Privacy and Security in Bayesian Inference

Louis J. M. Aslett[1], Murray Pollock[2], Hongsheng Dai[3] & Gareth O. Roberts[2]

[1] Durham University
[2] University of Warwick
[3] University of Essex

Seminar in Department of Mathematics and Statistics
Lancaster Universtiy
5 December 2018

Durham University

Introduction

Motivation

Security in statistics applications is a growing concern:

- computing in a 'hostile' environment (e.g. cloud computing);
- donation of sensitive/personal data (e.g. medical/genetic studies);
- complex models on constrained devices (e.g. smart watches)
- running confidential algorithms on confidential data (e.g. engineering reliability)
- big(ger) data (e.g. pooling data sources)

Perspectives on “privacy”

- Differential privacy
  - on outcomes of ‘statistical queries’
  - guarantees of privacy for individual observations

Perspectives on “privacy”

- Differential privacy
  - on outcomes of ‘statistical queries’
  - guarantees of privacy for individual observations

- Data privacy
  - at rest
  - during fitting
  - data pooling

Perspectives on “privacy”

- Differential privacy
  - on outcomes of ‘statistical queries’
  - guarantees of privacy for individual observations
- Data privacy
  - at rest
  - during fitting
  - data pooling
- Model privacy (see other work with Sam Livingstone, UCL)
  - prior distributions
  - model formulation

The perspective for today ...

- **Eve**, **Cain** and **Abel** have private data of the same type.
- There is a Bayesian model of mutual interest.
- Inference would be improved by pooling the data, but privacy constraints (eg GDPR) prevent this.

The perspective for today ...

- **Eve**, **Cain** and **Abel** have private data of the same type.
- There is a Bayesian model of mutual interest.
- Inference would be improved by pooling the data, but privacy constraints (eg GDPR) prevent this.

Can Eve, Cain and Abel pool their data in order to fit a Bayesian model without revealing the raw data?

Agreed model

$$\pi(\cdot \mid \psi)$$

$$\pi(\psi)$$

Private data

$$\{\mathbf{x}_i = (x_{i1}, \dots, x_{id})\}_{i=1}^{n_1}$$

$$\{\mathbf{x}_i = (x_{i1}, \dots, x_{id})\}_{i=n_1+1}^{n_1+n_2}$$

$$\{\mathbf{x}_i = (x_{i1}, \dots, x_{id})\}_{i=n_1+n_2+1}^{N}$$

Differential Privacy

Differential privacy quantifies the privacy level of 'statistical queries'. Need for the mutually fitted model.

Informally, for today this is: "how much can be learned about the original data when we learn about the Bayes posterior from MCMC samples?"

Differential Privacy

Differential privacy quantifies the privacy level of 'statistical queries'. Need for the mutually fitted model.

Informally, for today this is: "how much can be learned about the original data when we learn about the Bayes posterior from MCMC samples?"

Strong statement: we assume an adversary has access to arbitrary auxilliary information ... data being 'big' not a protection.

Definition (*Differential Privacy*)

We say that a randomised algorithm $\mathcal{M}$ is (ε, δ)-differentially private if for all $\mathcal{S} \subseteq \text{Range}(\mathcal{M})$ and for all x, y such that $\|x - y\|_1 \leq 1$:

$$\mathbb{P}(\mathcal{M}(x) \in \mathcal{S}) \leq \exp(\varepsilon)\mathbb{P}(\mathcal{M}(y) \in \mathcal{S}) + \delta$$

Previous work

Everyone sees fitted model parameters, differential privacy of output important. Previous perspectives applied at the combination step.

Previous work

Everyone sees fitted model parameters, differential privacy of output important. Previous perspectives applied at the combination step.

Close prior work, "On the Use of Penalty MCMC for Differential Privacy", S. Yildirim, 2016.

- Parties exchange noisy log-likelihood contributions (differentially private).
- Post process these with accept/reject step.
- View as penalty MCMC algorithm.
- Final posterior samples shown to be differentially private.

Previous work

Everyone sees fitted model parameters, differential privacy of output important. Previous perspectives applied at the combination step.

Close prior work, "On the Use of Penalty MCMC for Differential Privacy", S. Yildirim, 2016.

- Parties exchange noisy log-likelihood contributions (differentially private).
- Post process these with accept/reject step.
- View as penalty MCMC algorithm.
- Final posterior samples shown to be differentially private.

