Robust Optimisation Monte Carlo

Michael U. Gutmann

michael.gutmann@ed.ac.uk

Institute for Adaptive and Neural Computation
School of Informatics, University of Edinburgh

22 November 2019

Reference

Borislav Ikonomov and Michael U. Gutmann
Robust Optimisation Monte Carlo
arXiv:1904.00670, 2019

Key messages

1. Optimisation Monte Carlo (OMC) is an existing method for efficient Bayesian inference with implicit models.
2. While efficient OMC under-estimates uncertainty by collapsing regions of near-constant likelihood into a single point.
3. A robust generalisation, robust OMC, explains and corrects this failure mode while maintaining OMC's benefits.

Contents

Contents

Overall topic of the talk

Bayesian parameter inference for models where

1. the likelihood function is too costly to evaluate
2. exact sampling – simulating data from the model – is possible

Implicit models

(Diggle and Gratton, JRSS, 1982)

- Parametric statistical models specified by a data generating mechanism $g : (\boldsymbol{\theta}, \mathbf{u}) \mapsto \mathbf{x} = g(\boldsymbol{\theta}, \mathbf{u})$
  - $\boldsymbol{\theta}$: parameters
  - $\mathbf{u}$: stochasticity / nuisance parameters with distribution $p(\mathbf{u})$.
  - $\mathbf{x}$: generated data

Implicit models

(Diggle and Gratton, JRSS, 1982)

- Parametric statistical models specified by a data generating mechanism $g : (\boldsymbol{\theta}, \mathbf{u}) \mapsto \mathbf{x} = g(\boldsymbol{\theta}, \mathbf{u})$
  - $\boldsymbol{\theta}$: parameters
  - $\mathbf{u}$: stochasticity / nuisance parameters with distribution $p(\mathbf{u})$.
  - $\mathbf{x}$: generated data
- The (deterministic) function g and the distribution $p(\mathbf{u})$ define the conditional distribution $p(\mathbf{x}|\boldsymbol{\theta})$
  - evaluating $p(\mathbf{x}|\boldsymbol{\theta})$ is generally intractable
    $\rightarrow$ likelihood function is intractable
  - drawing samples $\mathbf{x}_i \sim p(\mathbf{x}|\boldsymbol{\theta})$ is possible
    $\rightarrow$ we can exploit this to perform Bayesian inference

Implicit models

- The function $g(\boldsymbol{\theta}, \mathbf{u})$ is generally not known in closed form but implemented as computer code
  - $\boldsymbol{\theta}$: input parameters
  - $\mathbf{u}$: random draws performed when running the code / seed of the random number generator used

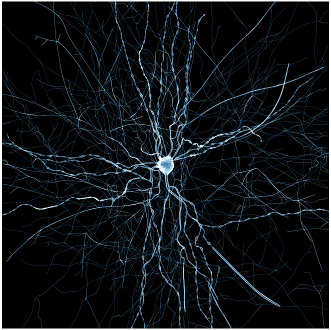
Implicit models

- The function $g(\boldsymbol{\theta}, \mathbf{u})$ is generally not known in closed form but implemented as computer code
  - $\boldsymbol{\theta}$: input parameters
  - $\mathbf{u}$: random draws performed when running the code / seed of the random number generator used
- Other names: Simulator-based models, stochastic simulation model, generative (latent-variable) models, ...

Importance

Such models and inference problems occur widely

- Evolutionary biology:
  Simulating evolution
- Neuroscience:
  Simulating neural activity
- Health science:
  Simulating the spread of an infectious disease
- Robotics:
  Simulating the outcome of an action
- Computer vision:
  Simulating naturalistic scenes
- ...



Simulated neural activity in rat somatosensory cortex
(Figure from `https://bbp.epfl.ch/nmc-portal`)

Bayesian inference for implicit models

- Task: Given
  - observed data $\mathbf{x}_o$,
  - an implicit model $g(\boldsymbol{\theta}, \mathbf{u})$, and
  - a prior distribution on $\boldsymbol{\theta}$,

  estimate the posterior $p(\boldsymbol{\theta}|\mathbf{x}_o)$ / obtain approximate samples from it.

