approximation quality
Modern Bayesian inference

1. Scalability

2. Arbitrary accuracy

3. Validation of approximation quality

Markov chain Monte Carlo
Modern Bayesian inference

1. Scalability
   - large datasets: demand prediction
   - streaming and distributed data: web-scale data
   - moderate-sized data with complex models: econ, genomics

2. Arbitrary accuracy

3. Validation of approximation quality
   - a priori: finite-time, finite-data guarantees
   - post hoc: quality measures

Modern Bayesian inference includes techniques such as Markov chain Monte Carlo (MCMC) and subsampling MCMC.
Modern Bayesian inference

1. Scalability
   - large datasets: demand prediction
   - streaming and distributed data: web-scale data
   - moderate-sized data with complex models: econ, genomics

2. Arbitrary accuracy

3. Validation of approximation quality
   - a priori: finite-time, finite-data guarantees
   - post hoc: quality measures

Modern Bayesian inference
- Markov chain Monte Carlo
- subsampling MCMC
Modern Bayesian inference

1. Scalability
   - large datasets: demand prediction
   - streaming and distributed data: web-scale data
   - moderate-sized data with complex models: econ, genomics

2. Arbitrary accuracy

3. Validation of approximation quality
   - a priori finite-time, finite-data guarantees
   - post hoc quality measures

Modern Bayesian inference techniques:
- Markov chain Monte Carlo
- subsampling MCMC
- variational Bayes
Modern Bayesian inference

1. Scalability
   - large datasets: demand prediction
   - streaming and distributed data: web-scale data
   - moderate-sized data with complex models: econ, genomics

2. Arbitrary accuracy

3. Validation of approximation quality
   - a priori finite-time, finite-data guarantees
   - post hoc quality measures

Markov chain Monte Carlo
subsampling MCMC
variational Bayes
Modern Bayesian inference

1. Scalability
   - large datasets: demand prediction
   - streaming and distributed data: web-scale data
   - moderate-sized data with complex models: econ, genomics
   - Markov chain Monte Carlo
   - subsampling MCMC

2. Arbitrary accuracy
   - variational Bayes

3. Validation of approximation quality
   - consensus methods
1. Scalability
   - large datasets: demand prediction
   - streaming and distributed data: web-scale data
   - moderate-sized data with complex models: econ, genomics

2. Arbitrary accuracy

3. Validation of approximation quality
   - a priori finite-time, finite-data guarantees
   - post hoc quality measures

Modern Bayesian inference

- Markov chain Monte Carlo
- subsampling MCMC
- variational Bayes
- consensus methods
Modern Bayesian inference

1. Scalability
   - large datasets: demand prediction
   - streaming and distributed data: web-scale data
   - moderate-sized data with complex models: econ, genomics

2. Arbitrary accuracy

3. Validation of approximation quality
   - a priori: finite-time, finite-data guarantees
   - post hoc: quality measures

Modern Bayesian inference methods:
- Markov chain Monte Carlo
- Subsampling MCMC
- Variational Bayes
- Consensus methods

This talk: we get all three!
Roadmap

• Approximate Bayes review

• Likelihood approximation and dataset compression

• Approximate sufficient statistics

• Accuracy guarantees
Key idea: likelihood approximation
Key idea: likelihood approximation

Monte Carlo
Key idea: likelihood approximation

Monte Carlo

$$
\mathbb{E}[f(\theta) \mid Y] \approx T^{-1} \sum_{t=1}^{T} f(\theta_t)
$$
Key idea: likelihood approximation

Monte Carlo

\[ \mathbb{E}[f(\theta) \mid Y] \approx T^{-1} \sum_{t=1}^{T} f(\theta_t) \quad \theta_t \overset{\text{i.i.d.}}{\sim} \pi(\theta \mid Y) \]
Key idea: likelihood approximation

Monte Carlo

\[ \mathbb{E}[f(\theta) \mid Y] \approx T^{-1} \sum_{t=1}^{T} f(\theta_t) \quad \theta_t \overset{\text{i.i.d.}}{\sim} \pi(\theta \mid Y) \]

Markov chain Monte Carlo (MCMC)
Key idea: likelihood approximation

Monte Carlo

\[ \mathbb{E}[f(\theta) \mid Y] \approx T^{-1} \sum_{t=1}^{T} f(\theta_t) \quad \theta_t \overset{\text{i.i.d.}}{\sim} \pi(\theta \mid Y) \]

Markov chain Monte Carlo (MCMC)

\[ \theta_t \text{ not independent} \]
Key idea: likelihood approximation

Monte Carlo
\[ \mathbb{E}[f(\theta) \mid Y] \approx T^{-1} \sum_{t=1}^{T} f(\theta_t) \]
\[ \theta_t \overset{\text{i.i.d.}}{\sim} \pi(\theta \mid Y) \]

Markov chain Monte Carlo (MCMC)
\[ \theta_t \text{ not independent} \]

\[ \theta_0 \]
Key idea: likelihood approximation

Monte Carlo

\[ \mathbb{E}[f(\theta) \mid Y] \approx T^{-1} \sum_{t=1}^{T} f(\theta_t) \quad \theta_t \overset{\text{i.i.d.}}{\sim} \pi(\theta \mid Y) \]

Markov chain Monte Carlo (MCMC)

\[ \theta_t \text{ not independent} \]
Key idea: likelihood approximation

Monte Carlo

\[ \mathbb{E}[f(\theta) \mid Y] \approx T^{-1} \sum_{t=1}^{T} f(\theta_t) \quad \theta_t \overset{\text{i.i.d.}}{\sim} \pi(\theta \mid Y) \]

Markov chain Monte Carlo (MCMC)

\[ \theta_t \text{ not independent} \]

evaluate \( \log p(Y \mid \theta)\pi_0(\theta) \) at \( \theta_0 \) and \( \theta'_1 \)
Key idea: likelihood approximation

Monte Carlo

\[ E[f(\theta) | Y] \approx T^{-1} \sum_{t=1}^{T} f(\theta_t) \quad \theta_t \overset{\text{i.i.d.}}{\sim} \pi(\theta | Y) \]

Markov chain Monte Carlo (MCMC)

\(\theta_1\) evaluate \(\log p(Y | \theta)\pi_0(\theta)\) at \(\theta_0\) and \(\theta_1\)
Key idea: likelihood approximation

Monte Carlo

\[ \mathbb{E}[f(\theta) \mid Y] \approx T^{-1} \sum_{t=1}^{T} f(\theta_t) \quad \theta_t \sim \pi(\theta \mid Y) \]

Markov chain Monte Carlo (MCMC)

\[ \theta_t \text{ not independent} \]
Key idea: likelihood approximation

Monte Carlo

\[ \mathbb{E}[f(\theta) \mid Y] \approx T^{-1} \sum_{t=1}^{T} f(\theta_t) \quad \theta_t \overset{\text{i.i.d.}}{\sim} \pi(\theta \mid Y) \]

Markov chain Monte Carlo (MCMC)

\[ \theta_t \text{ not independent} \]

\[ \theta_0 \rightarrow \theta_1 \rightarrow \theta_2' \]

Evaluate \( \log p(Y \mid \theta)\pi_0(\theta) \) at \( \theta_1 \) and \( \theta_2' \).
Key idea: likelihood approximation

\[ \mathbb{E}[f(\theta) \mid Y] \approx T^{-1} \sum_{t=1}^{T} f(\theta_t) \quad \theta_t \overset{\text{i.i.d.}}{\sim} \pi(\theta \mid Y) \]

\[ \theta_t \text{ not independent} \]

Monte Carlo

Markov chain Monte Carlo (MCMC)

Evaluate \( \log p(Y \mid \theta) \pi_0(\theta) \) at \( \theta_1 \) and \( \theta_2 \)
Key idea: likelihood approximation