Today: Can we produce a method with better efficiency properties than penalty MCMC by leveraging cryptographic methods?

Cryptography the solution?

Encryption can provide security guarantees …

Easy

$$\texttt{Enc}(k_p, m) \rightleftharpoons c \qquad\qquad \texttt{Dec}(k_s, c) = m$$

Hard without k_s

… but is typically ‘brittle’.

Cryptography the solution?

Encryption can provide security guarantees …

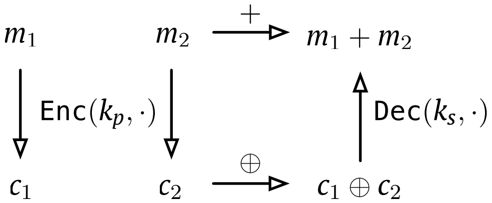
Easy

$$\texttt{Enc}(k_p, m) \rightleftharpoons c \qquad\qquad \texttt{Dec}(k_s, c) = m$$

Hard without k_s

… but is typically 'brittle'.

Arbitrary addition and multiplication is possible with **fully homomorphic encryption** schemes (Gentry, 2009).



Back to the problem …

Agreed model

$\pi(\cdot \mid \psi)$

$\pi(\psi)$

Private data

$\{\mathbf{x}_i = (x_{i1}, \ldots, x_{id})\}_{i=1}^{n_1}$

$\{\mathbf{x}_i = (x_{i1}, \ldots, x_{id})\}_{i=n_1+1}^{n_1+n_2}$

$\{\mathbf{x}_i = (x_{i1}, \ldots, x_{id})\}_{i=n_1+n_2+1}^{N}$

Back to the problem ...

Agreed model

$\pi(\cdot \mid \psi)$

$\pi(\psi)$

Private data

$\{\mathbf{x}_i = (x_{i1}, \ldots, x_{id})\}_{i=1}^{n_1}$

$\{\mathbf{x}_i = (x_{i1}, \ldots, x_{id})\}_{i=n_1+1}^{n_1+n_2}$

$\{\mathbf{x}_i = (x_{i1}, \ldots, x_{id})\}_{i=n_1+n_2+1}^{N}$

↓

$\mathbf{x}_i^\star = \text{Enc}(k_p, \mathbf{x}_i)$

Back to the problem ...

Agreed model

$\pi(\cdot \mid \psi)$

$\pi(\psi)$

Private data

$\{\mathbf{x}_i = (x_{i1}, \ldots, x_{id})\}_{i=1}^{n_1}$

$\{\mathbf{x}_i = (x_{i1}, \ldots, x_{id})\}_{i=n_1+1}^{n_1+n_2}$

$\{\mathbf{x}_i = (x_{i1}, \ldots, x_{id})\}_{i=n_1+n_2+1}^{N}$

$$\mathbf{x}_i^\star = \mathrm{Enc}(k_p, \mathbf{x}_i)$$

$$\pi(\psi \mid X) \propto \mathrm{Dec}\left[k_s, \prod_{i=1}^{N} \pi(\mathbf{x}_i^\star | \mathrm{Enc}(k_p, \psi))\mathrm{Enc}(k_p, \pi(\psi))\right]$$

Back to the problem ...

Agreed model

$$\pi(\cdot \,|\, \psi)$$

$$\pi(\psi)$$

✗ Likelihood restricted to low degree polynomials

✗ Can only handle very small N due to multiplicative depth

✗ MAP/posterior? How? MCMC?

✗ Who holds secret key?