Bayesian inference for implicit models

- Task: Given
  - observed data $\mathbf{x}_o$,
  - an implicit model $g(\boldsymbol{\theta}, \mathbf{u})$, and
  - a prior distribution on $\boldsymbol{\theta}$,

  estimate the posterior $p(\boldsymbol{\theta}|\mathbf{x}_o)$ / obtain approximate samples from it.
- Research fields: approximate Bayesian computation (ABC), likelihood-free inference, Bayesian indirect inference

  (overviews: Sisson et al 2018, Lintusaari et al 2017, Gutmann and Corander 2016, Drovandi 2015, Marin et al 2012)

Approximate Bayesian computation (ABC)

- ABC builds on the fact that samples from the posterior are given by samples from the prior for which simulated data $\mathbf{x}$ are close to the observed data $\mathbf{x}_o$.

Approximate Bayesian computation (ABC)

- ABC builds on the fact that samples from the posterior are given by samples from the prior for which simulated data $\mathbf{x}$ are close to the observed data $\mathbf{x}_o$.
- Two core ingredients of ABC algorithms:
  1. a distance function $d(\mathbf{x}, \mathbf{x}_o)$ between $\mathbf{x}$ and $\mathbf{x}_o$
  2. a search method to efficiently find such samples from the prior

Optimisation Monte Carlo (Meeds and Welling, NIPS 2015)

- Ingredients:
  - Distance $d(\mathbf{x}, \mathbf{x}_o) = ||\Phi(\mathbf{x}) - \Phi(\mathbf{x}_o)||_2$ with known Φ.
  - Search uses optimisation, which leads to increased efficiency.

Optimisation Monte Carlo (Meeds and Welling, NIPS 2015)

- Ingredients:
  - Distance $d(\mathbf{x}, \mathbf{x}_o) = ||\Phi(\mathbf{x}) - \Phi(\mathbf{x}_o)||_2$ with known Φ.
  - Search uses optimisation, which leads to increased efficiency.
- Assumptions:
  - (approximate) derivative of $\Phi(\mathbf{x}) = \Phi(g(\boldsymbol{\theta}, \mathbf{u})) = \mathbf{f}(\boldsymbol{\theta}, \mathbf{u})$ wrt $\boldsymbol{\theta}$ is available
  - $\dim(\boldsymbol{\theta}) \leq \dim(\Phi(\mathbf{x}))$

Optimisation Monte Carlo (Meeds and Welling, NIPS 2015)

- Ingredients:
  - Distance $d(\mathbf{x}, \mathbf{x}_o) = ||\Phi(\mathbf{x}) - \Phi(\mathbf{x}_o)||_2$ with known Φ.
  - Search uses optimisation, which leads to increased efficiency.
- Assumptions:
  - (approximate) derivative of $\Phi(\mathbf{x}) = \Phi(g(\boldsymbol{\theta}, \mathbf{u})) = \mathbf{f}(\boldsymbol{\theta}, \mathbf{u})$ wrt $\boldsymbol{\theta}$ is available
  - $\dim(\boldsymbol{\theta}) \leq \dim(\Phi(\mathbf{x}))$
- Algorithm to generate n weighted samples $\boldsymbol{\theta}_i^*$:

1: **for** $i \leftarrow 1$ to n **do**
2: $\quad \mathbf{u}_i \sim p(\mathbf{u})$ ▷ Set seed
3: $\quad \boldsymbol{\theta}_i^* = \arg\min_{\boldsymbol{\theta}} ||\mathbf{f}(\boldsymbol{\theta}, \mathbf{u}_i) - \Phi(\mathbf{x}_o)||$ ▷ Optimisation
4: $\quad$ Compute $\mathbf{J}_i$ with columns $\partial \mathbf{f}(\boldsymbol{\theta}_i^*, \mathbf{u}_i)/\partial \theta_k$
5: $\quad$ Compute $w_i = p(\boldsymbol{\theta}_i^*) * (\det(\mathbf{J}_i^\top \mathbf{J}_i))^{-1/2}$
6: $\quad$ Accept $\boldsymbol{\theta}_i^*$ as posterior sample with weight w_i

(Note that samples with too large final distances may be omitted.)