Monte Carlo

\[ \mathbb{E}[f(\theta) \mid Y] \approx T^{-1} \sum_{t=1}^{T} f(\theta_t) \quad \theta_t \overset{i.i.d.}{\sim} \pi(\theta \mid Y) \]

Markov chain Monte Carlo (MCMC)

\[ \theta_t \text{ not independent} \]
Key idea: likelihood approximation

Monte Carlo

$$\mathbb{E}[f(\theta) \mid Y] \approx T^{-1} \sum_{t=1}^{T} f(\theta_t) \quad \theta_t \overset{\text{i.i.d.}}{\sim} \pi(\theta \mid Y)$$

Markov chain Monte Carlo (MCMC)

\[ \theta_t \text{ not independent} \]

evaluate \( \log p(Y \mid \theta) \pi_0(\theta) \) at \( \theta_2 \) and \( \theta_3' \)
Key idea: likelihood approximation

Monte Carlo
\[ \mathbb{E}[f(\theta) \mid Y] \approx T^{-1} \sum_{t=1}^{T} f(\theta_t) \quad \theta_t \overset{\text{i.i.d.}}{\sim} \pi(\theta \mid Y) \]

Markov chain Monte Carlo (MCMC)
\[ \theta_t \text{ not independent} \]

evaluate \( \log p(Y \mid \theta)\pi_0(\theta) \) at \( \theta_2 \) and \( \theta'_3 \)
Key idea: likelihood approximation

Monte Carlo

\[ \mathbb{E}[f(\theta) \mid Y] \approx T^{-1} \sum_{t=1}^{T} f(\theta_t) \quad \theta_t \overset{\text{i.i.d.}}{\sim} \pi(\theta \mid Y) \]

Markov chain Monte Carlo (MCMC)

\[ \theta_t \text{ not independent} \]
Key idea: likelihood approximation

Monte Carlo
\[
\mathbb{E}[f(\theta) \mid Y] \approx T^{-1} \sum_{t=1}^{T} f(\theta_t) \quad \theta_t \overset{i.i.d.}{\sim} \pi(\theta \mid Y)
\]

Markov chain Monte Carlo (MCMC)
\[
\theta_t \text{ not independent}
\]
\[
Y = \{y_1, y_2, \ldots, y_N\}
\]
Key idea: likelihood approximation

Monte Carlo

\[ E[f(\theta) \mid Y] \approx T^{-1} \sum_{t=1}^{T} f(\theta_t) \quad \theta_t \overset{\text{i.i.d.}}{\sim} \pi(\theta \mid Y) \]

Markov chain Monte Carlo (MCMC)

\[ \theta_t \text{ not independent} \]

\[ Y = \{y_1, y_2, \ldots, y_N\} \]

Problem: \( \log p(Y \mid \theta) \) is expensive to evaluate if \( N \) is large
Key idea: likelihood approximation

Monte Carlo

\[ \mathbb{E}[f(\theta) \mid Y] \approx T^{-1} \sum_{t=1}^{T} f(\theta_t) \quad \theta_t \overset{i.i.d.}{\sim} \pi(\theta \mid Y) \]

Markov chain Monte Carlo (MCMC)

\[ Y = \{y_1, y_2, \ldots, y_N\} \]

Problem: \( \log p(Y \mid \theta) \) is expensive to evaluate if \( N \) is large

\( \Omega(N \times T) \) time
Key idea: likelihood approximation

Monte Carlo

\[
\mathbb{E}[f(\theta) \mid Y] \approx T^{-1} \sum_{t=1}^{T} f(\theta_t) \quad \theta_t \overset{\text{i.i.d.}}{\sim} \pi(\theta \mid Y)
\]

Markov chain Monte Carlo (MCMC)

\[
\theta_t \, \text{not independent}
\]

\[
Y = \{y_1, y_2, \ldots, y_N\}
\]

Problem: \(\log p(Y \mid \theta)\) is expensive to evaluate if \(N\) is large

Solution:

\(\Omega(N \times T)\) time
Key idea: likelihood approximation

Monte Carlo

$$\mathbb{E}[f(\theta) \mid Y] \approx T^{-1} \sum_{t=1}^{T} f(\theta_t) \quad \theta_t \stackrel{i.i.d.}{\sim} \pi(\theta \mid Y)$$

Markov chain Monte Carlo (MCMC)

$$\theta_t \text{ not independent}$$

$$Y = \{y_1, y_2, \ldots, y_N\}$$

Problem: $\log p(Y \mid \theta)$ is expensive to evaluate if $N$ is large

Solution:

1. replace $\log p(Y \mid \theta)$ with a fast-to-compute proxy $\ell(\theta, g(Y))$
Key idea: likelihood approximation

Monte Carlo

\[ \mathbb{E}[f(\theta) \mid Y] \approx T^{-1} \sum_{t=1}^{T} f(\theta_t) \quad \theta_t \overset{i.i.d.}{\sim} \pi(\theta \mid Y) \]

Markov chain Monte Carlo (MCMC)

\[ \theta_t \text{ not independent} \]

\[ Y = \{y_1, y_2, \ldots, y_N\} \]

Problem: \( \log p(Y \mid \theta) \) is expensive to evaluate if \( N \) is large

Solution:

1. replace \( \log p(Y \mid \theta) \) with a fast-to-compute proxy \( \ell(\theta, g(Y)) \)

short summary of \( Y \)
Key idea: likelihood approximation

Monte Carlo

\[ \mathbb{E}[f(\theta) \mid Y] \approx T^{-1} \sum_{t=1}^{T} f(\theta_t) \quad \theta_t \overset{i.i.d.}{\sim} \pi(\theta \mid Y) \]

Markov chain Monte Carlo (MCMC)

\[ \theta_t \text{ not independent} \]

\[ Y = \{ y_1, y_2, \ldots, y_N \} \]

Problem: \( \log p(Y \mid \theta) \) is expensive to evaluate if \( N \) is large

Solution:

1. replace \( \log p(Y \mid \theta) \) with a fast-to-compute proxy \( \ell(\theta, g(Y)) \)
2. choose \( \ell(\theta, g(Y)) \) so approximate posterior is accurate

\( \Omega(N \times T) \) time
Ideal case: exponential family

\[ Y = \{ y_1, y_2, \ldots, y_N \} \]
Ideal case: exponential family

\[ Y = \{y_1, y_2, \ldots, y_N\} \]

\[ \log p(y_n|\theta) = \eta(\theta) \cdot \tau(y_n) \]
Ideal case: exponential family

\[ Y = \{y_1, y_2, \ldots, y_N\} \]

\[ \log p(y_n | \theta) = \eta(\theta) \cdot \tau(y_n) \]

reparameterization
Ideal case: exponential family

\[ Y = \{y_1, y_2, \ldots, y_N\} \]

\[ \log p(y_n|\theta) = \eta(\theta) \cdot \tau(y_n) \]

reparameterization  sufficient statistics
Ideal case: exponential family

\[ Y = \{y_1, y_2, \ldots, y_N\} \]

\[ \sum_{n=1}^{N} \log p(y_n | \theta) = \eta(\theta) \cdot \sum_{n=1}^{N} \tau(y_n) \]
Ideal case: exponential family

\[ Y = \{y_1, y_2, \ldots, y_N\} \]

\[
\log p(Y|\theta) = \sum_{n=1}^{N} \log p(y_n|\theta) = \eta(\theta) \cdot \sum_{n=1}^{N} \tau(y_n)
\]
Ideal case: exponential family

\[ Y = \{y_1, y_2, \ldots, y_N\} \]

\[ \log p(Y|\theta) = \sum_{n=1}^{N} \log p(y_n|\theta) = \eta(\theta) \cdot \sum_{n=1}^{N} \tau(y_n) \]

\[ g(Y) \]
Ideal case: exponential family

\[ Y = \{y_1, y_2, \ldots, y_N\} \]

\[
\log p(Y|\theta) = \sum_{n=1}^{N} \log p(y_n|\theta) = \eta(\theta) \cdot \sum_{n=1}^{N} \tau(y_n) g(Y)
\]