Private data

$$\{\mathbf{x}_i = (x_{i1}, \dots, x_{id})\}_{i=1}^{n_1}$$

$$\{\mathbf{x}_i = (x_{i1}, \dots, x_{id})\}_{i=n_1+1}^{n_1+n_2}$$

$$\{\mathbf{x}_i = (x_{i1}, \dots, x_{id})\}_{i=n_1+n_2+1}^{N}$$

$$\mathbf{x}_i^\star = \mathrm{Enc}(k_p, \mathbf{x}_i)$$

$$\pi(\psi \,|\, X) \propto$$

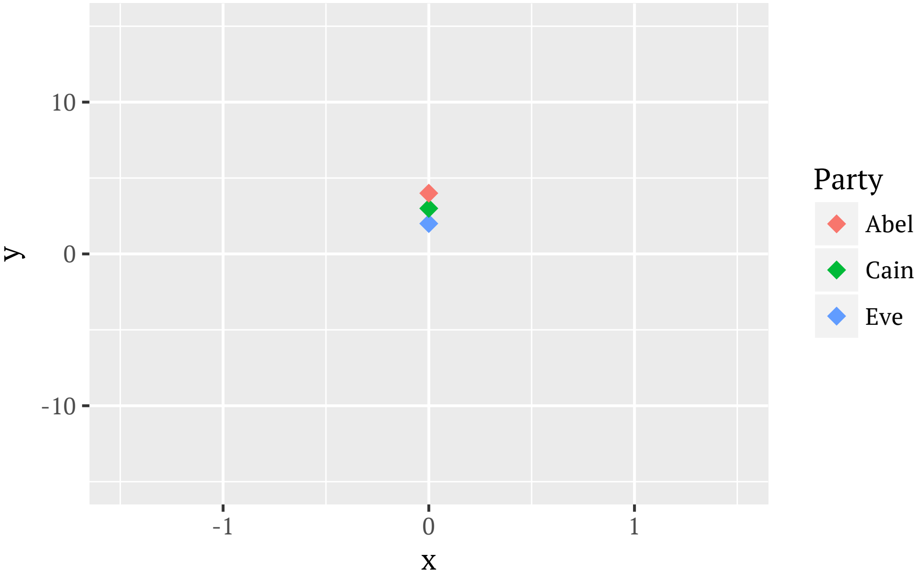
$$\mathrm{Dec}\left[k_s, \prod_{i=1}^{N} \pi(\mathbf{x}_i^\star | \mathrm{Enc}(k_p, \psi))\mathrm{Enc}(k_p, \pi(\psi))\right]$$

Homomorphic Secret Sharing (HSS)

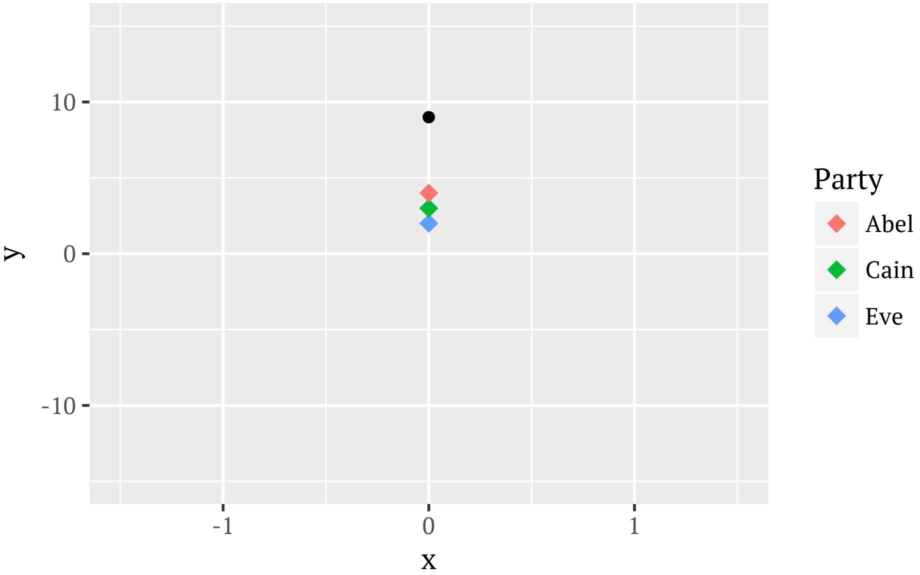
We require the properties of homomorphic encryption, but for multiple users.