Contents

Application: Vision as inverse graphics

- Implicit model given by a graphics renderer
- We used Open Differential Renderer (Loper and Black, 2014)

20 parameters:
- Shape
- Rotation/Pose
- Illumination
- Colour

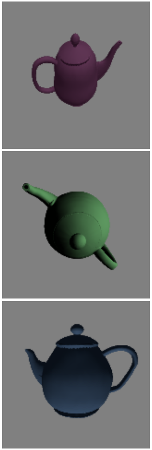
Renderer

(forward problem)

$\rightleftharpoons$

Inference

(inverse problem)



Why Bayesian inference and not point-estimation?

- In some cases, quantifying uncertainty is very important
- The inverse problem may have multiple solutions (posterior may be multi-modal)

  Example considered: Infer object colour when external lighting conditions are unknown.

(a) Gray teapot under red light.

(b) Red teapot under white light.

Results for colour inference task

- We used OMC and the (simpler) rejection ABC algorithm.

Results for colour inference task

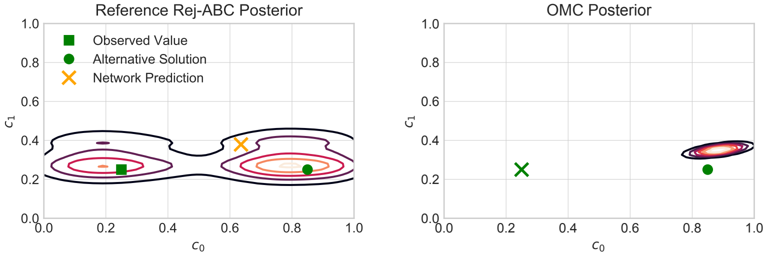
- We used OMC and the (simpler) rejection ABC algorithm.
- Rejection ABC relies on trial and error instead of optimisation to determine the posterior samples. Slow but reliable.

Results for colour inference task

- We used OMC and the (simpler) rejection ABC algorithm.
- Rejection ABC relies on trial and error instead of optimisation to determine the posterior samples. Slow but reliable.
- Same distance function $d(\mathbf{x}, \mathbf{x}_o)$: Euclidean distance between parameter predictions made by a neural network trained on images generated from the renderer under white light.

Results for colour inference task

- We used OMC and the (simpler) rejection ABC algorithm.
- Rejection ABC relies on trial and error instead of optimisation to determine the posterior samples. Slow but reliable.
- Same distance function $d(\mathbf{x}, \mathbf{x}_o)$: Euclidean distance between parameter predictions made by a neural network trained on images generated from the renderer under white light.
- Posteriors for two colours c_0 and c_1 (red and green):

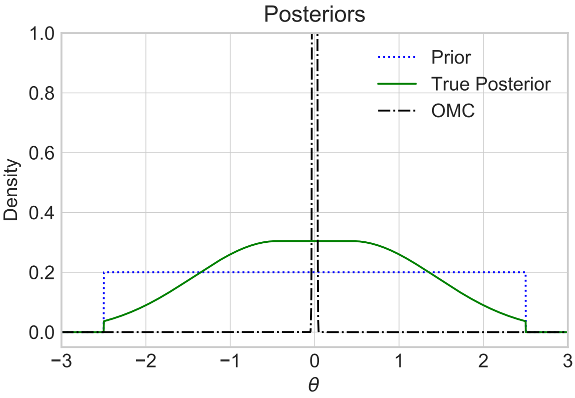


Why did OMC fail?

- The OMC weights $w_i = p(\boldsymbol{\theta}_i^*) * (\det(\mathbf{J}_i{}^\top \mathbf{J}_i))^{-1/2}$ are unstable (ESS was 1.2!)
- This happens when the (approximate) likelihood function has nearly flat regions so that $\det(\mathbf{J}_i{}^\top \mathbf{J}_i) \approx 0$.