Run MCMC for \( T \) iterations:
Ideal case: exponential family

\[ Y = \{y_1, y_2, \ldots, y_N\} \]

\[
\log p(Y | \theta) = \sum_{n=1}^{N} \log p(y_n | \theta) = \eta(\theta) \cdot \sum_{n=1}^{N} \tau(y_n) \]

Run MCMC for \( T \) iterations:

- Naively: \( \Omega(N \times T) \) time
Ideal case: exponential family

\[ Y = \{y_1, y_2, \ldots, y_N\} \]

\[
\log p(Y | \theta) = \sum_{n=1}^{N} \log p(y_n | \theta) = \eta(\theta) \cdot \sum_{n=1}^{N} \tau(y_n) \]

Run MCMC for \( T \) iterations:

- Naively: \( \Omega(N \times T) \) time

\( \log p(Y | \theta) \) takes \( \Omega(N) \) time to evaluate
Ideal case: exponential family

\[ Y = \{y_1, y_2, \ldots, y_N\} \]

\[
\log p(Y|\theta) = \sum_{n=1}^{N} \log p(y_n|\theta) = \eta(\theta) \cdot \sum_{n=1}^{N} \tau(y_n) \]

Run MCMC for \( T \) iterations:

- Naively: \( \Omega(N \times T) \) time to evaluate

\[
\text{log } p(Y|\theta) \text{ takes } \Omega(N) \text{ time to evaluate} \]
Ideal case: exponential family

\[ Y = \{y_1, y_2, \ldots, y_N\} \]

\[ \log p(Y|\theta) = \sum_{n=1}^{N} \log p(y_n|\theta) = \eta(\theta) \cdot \sum_{n=1}^{N} \tau(y_n) g(Y) \]

Run MCMC for \( T \) iterations:

- Naively: \( \Omega(N \times T) \) time
- Using EF structure: \( O(N + T) \) time

\[ \log p(Y|\theta) \text{ takes } \Omega(N) \text{ time to evaluate} \]
Ideal case: exponential family

\[ Y = \{y_1, y_2, \ldots, y_N\} \]

\[
\log p(Y|\theta) = \sum_{n=1}^{N} \log p(y_n|\theta) = \eta(\theta) \cdot \sum_{n=1}^{N} \tau(y_n) \]

Run MCMC for \( T \) iterations:

- Naively: \( \Omega(N \times T) \) time
- Using EF structure: \( O(N + T) \) time

\[ \log p(Y|\theta) \text{ takes } \Omega(N) \text{ time to evaluate} \]
Ideal case: exponential family

\[ Y = \{y_1, y_2, \ldots, y_N\} \]

\[
\log p(Y|\theta) = \sum_{n=1}^{N} \log p(y_n|\theta) = \eta(\theta) \cdot \sum_{n=1}^{N} \tau(y_n) \]

Run MCMC for \( T \) iterations:

- Naively: \( \Omega(N \times T) \) time

- Using EF structure: \( O(N + T) \) time

\( \log p(Y|\theta) \) takes \( \Omega(N) \) time to evaluate

\( \log p(Y|\theta) \) takes \( O(1) \) time to evaluate

\( g(Y) \) is computed
Ideal case: exponential family

\[ Y = \{y_1, y_2, \ldots, y_N\} \]

\[
\log p(Y | \theta) = \sum_{n=1}^{N} \log p(y_n | \theta) = \eta(\theta) \cdot \sum_{n=1}^{N} \tau(y_n) \]

Run MCMC for \( T \) iterations:

- Naively: \( \Omega(N \times T) \) time

- Using EF structure: \( O(N + T) \) time

\[
\log p(Y | \theta) \text{ takes } \Omega(N) \text{ time to evaluate}
\]

\[
\log p(Y | \theta) \text{ takes } O(1) \text{ time to evaluate}
\]

\[
\text{compute } g(Y)
\]
Streaming and distributed exponential family inference

\[ Y = \{ y_1, y_2, \ldots, y_N \} \]

\[
\log p(Y | \theta) = \sum_{n=1}^{N} \log p(y_n | \theta) = \eta(\theta) \cdot \sum_{n=1}^{N} \tau(y_n)
\]
Streaming and distributed exponential family inference

\[ Y = \{ y_1, y_2, \ldots, y_N \} \]

\[ \log p(Y|\theta) = \sum_{n=1}^{N} \log p(y_n|\theta) = \eta(\theta) \cdot \sum_{n=1}^{N} \tau(y_n) \]
Streaming and distributed exponential family inference

\[ Y = \{y_1, y_2, \ldots, y_N\} \]

\[ \log p(Y | \theta) = \sum_{n=1}^{N} \log p(y_n | \theta) = \eta(\theta) \cdot \sum_{n=1}^{N} \tau(y_n) \]

**Streaming**

\[ \tau = 0 \]
Streaming and distributed exponential family inference

\[ Y = \{y_1, y_2, \ldots, y_N\} \]

\[
\log p(Y|\theta) = \sum_{n=1}^{N} \log p(y_n|\theta) = \eta(\theta) \cdot \sum_{n=1}^{N} \tau(y_n)
\]

**Streaming**

\( \tau = 0 \)

**for** \( n = 1, \ldots, N \)
Streaming and distributed exponential family inference

\[ Y = \{y_1, y_2, \ldots, y_N\} \]

\[ \log p(Y|\theta) = \sum_{n=1}^{N} \log p(y_n|\theta) = \eta(\theta) \cdot \sum_{n=1}^{N} \tau(y_n) \]

**Streaming**

\[ \tau = 0 \]

**for** \( n = 1, \ldots, N \)

\[ \tau = \tau + \tau(y_n) \]
Streaming and distributed exponential family inference

\( Y = \{y_1, y_2, \ldots, y_N\} \)

\[
\log p(Y|\theta) = \sum_{n=1}^{N} \log p(y_n|\theta) = \eta(\theta) \cdot \sum_{n=1}^{N} \tau(y_n)
\]

**STREAMING**

\( \tau = 0 \)

for \( n = 1, \ldots, N \)

\[
\tau = \tau + \tau(y_n)
\]

construct \( \log p(Y|\theta) \) using \( \tau \)
Streaming and distributed exponential family inference

\[ Y = \{y_1, y_2, \ldots, y_N\} \]

\[ \log p(Y|\theta) = \sum_{n=1}^{N} \log p(y_n|\theta) = \eta(\theta) \cdot \sum_{n=1}^{N} \tau(y_n) \]

**STREAMING**

\[ \tau = 0 \]

**for** \( n = 1, \ldots, N \)

\[ \tau = \tau + \tau(y_n) \]

construct \( \log p(Y|\theta) \) using \( \tau \)

run MCMC
Streaming and distributed exponential family inference

\[ Y = \{y_1, y_2, \ldots, y_N\} \]

\[
\log p(Y | \theta) = \sum_{n=1}^{N} \log p(y_n | \theta) = \eta(\theta) \cdot \sum_{n=1}^{N} \tau(y_n)
\]

**Streaming**

\[ \tau = 0 \]

\[ \text{for } n = 1, \ldots, N \]

\[ \tau = \tau + \tau(y_n) \]

construct \( \log p(Y | \theta) \) using \( \tau \)

run MCMC

**Distributed**
Streaming and distributed exponential family inference

$$Y = \{y_1, y_2, \ldots, y_N\}$$

$$\log p(Y|\theta) = \sum_{n=1}^{N} \log p(y_n|\theta) = \eta(\theta) \cdot \sum_{n=1}^{N} \tau(y_n)$$