- Yao's garbled circuit protocol
  - Boolean circuit construction
- Modern multiparty computation (eg SPDZ)
  - Computationally very intensive
  - Communication for multiplication operations
- Homomorphic secret sharing
  - Fast computation
  - Information theoretic security is possible

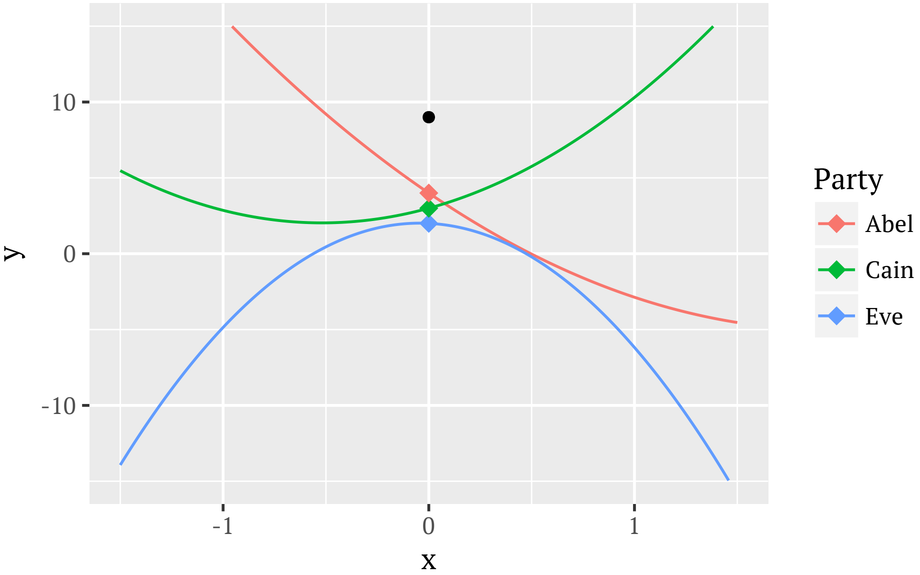
(Very Simplified) Homomorphic Secret Sharing



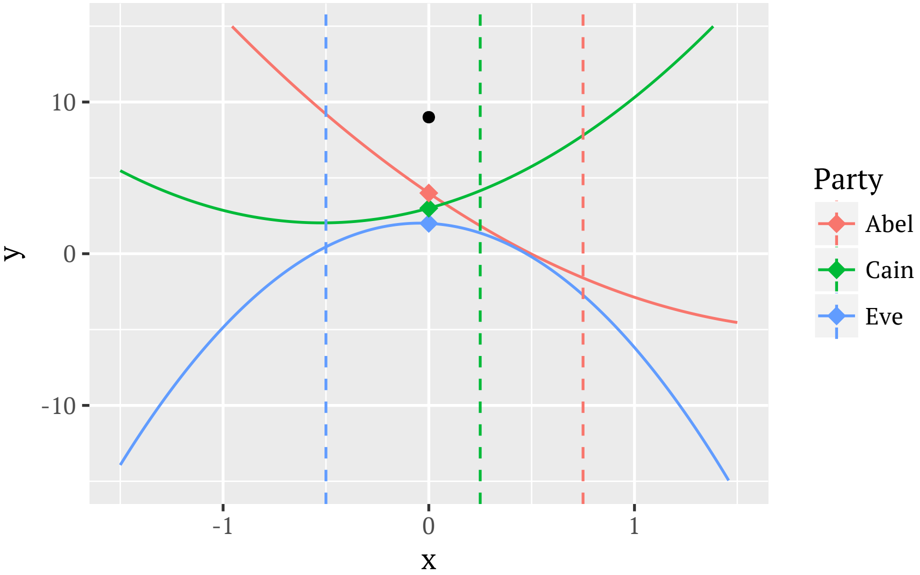
(Very Simplified) Homomorphic Secret Sharing



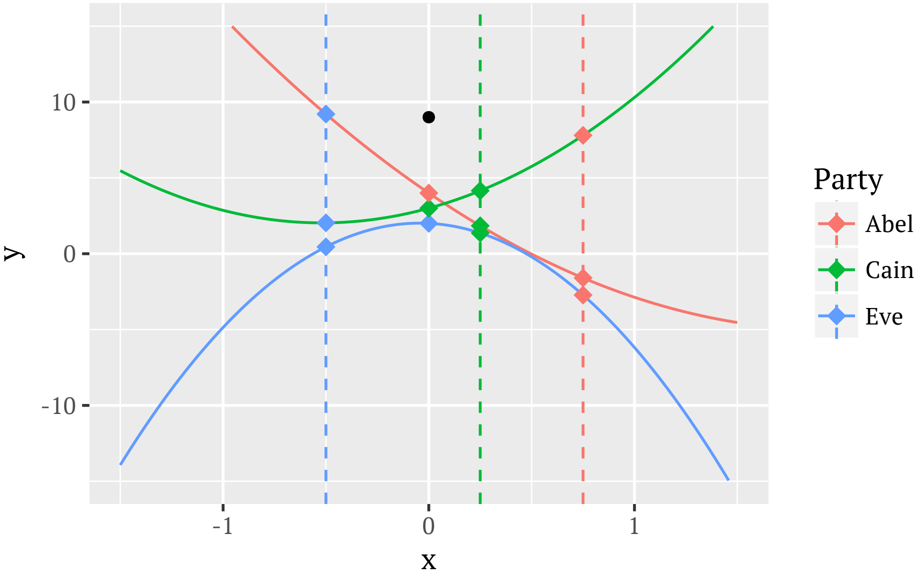
(Very Simplified) Homomorphic Secret Sharing