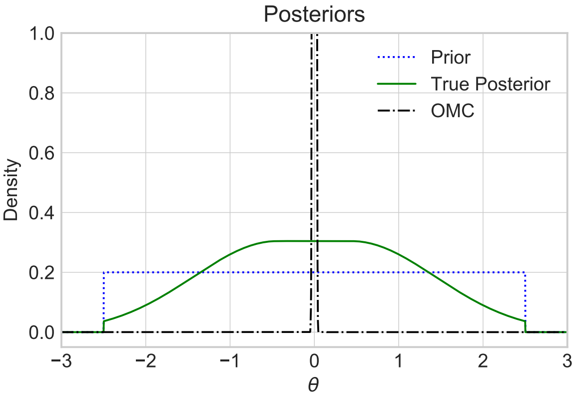
Why did OMC fail?

- The OMC weights $w_i = p(\boldsymbol{\theta}_i^*) * (\det(\mathbf{J}_i^\top \mathbf{J}_i))^{-1/2}$ are unstable (ESS was 1.2!)
- This happens when the (approximate) likelihood function has nearly flat regions so that $\det(\mathbf{J}_i^\top \mathbf{J}_i) \approx 0$.



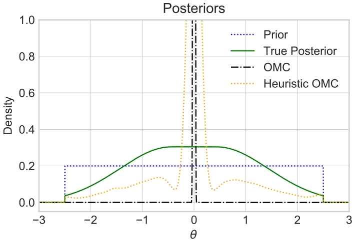
Why did OMC fail?

- The OMC weights $w_i = p(\boldsymbol{\theta}_i^*) * (\det(\mathbf{J}_i^\top \mathbf{J}_i))^{-1/2}$ are unstable (ESS was 1.2!)
- This happens when the (approximate) likelihood function has nearly flat regions so that $\det(\mathbf{J}_i^\top \mathbf{J}_i) \approx 0$.
  Note: stated OMC assumptions are not violated.

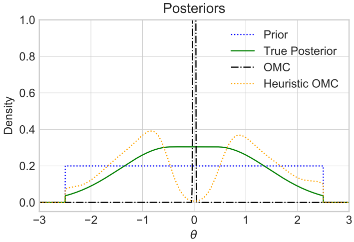


Stabilising the weights/matrices does not help

- Taking the pseudo-inverse or pseudo-determinant of $\mathbf{J}_i^\top \mathbf{J}_i$ does not help.



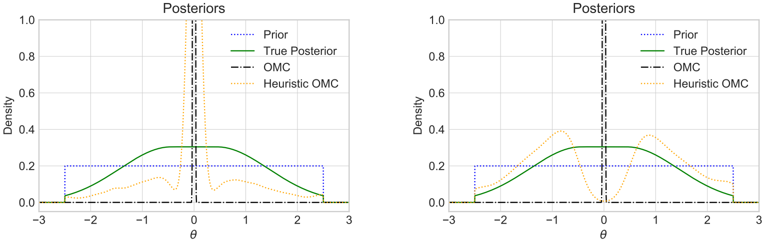
(a) Pseudo-inverse



(b) Pseudo-determinant

Stabilising the weights/matrices does not help

- Taking the pseudo-inverse or pseudo-determinant of $\mathbf{J}_i^\top \mathbf{J}_i$ does not help.
- The weights are not the real issue. The problem is more fundamental:
  OMC uses a single point to represent an entire region where the likelihood is (nearly) constant.



(a) Pseudo-inverse (b) Pseudo-determinant

Contents

Key properties of ROMC

(Ikonomov and Gutmann, *arXiv:1904.00670*, 2019)

1. Fixes OMC's failure case: It handles likelihood functions that are (nearly) flat on significant regions in parameter space.

2. Works for general distance functions $d(g(\boldsymbol{\theta}, \mathbf{u}), \mathbf{x}_o)$ and not only Euclidean distances between summary statistics.
   (condition $\dim(\boldsymbol{\theta}) \leq \dim(\Phi(\mathbf{x}))$ disappears)

3. Does not require (approximate) derivatives, while OMC does.

4. Can be run as post-processing to OMC or from scratch.

The ROMC framework

ROMC is a framework for inference. It has three key steps:

1. For $i = 1, \dots n'$, sample $\mathbf{u}_i \sim p(\mathbf{u})$ and determine

$$\boldsymbol{\theta}_i^* = \arg\min_{\boldsymbol{\theta}} d(g(\boldsymbol{\theta}, \mathbf{u}_i), \mathbf{x}_o)$$

   Same as in OMC but we can use general distances $d(\mathbf{x}, \mathbf{x}_o)$.