**STREAMING**

$$\tau = 0$$

for $$n = 1, \ldots, N$$

$$\tau = \tau + \tau(y_n)$$

construct $$\log p(Y|\theta)$$ using $$\tau$$

run MCMC

**DISTRIBUTED**

for $$b = 1, \ldots, B$$ in parallel
Streaming and distributed exponential family inference

\[ Y = \{y_1, y_2, \ldots, y_N\} \]

\[ \log p(Y|\theta) = \sum_{n=1}^{N} \log p(y_n|\theta) = \eta(\theta) \cdot \sum_{n=1}^{N} \tau(y_n) \]

**STREAMING**
\[ \tau = 0 \]
for \( n = 1, \ldots, N \)
\[ \tau = \tau + \tau(y_n) \]
construct \( \log p(Y|\theta) \) using \( \tau \)
run MCMC

**DISTRIBUTED**
for \( b = 1, \ldots, B \) in parallel
1. worker \( b \) reads data subset
\[ Y_b = \{ y_{B(b-1)/n}, \ldots, y_{Bb/n-1} \} \]
Streaming and distributed exponential family inference

\[ Y = \{y_1, y_2, \ldots, y_N\} \]

\[
\log p(Y|\theta) = \sum_{n=1}^{N} \log p(y_n|\theta) = \eta(\theta) \cdot \sum_{n=1}^{N} \tau(y_n)
\]

**STREAMING**

\[ \tau = 0 \]

**for** \( n = 1, \ldots, N \)

\[ \tau = \tau + \tau(y_n) \]

construct \( \log p(Y|\theta) \) using \( \tau \)

run MCMC

**DISTRIBUTED**

**for** \( b = 1, \ldots, B \) in parallel

1. worker \( b \) reads data subset

\[ Y_b = \{ y_{B(b-1)/n}, \ldots, y_{Bb/n-1} \} \]

2. worker \( b \) computes \( \tau_b = \tau(Y_b) \)
Streaming and distributed exponential family inference

\[ Y = \{y_1, y_2, \ldots, y_N\} \]

\[
\log p(Y|\theta) = \sum_{n=1}^{N} \log p(y_n|\theta) = \eta(\theta) \cdot \sum_{n=1}^{N} \tau(y_n)
\]

**STREAMING**

\[
\tau = 0
\]

**for** \( n = 1, \ldots, N \)

\[
\tau = \tau + \tau(y_n)
\]

construct \( \log p(Y|\theta) \) using \( \tau \)

run MCMC

**DISTRIBUTED**

**for** \( b = 1, \ldots, B \) **in parallel**

1. worker \( b \) reads data subset

\[
Y_b = \{ y_{B(b-1)/n}, \ldots, y_{Bb/n-1} \}
\]

2. worker \( b \) computes \( \tau_b = \tau(Y_b) \)

\[
\tau = \tau_1 + \cdots + \tau_b
\]
Streaming and distributed exponential family inference

\[ Y = \{y_1, y_2, \ldots, y_N\} \]

\[ \log p(Y|\theta) = \sum_{n=1}^{N} \log p(y_n|\theta) = \eta(\theta) \cdot \sum_{n=1}^{N} \tau(y_n) \]

**Streaming**

\( \tau = 0 \)

**for** \( n = 1, \ldots, N \)

\( \tau = \tau + \tau(y_n) \)

**construct** \( \log p(Y|\theta) \) **using** \( \tau \)

**run** MCMC

**Distributed**

**for** \( b = 1, \ldots, B \) **in parallel**

1. **worker** \( b \) **reads data subset**

\[ Y_b = \{y_{B(b-1)/n}, \ldots, y_{Bb/n-1}\} \]

2. **worker** \( b \) **computes** \( \tau_b = \tau(Y_b) \)

\( \tau = \tau_1 + \cdots + \tau_b \)

**construct** \( \log p(Y|\theta) \) **using** \( \tau \)
Streaming and distributed exponential family inference

\[ Y = \{y_1, y_2, \ldots, y_N\} \]

\[ \log p(Y|\theta) = \sum_{n=1}^{N} \log p(y_n|\theta) = \eta(\theta) \cdot \sum_{n=1}^{N} \tau(y_n) \]

**STREAMING**

\[ \tau = 0 \]

**for** \( n = 1, \ldots, N \)

\[ \tau = \tau + \tau(y_n) \]

**construct** \( \log p(Y|\theta) \) **using** \( \tau \)

**run MCMC**

**DISTRIBUTED**

**for** \( b = 1, \ldots, B \) **in parallel**

1. worker \( b \) **reads** data subset

\[ Y_b = \{ y_{B(b-1)/n}, \ldots, y_{Bb/n-1} \} \]

2. worker \( b \) **computes** \( \tau_b = \tau(Y_b) \)

\[ \tau = \tau_1 + \cdots + \tau_b \]

**construct** \( \log p(Y|\theta) \) **using** \( \tau \)

**run MCMC**
Streaming and distributed exponential family inference

\[ \tau = 0 \text{ for } n = 1, \ldots, N \]

\[ \tau = \tau + \tau(y_n) \]

construct \( \log p(Y|\theta) \) using \( \tau \)

run MCMC

\[ Y = \{y_1, y_2, \ldots, y_N\} \]

\[ \log p(Y|\theta) = \prod_{n=1}^{N} \log p(y_n|\theta) = \cdot \prod_{n=1}^{N} \cdot \sum_{y_n} \]

DISTRIBUTED

for \( b = 1, \ldots, B \) in parallel

1. worker \( b \) reads data subset \( Y_b = \{y_{(b-1)/n}, \ldots, y_{b/n-1}\} \)

2. worker \( b \) computes \( \tau_b = \tau(Y_b) \)

\[ \tau = \tau_1 + \ldots + \tau_b \]

construct \( \log p(Y|\theta) \) using \( \tau \)

run MCMC
Roadmap

• Approximate Bayes review

• Likelihood approximation and dataset compression

• Approximate sufficient statistics

• Accuracy guarantees
Polynomials are good for approximation

\[-5.8x^6 + 14.9x^4 - 10.1x^2 + 1.1\]
Polynomials are good for approximation

\[-5.8x^6 + 14.9x^4 - 10.1x^2 + 1.1\]

1. Computationally convenient
Polynomials are good for approximation

\[-5.8x^6 + 14.9x^4 - 10.1x^2 + 1.1\]

1. Computationally convenient

2. Can approximate any smooth function
Polynomials are good for approximation

\[-5.8x^6 + 14.9x^4 - 10.1x^2 + 1.1\]

1. Computationally convenient

2. Can approximate any smooth function

3. Approximation properties are well-understood
Polynomials are good for approximation

\[-5.8x^6 + 14.9x^4 - 10.1x^2 + 1.1\]

1. Computationally convenient
2. Can approximate any smooth function
3. Approximation properties are well-understood
Polynomials are good for approximation

\[-5.8x^6 + 14.9x^4 - 10.1x^2 + 1.1\]