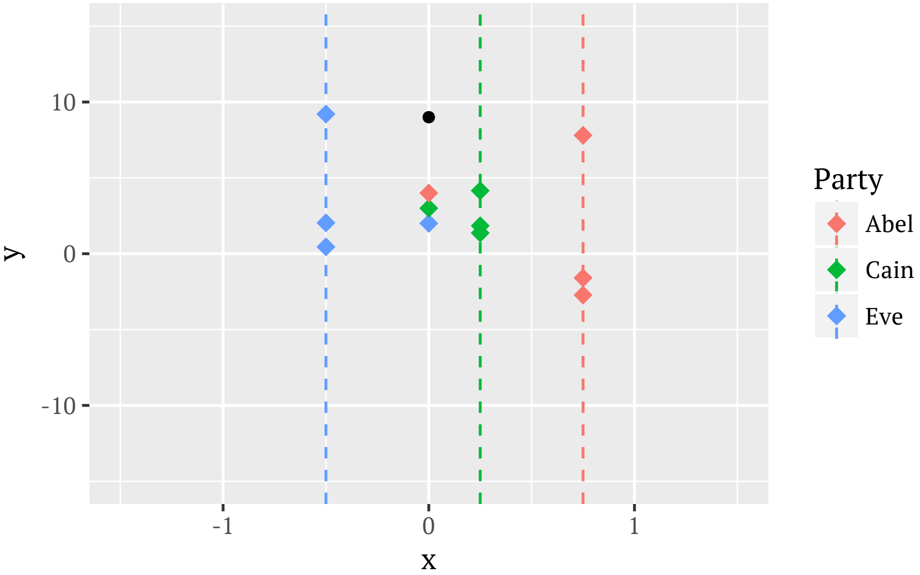
(Very Simplified) Homomorphic Secret Sharing



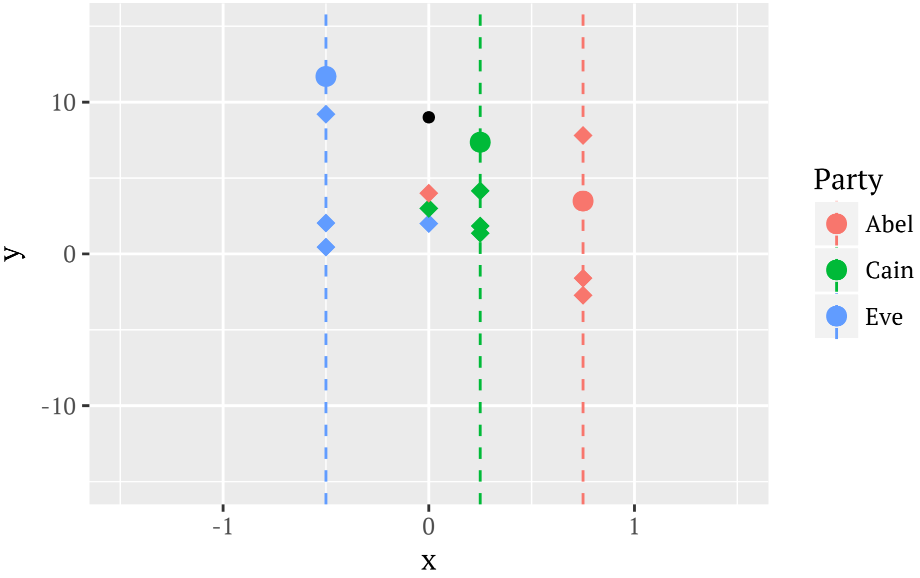
(Very Simplified) Homomorphic Secret Sharing