The ROMC framework

ROMC is a framework for inference. It has three key steps:

1. For $i = 1, \dots n'$, sample $\mathbf{u}_i \sim p(\mathbf{u})$ and determine

$$\boldsymbol{\theta}_i^* = \arg\min_{\boldsymbol{\theta}} d(g(\boldsymbol{\theta}, \mathbf{u}_i), \mathbf{x}_o)$$

   Same as in OMC but we can use general distances $d(\mathbf{x}, \mathbf{x}_o)$.

2. Use the minimal distances $d_i^* = d(g(\boldsymbol{\theta}_i^*, \mathbf{u}_i))$ to choose an acceptance threshold ϵ / keep the n best $\boldsymbol{\theta}_i^*$.

The ROMC framework

ROMC is a framework for inference. It has three key steps:

1. For $i = 1, \dots n'$, sample $\mathbf{u}_i \sim p(\mathbf{u})$ and determine

$$\boldsymbol{\theta}_i^* = \arg\min_{\boldsymbol{\theta}} d(g(\boldsymbol{\theta}, \mathbf{u}_i), \mathbf{x}_o)$$

   Same as in OMC but we can use general distances $d(\mathbf{x}, \mathbf{x}_o)$.

2. Use the minimal distances $d_i^* = d(g(\boldsymbol{\theta}_i^*, \mathbf{u}_i))$ to choose an acceptance threshold ϵ / keep the n best $\boldsymbol{\theta}_i^*$.
3. For each i where $d_i^* \leq \epsilon$, define a proposal distribution q_i on the "acceptance region" $C_\epsilon^i = \{\boldsymbol{\theta} : d(g(\boldsymbol{\theta}, \mathbf{u}_i), \mathbf{x}_o) \leq \epsilon\}$

The ROMC framework

ROMC is a framework for inference. It has three key steps:

1. For $i = 1, \ldots n'$, sample $\mathbf{u}_i \sim p(\mathbf{u})$ and determine

$$\boldsymbol{\theta}_i^* = \arg\min_{\boldsymbol{\theta}} d(g(\boldsymbol{\theta}, \mathbf{u}_i), \mathbf{x}_o)$$

   Same as in OMC but we can use general distances $d(\mathbf{x}, \mathbf{x}_o)$.

2. Use the minimal distances $d_i^* = d(g(\boldsymbol{\theta}_i^*, \mathbf{u}_i))$ to choose an acceptance threshold ϵ / keep the n best $\boldsymbol{\theta}_i^*$.
3. For each i where $d_i^* \leq \epsilon$, define a proposal distribution q_i on the "acceptance region" $C_\epsilon^i = \{\boldsymbol{\theta} : d(g(\boldsymbol{\theta}, \mathbf{u}_i), \mathbf{x}_o) \leq \epsilon\}$

Approximate posterior is represented by weighted samples $\boldsymbol{\theta}_{ij}$:

$$\boldsymbol{\theta}_{ij} \sim q_i(\boldsymbol{\theta}) \qquad w_{ij} = \mathbb{1}_{C_\epsilon^i}(\boldsymbol{\theta}_{ij}) \frac{p(\boldsymbol{\theta}_{ij})}{q_i(\boldsymbol{\theta}_{ij})} \qquad \text{(i=1, \ldots n; \quad j=1, \ldots, m)}$$

Construction of the proposal distribution

(General idea, see paper for details)

- Using $\boldsymbol{\theta}_i^*$ and the optimisation trajectory, we build a model of the acceptance regions $C_\epsilon^i = \{\boldsymbol{\theta} : d(g(\boldsymbol{\theta}, \mathbf{u}_i), \mathbf{x}_o) \leq \epsilon\}$

Construction of the proposal distribution

(General idea, see paper for details)

- Using $\boldsymbol{\theta}_i^*$ and the optimisation trajectory, we build a model of the acceptance regions $C_\epsilon^i = \{\boldsymbol{\theta} : d(g(\boldsymbol{\theta}, \mathbf{u}_i), \mathbf{x}_o) \leq \epsilon\}$
- Simple but effective: model C_ϵ^i as a hypercube or ellipse and define q_i to be the uniform distribution on it.