1. Computationally convenient
2. Can approximate any smooth function
3. Approximation properties are well-understood
Polynomial approximate sufficient statistics (PASS)
Polynomial approximate sufficient statistics (PASS)

\[ Y = \{y_1, y_2, \ldots, y_N\} \]
Polynomial approximate sufficient statistics (PASS)

\[ Y = \{y_1, y_2, \ldots, y_N\} \]
\[ X = \{x_1, x_2, \ldots, x_N\} \]
Polynomial approximate sufficient statistics (PASS)

\[ Y = \{y_1, y_2, \ldots, y_N\} \]
\[ X = \{x_1, x_2, \ldots, x_N\} \]
\[ x_n = (x_{n1}, x_{n2}, \ldots, x_{nd}) \]
Polynomial approximate sufficient statistics (PASS)

\[ Y = \{y_1, y_2, \ldots, y_N\} \]
\[ X = \{x_1, x_2, \ldots, x_N\} \]
\[ x_n = (x_{n1}, x_{n2}, \ldots, x_{nd}) \]
\[ \log p(y_n|x_n, \theta) \approx \eta(\theta) \cdot \tau(y_n, x_n) \]
Polynomial approximate sufficient statistics (PASS)

\[ Y = \{ y_1, y_2, \ldots, y_N \} \]
\[ X = \{ x_1, x_2, \ldots, x_N \} \]
\[ x_n = (x_{n1}, x_{n2}, \ldots, x_{nd}) \]
\[ \log p(y_n|x_n, \theta) \approx \eta(\theta) \cdot \tau(y_n, x_n) \]
\[ \tau(y_n, x_n) = (y_n, x_{n1}, x_{n2}, \ldots, x_{nd}) \]
Polynomial approximate sufficient statistics (PASS)

\[ Y = \{y_1, y_2, \ldots, y_N\} \]
\[ X = \{x_1, x_2, \ldots, x_N\} \]
\[ x_n = (x_{n1}, x_{n2}, \ldots, x_{nd}) \]
\[ \log p(y_n|x_n, \theta) \approx \eta(\theta) \cdot \tau(y_n, x_n) \]
\[ \tau(y_n, x_n) = (y_n, x_{n1}, x_{n2}, \ldots, x_{nd}, y_n^2, x_{n1}^2, x_{n2}^2, \ldots, x_{nd}^2) \]
Polynomial approximate sufficient statistics (PASS)

\[ Y = \{y_1, y_2, \ldots, y_N\} \]
\[ X = \{x_1, x_2, \ldots, x_N\} \]
\[ x_n = (x_{n1}, x_{n2}, \ldots, x_{nd}) \]
\[ \log p(y_n|x_n, \theta) \approx \eta(\theta) \cdot \tau(y_n, x_n) \]
\[ \tau(y_n, x_n) = (y_n, x_{n1}, x_{n2}, \ldots, x_{nd}, y_n^2, x_{n1}^2, x_{n2}^2, \ldots, x_{nd}^2, y_n x_{n1}, \ldots, y_n x_{nd}) \]
Polynomial approximate sufficient statistics (PASS)

\[ Y = \{y_1, y_2, \ldots, y_N\} \]
\[ X = \{x_1, x_2, \ldots, x_N\} \]
\[ x_n = (x_{n1}, x_{n2}, \ldots, x_{nd}) \]
\[ \log p(y_n|x_n, \theta) \approx \eta(\theta) \cdot \tau(y_n, x_n) \]
\[ \tau(y_n, x_n) = (y_n, x_{n1}, x_{n2}, \ldots, x_{nd}, y_n^2, x_{n1}^2, x_{n2}^2, \ldots, x_{nd}^2, y_n x_{n1}, \ldots, y_n x_{nd}, x_{n1} x_{n2}, x_{n1} x_{n3}, \ldots) \]
Polynomial approximate sufficient statistics (PASS)

\[ Y = \{ y_1, y_2, \ldots, y_N \} \]
\[ X = \{ x_1, x_2, \ldots, x_N \} \]
\[ x_n = (x_{n1}, x_{n2}, \ldots, x_{nd}) \]
\[ \log p(y_n|x_n, \theta) \approx \eta(\theta) \cdot \tau(y_n, x_n) \]
\[ \tau(y_n, x_n) = (y_n, x_{n1}, x_{n2}, \ldots, x_{nd}, \]
\[ y_n^2, x_{n1}^2, x_{n2}^2, \ldots, x_{nd}^2, \]
\[ y_n x_{n1}, \ldots, y_n x_{nd}, \]
\[ x_{n1} x_{n2}, x_{n1} x_{n3}, \ldots, \]
\[ \ldots, \]
Polynomial approximate sufficient statistics (PASS)

\[ Y = \{y_1, y_2, \ldots, y_N\} \]
\[ X = \{x_1, x_2, \ldots, x_N\} \]
\[ x_n = (x_{n1}, x_{n2}, \ldots, x_{nd}) \]
\[ \log p(y_n|x_n, \theta) \approx \eta(\theta) \cdot \tau(y_n, x_n) \]
\[ \tau(y_n, x_n) = (y_n, x_{n1}, x_{n2}, \ldots, x_{nd}, \\
    y_n^2, x_{n1}^2, x_{n2}^2, \ldots, x_{nd}^2, \\
    y_n x_{n1}, \ldots, y_n x_{nd}, \\
    x_{n1} x_{n2}, x_{n1} x_{n3}, \ldots, \\
    \ldots, \\
    y_n^M, x_{n1}^M, x_{n2}^M, \ldots, x_{nd}^M) \]
Polynomial approximate sufficient statistics (PASS)

\[
Y = \{y_1, y_2, \ldots, y_N\}
\]

\[
X = \{x_1, x_2, \ldots, x_N\}
\]

\[
x_n = (x_{n1}, x_{n2}, \ldots, x_{nd})
\]

\[
\log p(y_n|x) \propto \tau(y_n, x_n) = (y_n, x_{n1}, x_{n2}, \ldots, x_{nd}, \ y_n^2, x_{n1}^2, x_{n2}^2, \ldots, x_{nd}^2, \ y_n x_{n1}, \ldots, y_n x_{nd}, \ x_{n1} x_{n2}, x_{n1} x_{n3}, \ldots, \ \ldots, \ y_n^M, x_{n1}^M, x_{n2}^M, \ldots, x_{nd}^M)
\]

1. Scalability
Polynomial approximate sufficient statistics (PASS)

\[ Y = \{y_1, y_2, \ldots, y_N\} \]
\[ X = \{x_1, x_2, \ldots, x_N\} \]
\[ x_n = (x_{n1}, x_{n2}, \ldots, x_{nd}) \]
\[ \log p(y_n|x) \]

1. Scalability

2. Arbitrary accuracy

\[ \tau(y_n, x_n) = \]
\[ x_{n1}x_{n2}, x_{n1}x_{n3}, \ldots, \]
\[ \ldots, \]
\[ y_n^M, x_{n1}^M, x_{n2}^M, \ldots, x_{nd}^M \]
PASS for generalized linear models (PASS-GLM)
PASS for generalized linear models (PASS-GLM)

generalized linear models: $\log p(y_n \mid x_n, \theta) = \phi(y_n, \theta \cdot x_n)$
PASS for generalized linear models (PASS-GLM)

generalized linear models: $\log p(y_n \mid x_n, \theta) = \phi(y_n, \theta \cdot x_n)$

• Poisson regression (count data):
PASS for generalized linear models (PASS-GLM)

generalized linear models: \( \log p(y_n | x_n, \theta) = \phi(y_n, \theta \cdot x_n) \)

- Poisson regression (count data):
  ➞ # of foreclosures by region
PASS for generalized linear models (PASS-GLM)

generalized linear models: \( \log p(y_n \mid x_n, \theta) = \phi(y_n, \theta \cdot x_n) \)

- Poisson regression (count data):
  - \# of foreclosures by region
- Logistic regression (binary data)
PASS for generalized linear models (PASS-GLM)

**generalized linear models:** $\log p(y_n \mid x_n, \theta) = \phi(y_n, \theta \cdot x_n)$

- Poisson regression (count data):
  ➡ # of foreclosures by region

- Logistic regression (binary data):
  ➡ patient has cancer?
PASS for generalized linear models (PASS-GLM)

generalized linear models: $\log p(y_n \mid x_n, \theta) = \phi(y_n, \theta \cdot x_n)$

- Poisson regression (count data):
  ➞ # of foreclosures by region
- Logistic regression (binary data)
  ➞ patient has cancer?
- Robust regression (continuous data) 😞
PASS for generalized linear models (PASS-GLM)

generalized linear models: $\log p(y_n | x_n, \theta) = \phi(y_n, \theta \cdot x_n)$