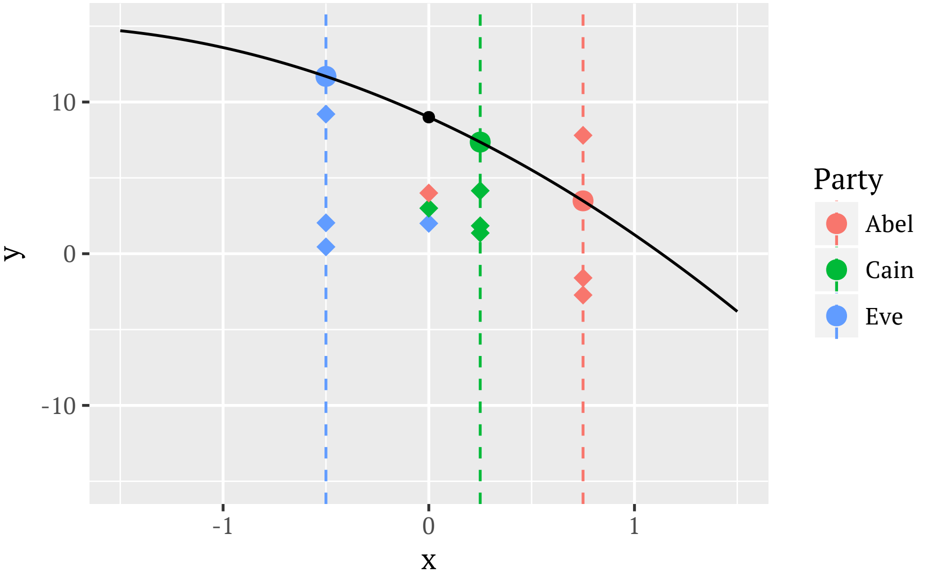
(Very Simplified) Homomorphic Secret Sharing



(Very Simplified) Homomorphic Secret Sharing



(Very Simplified) Homomorphic Secret Sharing



Security of Homomorphic Secret Sharing

How secure is this secret sharing compared to, say, homomorphic encryption, RSA, AES, ...?

Security of Homomorphic Secret Sharing

How secure is this secret sharing compared to, say, homomorphic encryption, RSA, AES, ...?

Assuming parties do not collude:

Information Theoretically Secure

This means that an adversary with *unbounded* compute power cannot determine your secret data.

Advanced Homomorphic Secret Sharing

Real HSS methods are more advanced, based on polynomials over a finite field, where:

- Upto $\frac{1}{3}$ of parties may be corrupted.
- Combining cryptographic and secret sharing to manage dishonest majority scenarios (but losing information theoretic security).
- Security against active attackers.
- Both perfectly and imperfectly secure communication channels.
- ...

Confidential MCMC

Metropolis-Hastings

To sample from a target (unnormalised) density of interest, $\pi(\theta)$.

1. Initialise with a sample θ_0.
2. Given a sample θ_i, propose a new sample $\theta' \sim q(\cdot \mid \theta_i)$.
3. Compute $\alpha(\theta_i, \theta') = \min\left\{1, r(\theta_i, \theta')\right\}$ where

$$r(\theta_i, \theta') := \frac{\pi(\theta')q(\theta_i \mid \theta')}{\pi(\theta_i)q(\theta' \mid \theta_i)} \tag{1}$$

4. With probability $\alpha(\theta_i, \theta')$ set $\theta_{i+1} = \theta'$, else set $\theta_{i+1} = \theta_i$.
5. Repeat steps 2–4 for a fixed number of iterations.