- Poisson regression (count data):
  ➞ # of foreclosures by region
- Logistic regression (binary data)
  ➞ patient has cancer?
- Robust regression (continuous data)
  ➞ birth rate by region
PASS for generalized linear models (PASS-GLM)

generalized linear models: \( \log p(y_n | x_n, \theta) = \phi(y_n, \theta \cdot x_n) \)

- Poisson regression (count data):
  ➞ # of foreclosures by region
- Logistic regression (binary data)
  ➞ patient has cancer?
- Robust regression (continuous data)
  ➞ birth rate by region

\[ \tau(y_n, x_n) = \left(a(k, M)y_n^{k_0} \prod_{i=1}^{d} x_{ni}^{k_i}\right)_{k \in \mathbb{N}^{d+1}} \sum_{i} k \leq M \]

\[ \eta(\theta) = \left(\prod_{i=1}^{d} \theta_i^{k_i}\right)_{k \in \mathbb{N}^{d+1}} \sum_{i} k \leq M \]
Case study: logistic regression
Case study: logistic regression

Data \((x_n, y_n) \in \mathbb{R}^d \times \{\pm 1\}\)
Case study: logistic regression

data \((x_n, y_n) \in \mathbb{R}^d \times \{\pm 1\}\)

parameter \(\theta \in \mathbb{R}^d\)
Case study: logistic regression

data $(x_n, y_n) \in \mathbb{R}^d \times \{\pm1\}$

parameter $\theta \in \mathbb{R}^d$

log-likelihood $\log p(y_n | x_n, \theta) = -\log(1 + e^{-y_n x_n \cdot \theta})$
Case study: logistic regression

Data $(x_n, y_n) \in \mathbb{R}^d \times \{\pm 1\}$

Parameter $\theta \in \mathbb{R}^d$

Log-likelihood

$$\log p(y_n \mid x_n, \theta) = -\log(1 + e^{-y_n x_n \cdot \theta})$$

$$= \phi(y_n x_n \cdot \theta)$$
Case study: logistic regression

data \((x_n, y_n) \in \mathbb{R}^d \times \{\pm 1\}\)

parameter \(\theta \in \mathbb{R}^d\)

log-likelihood \(\log p(y_n \mid x_n, \theta) = -\log(1 + e^{-y_n x_n \cdot \theta})\)

\(= \phi(y_n x_n \cdot \theta) \approx \phi_2(y_n x_n \cdot \theta)\)
data \((x_n, y_n) \in \mathbb{R}^d \times \{\pm 1\}\)

parameter \(\theta \in \mathbb{R}^d\)

log-likelihood \(\log p(y_n \mid x_n, \theta) = -\log(1 + e^{-y_n x_n \cdot \theta})\)

\[\approx \phi(y_n x_n \cdot \theta) \approx \phi_2(y_n x_n \cdot \theta)\]
PASS-GLM effective in distributed and streaming settings
PASS-GLM effective in distributed and streaming settings

Criteo advertising dataset
PASS-GLM effective in distributed and streaming settings

Criteo advertising dataset

- Distributed
PASS-GLM effective in distributed and streaming settings

Criteo advertising dataset

- Distributed
  - 6M observations with 1K covariates
PASS-GLM effective in distributed and streaming settings

Criteo advertising dataset

- Distributed
  - 6M observations with 1K covariates
- **16 seconds** using 22 cores
PASS-GLM effective in distributed and streaming settings

Criteo advertising dataset

- Distributed
  - 6M observations with 1K covariates
  - 16 seconds using 22 cores

- Streaming
PASS-GLM effective in distributed and streaming settings

Criteo advertising dataset

- **Distributed**
  - 6M observations with 1K covariates
  - **16 seconds** using 22 cores

- **Streaming**
  - 40M observations with 20K covariates
PASS-GLM effective in distributed and streaming settings

Criteo advertising dataset

- Distributed
  - 6M observations with 1K covariates
  - **16 seconds** using 22 cores

- Streaming
  - 40M observations with 20K covariates
  - Competitive with SGD
Competitive approximation performance

- Webspam dataset
- $N = 350,000$
- $d = 127$
Competitive approximation performance

- Webspam dataset
- $N = 350,000$
- $d = 127$
Competitive approximation performance

- Webspam dataset
- \( N = 350,000 \)
- \( d = 127 \)
Competitive approximation performance

- Webspam dataset
- $N = 350,000$
- $d = 127$
Competitive approximation performance

- Webspam dataset
- $N = 350,000$
- $d = 127$

Ongoing work: Poisson regression-type models for cancer genomics
Competitive approximation performance

- Webspam dataset
- \( N = 350,000 \)
- \( d = 127 \)

Ongoing work: Poisson regression-type models for cancer genomics

\( N = \sim 10 \text{ million} \)
Competitive approximation performance

- Webspam dataset
- $N = 350,000$
- $d = 127$

Ongoing work: Poisson regression-type models for cancer genomics

$N = \sim 10$ million
$d = 10-20$
Competitive approximation performance

- Webspam dataset
- $N = 350,000$
- $d = 127$

Ongoing work: Poisson regression-type models for cancer genomics

$N = \sim 10$ million
$d = 10-20$
100x speed-up
• Approximate Bayes review

• Likelihood approximation and dataset compression

• Approximate sufficient statistics

• Accuracy guarantees
Accuracy guarantees
Accuracy guarantees

- Problem: existing scalable methods lack accuracy guarantees
Accuracy guarantees

- **Problem:** existing scalable methods lack accuracy guarantees

- **Question:** how do we measure closeness of the exact and approximate posteriors?
Accuracy guarantees

• Problem: existing scalable methods lack accuracy guarantees

• Question: how do we measure closeness of the exact and approximate posteriors?
Accuracy guarantees

• **Problem:** existing scalable methods lack accuracy guarantees

• **Question:** how do we measure closeness of the exact and approximate posteriors?

• **Recall:** want to compute means, variances, tail probabilities, etc.
Accuracy guarantees

• **Problem:** existing scalable methods lack accuracy guarantees

• **Question:** how do we measure closeness of the exact and approximate posteriors?

• **Recall:** want to compute means, variances, tail probabilities, etc.

• Good choice of measure: **1- and 2-Wasserstein distances** $d_W$
Accuracy guarantees

- **Problem:** existing scalable methods lack accuracy guarantees
- **Question:** how do we measure closeness of the exact and approximate posteriors?
- **Recall:** want to compute means, variances, tail probabilities, etc.
- Good choice of measure: 1- and 2-Wasserstein distances $d_W$
- **Why?** $d_W(p, q)$ small implies
Accuracy guarantees

- **Problem:** existing scalable methods lack accuracy guarantees 🅿️

- **Question:** how do we measure closeness of the exact and approximate posteriors?

- **Recall:** want to compute means, variances, tail probabilities, etc.