Bayesian inference

Often assume independence so that

$$\pi(\theta) \equiv \pi(\theta \mid \mathbf{y}) \propto p(\theta) \prod_{i=1}^{N} p(y_i \mid \theta)$$

In privacy setting, consider partition of observation indices, $\{\mathcal{I}_i\}_{i=1}^m$, st

$$\bigcup_{i=1}^{m} \mathcal{I}_i = \{1, \dots, N\} \text{ and } \mathcal{I}_i \cap \mathcal{I}_j = \emptyset \;\; \forall i \neq j$$

where participant j only has access to $\{y_i\}_{i \in \mathcal{I}_j}$. Then write Bayesian posterior:

$$\pi(\theta \mid \mathbf{y}) \propto p(\theta) \prod_{j=1}^{m} \prod_{i \in \mathcal{I}_j} p(y_i \mid \theta)$$

Log-likelihood shares

Define portion of likelihood computable by participant j,

$$p_j^\star(\theta) := \prod_{i \in \mathcal{I}_j} p(y_i \,|\, \theta)$$

Then,

$$\log \pi(\theta) = \log p(\theta) + \sum_{j=1}^{m} \log p_j^\star(\theta)$$

Log-likelihood shares

Define portion of likelihood computable by participant j,

$$p_j^\star(\theta) := \prod_{i \in \mathcal{I}_j} p(y_i \,|\, \theta)$$

Then,

$$\log \pi(\theta) = \log p(\theta) + \sum_{j=1}^{m} \log p_j^\star(\theta)$$

and acceptance ratio becomes,

$$\begin{aligned} \log r(\theta_i, \theta') = {} & \log p(\theta') - \log p(\theta_i) \\ & + \sum_{j=1}^{m} \left(\log p_j^\star(\theta') - \log p_j^\star(\theta_i) \right) \\ & + \log q(\theta_i \,|\, \theta') - \log q(\theta' \,|\, \theta_i) \end{aligned}$$

All done?

So, are we finished? Simply compute the acceptance ratio using homomorphic secret shares?

All done?

So, are we finished? Simply compute the acceptance ratio using homomorphic secret shares?

Not so fast … completely deterministic so no differential privacy guarantee can be provided when parties observe value of acceptance ratio!

Achieving differential privacy

Rewrite Metropolis-Hastings in an exactly equivalent way:

1. Initialise with a sample θ_0.
2. Given a sample θ_i, propose a new sample $\theta' \sim q(\cdot \mid \theta_i)$.

Achieving differential privacy

Rewrite Metropolis-Hastings in an exactly equivalent way:

1. Initialise with a sample θ_0.
2. Given a sample θ_i, propose a new sample $\theta' \sim q(\cdot \mid \theta_i)$.
3. Sample $U \sim \text{Unif}(0, 1)$ and compute $\eta = \log r(\theta_i, \theta') - \log U$
4. Set
$$\theta_{i+1} = \begin{cases} \theta_i & \text{if } \eta < 0 \\ \theta' & \text{if } \eta \geq 0 \end{cases}$$
5. Repeat steps 2–4 for a fixed number of iterations.

Achieving differential privacy

Rewrite Metropolis-Hastings in an exactly equivalent way:

1. Initialise with a sample θ_0.
2. Given a sample θ_i, propose a new sample $\theta' \sim q(\cdot \mid \theta_i)$.
3. Sample $U \sim \text{Unif}(0, 1)$ and compute $\eta = \log r(\theta_i, \theta') - \log U$
4. Set

$$\theta_{i+1} = \begin{cases} \theta_i & \text{if } \eta < 0 \\ \theta' & \text{if } \eta \geq 0 \end{cases}$$

5. Repeat steps 2–4 for a fixed number of iterations.

If we can compute η *and* establish $\eta \gtrless 0$, then the HSS step is a randomised algorithm.

Requirements

Main objective: hide the acceptance ratio $\log r(\theta_i, \theta')$.

Requirements

Main objective: hide the acceptance ratio $\log r(\theta_i, \theta')$.

But, this requires also hiding uniform random sample $U \sim \text{Unif}(0, 1)$. If a participant observes U, they can:

- learn $\log r(\theta_i, \theta')$ if they observe η.
- learn a bound on $\log r(\theta_i, \theta')$ if they observe $\eta \gtreqless 0$.

Requirements

Main objective: hide the acceptance ratio $\log r(\theta_i, \theta')$.

But, this requires also hiding uniform random sample $U \sim \text{Unif}(0, 1)$. If a participant observes U, they can:

- learn $\log r(\theta_i, \theta')$ if they observe η.
- learn a bound on $\log r(\theta_i, \theta')$ if they observe $\eta \gtrless 0$.

Note:

$$U \sim \text{Unif}(0, 1) \implies -\log U \sim \text{Exp}(1)$$

From Devroye (1986),

$$T, V, W \sim \text{Unif}(0, 1) \implies W(-\log TV) \sim \text{Exp}(1)$$

$\therefore$ collaboratively compute with two participants, one secret shares W, the other $-\log TV$.

Confidential MCMC algorithm

1. Initialise with a sample θ_0.
2. Given a sample θ_i, propose a new sample $\theta' \sim q(\cdot \mid \theta_i)$.
3. Participant 1 samples $U, V \sim \text{Unif}(0, 1)$
4. Participant 2 samples $W \sim \text{Unif}(0, 1)$
5. Compute $\eta := \log r(\theta_i, \theta') + W \log UV$ via homomorphic secret shares
6. Set
$$\theta_{i+1} = \begin{cases} \theta_i & \text{if } \eta < 0 \\ \theta' & \text{if } \eta \geq 0 \end{cases}$$
7. Repeat steps 2–5 for a fixed number of iterations.

Level of Differential Privacy

Can entirely hide the value of η by taking product with random positive value, so can assume we just observe accept/reject decision.

Level of Differential Privacy

Can entirely hide the value of η by taking product with random positive value, so can assume we just observe accept/reject decision.

In one iteration, we achieve same level of differential privacy as when observing a single iid draw from posterior:

Lemma (single iteration DP)

A single iteration of the confidential MCMC algorithm has differential privacy,

$$\frac{\mathbb{P}(\eta < 0 \,|\, \mathbf{y})}{\mathbb{P}(\eta < 0 \,|\, \mathbf{y}_{-i})} \leq e^{2C}$$

where $C = \sup_{y, y', \theta} |\log \pi(y \,|\, \theta) - \log \pi(y' \,|\, \theta)|$.

Level of Differential Privacy

Under repeated sampling to form a full MCMC output, differential privacy can still be achieved:

Theorem (MCMC trace DP)

Let d_θ be the dimension of parameter θ and let

$$\sup_{\mathbf{y},\theta} \left| \frac{\partial \log \pi(\mathbf{y} \mid \theta)}{\partial \theta_i} \right| \leq M$$

Then, by advanced composition of k iterations, differential privacy attains

$$\left(\varepsilon = \tilde{\varepsilon} \left(\sqrt{2k \log(1/\delta)} + k \left(e^{\tilde{\varepsilon}} - 1 \right) \right), \delta \right)$$

where $\tilde{\varepsilon} = 4 d_\theta n^{-1/2} M$

Improving?

Fixing some parameters and bounding for a very rough fit-on-the-slide comparison ...

The two primary terms in ε for a fixed k iterations leads private penalty method being larger by a multiplicative factor,

$$\approx \frac{\sqrt{2\beta' \log n}}{2\sigma} \quad \text{and} \quad \approx \frac{\beta' \log n}{2\sigma^2}$$

where β' is a selectable parameter > 1.

Improving?

Fixing some parameters and bounding for a very rough fit-on-the-slide comparison …

The two primary terms in ε for a fixed k iterations leads private penalty method being larger by a multiplicative factor,

$$\approx \frac{\sqrt{2\beta' \log n}}{2\sigma} \qquad \text{and} \qquad \approx \frac{\beta' \log n}{2\sigma^2}$$

where β' is a selectable parameter > 1.

Crucially,

- the penalty method has poorer performance in Peskun sense (Nicholls et al, 2012);
- this new method allows arbitrarily small user chosen δ.

Conclusion

Work in progress ...

1. Fuller characterisation of improvement provided vs not using cryptographic methods
2. Implementation is in development with
   - Shamir's secret sharing extended to including multiplication
   - fully secure network communication built in
   - automatic parsing and evaluation of a provided function circuits
3. Performance of the technique:
   - minimising circuit size?
   - optimal ordering of operations (accomodate latency)?
   - preemptive computation?
4. Important extensions:
   - beyond honest-but-curious security
   - eliminating communication

Work in progress …

1. Fuller characterisation of improvement provided vs not using cryptographic methods
2. Implementation is in development with
   - Shamir's secret sharing extended to including multiplication
   - fully secure network communication built in
   - automatic parsing and evaluation of a provided function circuits
3. Performance of the technique:
   - minimising circuit size?
   - optimal ordering of operations (accomodate latency)?
   - preemptive computation?
4. Important extensions:
   - beyond honest-but-curious security
   - eliminating communication

Thank you!