- **Good choice of measure:** 1- and 2-Wasserstein distances $d_W$

- **Why?** $d_W(p, q)$ small implies
  - means and variances close ✅
Accuracy guarantees

- **Problem:** existing scalable methods lack accuracy guarantees
- **Question:** how do we measure closeness of the exact and approximate posteriors?
- **Recall:** want to compute means, variances, tail probabilities, etc.
- Good choice of measure: **1- and 2-Wasserstein distances** $d_W$
- **Why?** $d_W(p, q)$ small implies
  - means and variances close ✓
  - (smoothed) tail probabilities close ✓
Wasserstein distance with approximate likelihoods
Wasserstein distance with approximate likelihoods

\[ \pi(\theta|Y) = \frac{p(Y|\theta)\pi_0(\theta)}{Z} \]
Wasserstein distance with approximate likelihoods

\[
\pi(\theta|Y) = \frac{p(Y|\theta)\pi_0(\theta)}{Z} \quad \tilde{\pi}(\theta|Y) = \frac{e^{\ell(\theta, g(Y))}\pi_0(\theta)}{\tilde{Z}}
\]
Wasserstein distance with approximate likelihoods

\[ \pi(\theta | Y) = \frac{p(Y | \theta) \pi_0(\theta)}{Z} \]

\[ \tilde{\pi}(\theta | Y) = \frac{e^{\ell(\theta, g(Y))} \pi_0(\theta)}{\tilde{Z}} \]

\[ \varepsilon(\theta) = \| \nabla_\theta \log p(Y | \theta) - \nabla_\theta \ell(\theta, g(Y)) \|_2 \]
Wasserstein distance with approximate likelihoods

\[
\pi(\theta|Y) = \frac{p(Y|\theta)\pi_0(\theta)}{Z} \quad \tilde{\pi}(\theta|Y) = \frac{e^{\ell(\theta,g(Y))}\pi_0(\theta)}{\tilde{Z}}
\]

\[
\varepsilon(\theta) = \|\nabla_\theta \log p(Y|\theta) - \nabla_\theta \ell(\theta, g(Y))\|_2
\]

Wasserstein distance with approximate likelihoods

\[ \pi(\theta | Y) = \frac{p(Y | \theta) \pi_0(\theta)}{Z} \quad \tilde{\pi}(\theta | Y) = \frac{e^{\ell(\theta, g(Y))} \pi_0(\theta)}{\tilde{Z}} \]

\[ \varepsilon(\theta) = \left\| \nabla_{\theta} \log p(Y | \theta) - \nabla_{\theta} \ell(\theta, g(Y)) \right\|_2 \]


- \( \pi \) is “well-behaved” and
Wasserstein distance with approximate likelihoods

\[
\pi(\theta | Y) = \frac{p(Y | \theta)\pi_0(\theta)}{Z} \quad \tilde{\pi}(\theta | Y) = \frac{e^{\ell(\theta, g(Y))}\pi_0(\theta)}{\tilde{Z}}
\]

\[
\varepsilon(\theta) = \| \nabla_\theta \log p(Y | \theta) - \nabla_\theta \ell(\theta, g(Y)) \|_2
\]

**Theorem (H. & Zou 2017, H. 2018).** Assume

- \( \pi \) is “well-behaved” and
- \( \varepsilon(\theta) \leq \varepsilon \).
Wasserstein distance with approximate likelihoods

\[ \pi(\theta|Y) = \frac{p(Y|\theta)\pi_0(\theta)}{Z} \quad \tilde{\pi}(\theta|Y) = \frac{e^{\ell(\theta, g(Y))}\pi_0(\theta)}{\tilde{Z}} \]

\[ \varepsilon(\theta) = \| \nabla_\theta \log p(Y|\theta) - \nabla_\theta \ell(\theta, g(Y)) \|_2 \]

**Theorem (H. & Zou 2017, H. 2018).** Assume

- \( \pi \) is “well-behaved” and
- \( \varepsilon(\theta) \leq \varepsilon \).

Then \( d_W(\pi, \tilde{\pi}) \leq c_\pi \varepsilon \).
PASS-LR provides a high-quality approximation

\[ \tilde{\pi}_M = \text{order } M \text{ PASS-LR approximate posterior} \]
PASS-LR provides a high-quality approximation

\[ \tilde{\pi}_M = \text{order } M \text{ PASS-LR approximate posterior} \]

Theorem (H., Adams, Broderick 2017).
PASS-LR provides a high-quality approximation

\[ \tilde{\pi}_M = \text{order } M \text{ PASS-LR approximate posterior} \]

Theorem (H., Adams, Broderick 2017). Assume prior and data are “well-behaved”. 
PASS-LR provides a high-quality approximation

Theorem (H., Adams, Broderick 2017). Assume prior and data are “well-behaved”.

Then there exist $c > 0$ and $0 < r < 1$ such that
PASS-LR provides a high-quality approximation

Theorem (H., Adams, Broderick 2017). Assume prior and data are “well-behaved”.

Then there exist $c > 0$ and $0 < r < 1$ such that

$$d_{W}(\pi, \tilde{\pi}_M) \leq cdr^M$$
PASS-LR provides a high-quality approximation

\[ \tilde{\pi}_M = \text{order } M \text{ PASS-LR approximate posterior} \]

Theorem (H., Adams, Broderick 2017). Assume prior and data are “well-behaved”.

Then there exist \( c > 0 \) and \( 0 < r < 1 \) such that

\[ d_W(\pi, \tilde{\pi}_M) \leq cdr^M \]

- Similar results for other GLMs
Concluding thoughts
Concluding thoughts

- Empirical work
Concluding thoughts

- Empirical work
  - Hierarchical models
Concluding thoughts

- Empirical work
  - Hierarchical models
  - Applications to a wider range of GLMs
Concluding thoughts

• Empirical work
  • Hierarchical models
  • Applications to a wider range of GLMs
  • Practitioner buy-in
Concluding thoughts

- Empirical work
  - Hierarchical models
  - Applications to a wider range of GLMs
  - Practitioner buy-in
- Very- and ultra-high dimensional parameter spaces
Concluding thoughts

- Empirical work
  - Hierarchical models
  - Applications to a wider range of GLMs
  - Practitioner buy-in
- Very- and ultra-high dimensional parameter spaces
- Non-parametric models:
Concluding thoughts

- Empirical work
  - Hierarchical models
  - Applications to a wider range of GLMs
  - Practitioner buy-in
- Very- and ultra-high dimensional parameter spaces
- Non-parametric models:
  - Coresets for Gaussian processes, connections to inducing point methods
Concluding thoughts

• Empirical work
  • Hierarchical models
  • Applications to a wider range of GLMs
  • Practitioner buy-in

• Very- and ultra-high dimensional parameter spaces

• Non-parametric models:
  • Coresets for Gaussian processes, connections to inducing point methods

• Combinatorial parameter spaces
Thanks!


Using diffusions to understand approximation error
Using diffusions to understand approximation error

• **Diffusion:** continuous-time Markov process with unique stationary distribution
Using diffusions to understand approximation error

- **Diffusion:** continuous-time Markov process with unique stationary distribution

\[ b(\theta) = \nabla_\theta \log \pi(\theta | Y) \]
\[ \tilde{b}(\theta) = \nabla_\theta \log \tilde{\pi}(\theta | Y) \]
Using diffusions to understand approximation error

- **Diffusion**: continuous-time Markov process with unique stationary distribution

\[
\begin{align*}
    b(\theta) &= \nabla_\theta \log \pi(\theta | Y) \\
    \tilde{b}(\theta) &= \nabla_\theta \log \tilde{\pi}(\theta | Y)
\end{align*}
\]

\[
\begin{align*}
    d\theta_t &= b(\theta_t)dt + \sqrt{2}dW_t \\
    d\tilde{\theta}_t &= \tilde{b}(\tilde{\theta}_t)dt + \sqrt{2}d\tilde{W}_t
\end{align*}
\]
Using diffusions to understand approximation error

- **Diffusion**: continuous-time Markov process with unique stationary distribution

\[
\begin{align*}
b(\theta) &= \nabla_\theta \log \pi(\theta | Y) \\
\tilde{b}(\theta) &= \nabla_\theta \log \tilde{\pi}(\theta | Y)
\end{align*}
\]

d\theta_t = b(\theta_t)dt + \sqrt{2}dW_t \\
d\tilde{\theta}_t = \tilde{b}(\tilde{\theta}_t)dt + \sqrt{2}d\tilde{W}_t
Using diffusions to understand approximation error

- **Diffusion**: continuous-time Markov process with unique stationary distribution

\[
\begin{align*}
    \hat{b}(\theta) &= \nabla_{\theta} \log \pi(\theta | Y) \\
    b(\theta) &= \nabla_{\theta} \log \hat{\pi}(\theta | Y)
\end{align*}
\]

\[
\begin{align*}
    d\theta_t &= b(\theta_t)dt + \sqrt{2}dW_t \\
    d\tilde{\theta}_t &= \tilde{b}(\tilde{\theta}_t)dt + \sqrt{2}d\tilde{W}_t
\end{align*}
\]
Using diffusions to understand approximation error

- **Diffusion**: continuous-time Markov process with unique stationary distribution

\[
\begin{align*}
\dot{b}(\theta) &= \nabla_\theta \log \pi(\theta | Y) \\
\tilde{b}(\theta) &= \nabla_\theta \log \tilde{\pi}(\theta | Y)
\end{align*}
\]

\[
\begin{align*}
\text{d}\theta_t &= b(\theta_t) \text{d}t + \sqrt{2} \text{d}W_t \\
\text{d}\tilde{\theta}_t &= \tilde{b}(\tilde{\theta}_t) \text{d}t + \sqrt{2} \text{d}\tilde{W}_t
\end{align*}
\]
Using diffusions to understand approximation error

- **Diffusion**: continuous-time Markov process with unique stationary distribution

\[ b(\theta) = \nabla_{\theta} \log \pi(\theta | Y) \]
\[ \tilde{b}(\theta) = \nabla_{\theta} \log \tilde{\pi}(\theta | Y) \]

\[ d\theta_t = b(\theta_t)dt + \sqrt{2}dW_t \]
\[ d\tilde{\theta}_t = \tilde{b}(\tilde{\theta}_t)dt + \sqrt{2}d\tilde{W}_t \]
Using diffusions to understand approximation error

- **Diffusion:** continuous-time Markov process with unique stationary distribution

\[
\begin{align*}
\frac{db}{dt} &= \nabla_{\theta} \log \pi(\theta | Y) \\
\tilde{b}(\theta) &= \nabla_{\theta} \log \tilde{\pi}(\theta | Y) \\
d\theta_t &= b(\theta_t) dt + \sqrt{2} dW_t \\
d\tilde{\theta}_t &= \tilde{b}(\tilde{\theta}_t) dt + \sqrt{2} d\tilde{W}_t
\end{align*}
\]
Using diffusions to understand approximation error

- **Diffusion**: continuous-time Markov process with unique stationary distribution

\[
b(\theta) = \nabla_\theta \log \pi(\theta | Y) \\
\tilde{b}(\theta) = \nabla_\theta \log \tilde{\pi}(\theta | Y)
\]

\[
d\theta_t = b(\theta_t)dt + \sqrt{2}dW_t \\
\tilde{d}\tilde{\theta}_t = \tilde{b}(\tilde{\theta}_t)dt + \sqrt{2}d\tilde{W}_t
\]
Using diffusions to understand approximation error

- **Diffusion**: continuous-time Markov process with unique stationary distribution
- **Intuition**: if diffusion mixes quickly, then gradient errors don’t have time to build up

\[
\begin{align*}
\mathbf{b}(\theta) &= \nabla_{\theta} \log \pi(\theta | Y) \\
\tilde{\mathbf{b}}(\theta) &= \nabla_{\theta} \log \tilde{\pi}(\theta | Y)
\end{align*}
\]

\[
\begin{align*}
\mathbf{d}\theta_t &= \mathbf{b}(\theta_t)dt + \sqrt{2}dW_t \\
\tilde{\mathbf{d}}\tilde{\theta}_t &= \tilde{\mathbf{b}}(\tilde{\theta}_t)dt + \sqrt{2}d\tilde{W}_t
\end{align*}
\]
Wasserstein distance bounds from diffusions
Wasserstein distance bounds from diffusions

Wasserstein distance bounds from diffusions


Assume the diffusion converges at rate $r(t)$.

$$d\theta_t = b(\theta_t)dt + \sqrt{2}dW_t$$
Wasserstein distance bounds from diffusions


Assume the diffusion converges at rate $r(t)$.

Let $I(r) = \int r(t) \, dt$. 

\[ d\theta_t = b(\theta_t) \, dt + \sqrt{2} \, dW_t \]
Wasserstein distance bounds from diffusions


Assume the diffusion converges at rate $r(t)$.

Let $I(r) = \int r(t) \, dt$.

Then $d_W(\pi, \tilde{\pi}) \leq I(r) \mathbb{E}_{\tilde{\pi}}[\| b - \tilde{b} \|_2]$.

\[
d\theta_t = b(\theta_t) \, dt + \sqrt{2} \, dW_t
\]

\[
b(\theta) = \nabla_\theta \log \pi(\theta \mid Y)
\]

\[
\tilde{b}(\theta) = \nabla_\theta \log \tilde{\pi}(\theta \mid Y)
\]
Wasserstein distance bounds from diffusions


Assume the diffusion converges at rate $r(t)$.

Let $I(r) = \int r(t) \, dt$.

Then $d_W(\pi, \tilde{\pi}) \leq I(r) \mathbb{E}_{\tilde{\pi}}[\|b - \tilde{b}\|_2]$.
Wasserstein distance bounds from diffusions


Assume the diffusion converges at rate $r(t)$.

Let $I(r) = \int r(t) \, dt$.

Then $d_W(\pi, \tilde{\pi}) \leq I(r) \mathbb{E}_{\tilde{\pi}}[\|b - \tilde{b}\|_2]$. 

\[ d\theta_t = b(\theta_t)dt + \sqrt{2}dW_t \]

\[ b(\theta) = \nabla_{\theta} \log \pi(\theta \mid Y) \]

\[ \tilde{b}(\theta) = \nabla_{\theta} \log \tilde{\pi}(\theta \mid Y) \]
Wasserstein distance bounds from diffusions


Assume the diffusion converges at rate $r(t)$.

Let $I(r) = \int r(t) \, dt$.

Then $d_W(\pi, \tilde{\pi}) \leq I(r) \mathbb{E}_{\tilde{\pi}}[\|b - \tilde{b}\|_2].$

- Proof techniques:

\[
\begin{align*}
    d\theta_t &= b(\theta_t)dt + \sqrt{2}dW_t \\
    b(\theta) &= \nabla_\theta \log \pi(\theta | Y) \\
    \tilde{b}(\theta) &= \nabla_\theta \log \tilde{\pi}(\theta | Y)
\end{align*}
\]
Wasserstein distance bounds from diffusions


Assume the diffusion converges at rate $r(t)$.

Let $I(r) = \int r(t) \, dt$.

Then $d_W(\pi, \tilde{\pi}) \leq I(r) \mathbb{E}_{\tilde{\pi}}[\|b - \tilde{b}\|_2]$.

Proof techniques:
- Stein’s method (for 1-Wasserstein version)
Wasserstein distance bounds from diffusions

**Theorem (H. & Zou 2017, H. 2018).**

Assume the diffusion converges at rate $r(t)$. Let $I(r) = \int r(t) \, dt$. Then

$$d_W(\pi, \tilde{\pi}) \leq I(r) \left[ \mathbb{E}_{\tilde{\pi}}[\|b - \tilde{b}\|_2] + \mathbb{E}_{\pi}[\|c\|_2] \right].$$

- **Proof techniques:**
  - Stein’s method (for 1-Wasserstein version)
  - A coupling argument + Ito’s lemma (for 2-Wasserstein version)

\[ \begin{align*}
  d\theta_t &= b(\theta_t) dt + \sqrt{2} dW_t \\
  b(\theta) &= \nabla_{\theta} \log \pi(\theta | Y) \\
  \tilde{b}(\theta) &= \nabla_{\theta} \log \tilde{\pi}(\theta | Y)
\end{align*} \]