Construction of the proposal distribution

(General idea, see paper for details)

- Using $\boldsymbol{\theta}_i^*$ and the optimisation trajectory, we build a model of the acceptance regions $C_\epsilon^i = \{\boldsymbol{\theta} : d(g(\boldsymbol{\theta}, \mathbf{u}_i), \mathbf{x}_o) \leq \epsilon\}$
- Simple but effective: model C_ϵ^i as a hypercube or ellipse and define q_i to be the uniform distribution on it.
- Note: When computing the weight,

$$w_{ij} = \mathbb{1}_{C_\epsilon^i}(\boldsymbol{\theta}_{ij}) \frac{p(\boldsymbol{\theta}_{ij})}{q_i(\boldsymbol{\theta}_{ij})}$$

the indicator function checks whether $\boldsymbol{\theta}_{ij}$ is in the true acceptance region C_ϵ^i. $\Rightarrow$ Some robustness to modelling errors.

Construction of the proposal distribution

(General idea, see paper for details)

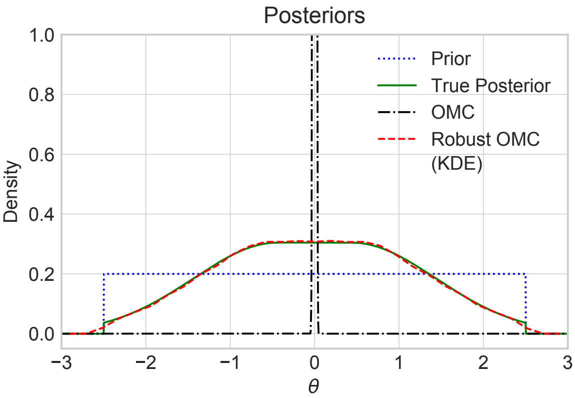
- Using $\boldsymbol{\theta}_i^*$ and the optimisation trajectory, we build a model of the acceptance regions $C_\epsilon^i = \{\boldsymbol{\theta} : d(g(\boldsymbol{\theta}, \mathbf{u}_i), \mathbf{x}_o) \leq \epsilon\}$
- Simple but effective: model C_ϵ^i as a hypercube or ellipse and define q_i to be the uniform distribution on it.
- Note: When computing the weight,

$$w_{ij} = \mathbb{1}_{C_\epsilon^i}(\boldsymbol{\theta}_{ij}) \frac{p(\boldsymbol{\theta}_{ij})}{q_i(\boldsymbol{\theta}_{ij})}$$

  the indicator function checks whether $\boldsymbol{\theta}_{ij}$ is in the true acceptance region C_ϵ^i. $\Rightarrow$ Some robustness to modelling errors.
- Check requires evaluating the distance $d(g(\boldsymbol{\theta}, \mathbf{u}_i), \mathbf{x}_o)$ and can be omitted/approximated to accelerate the inference.

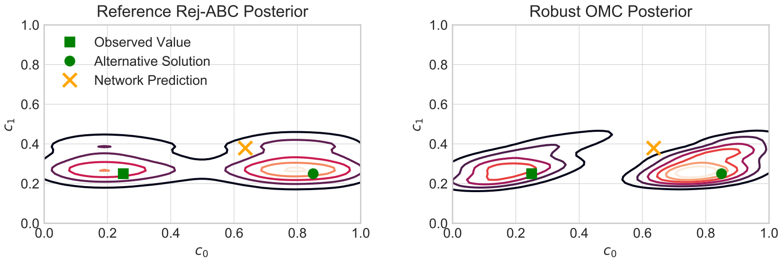
Results on the toy example

- Acceptance regions C_ϵ^i given by intervals on the line.
- ROMC handles the (nearly) flat likelihood function correctly.
- ROMC accurately represents uncertainty while OMC does not.



Results on the colour inference task

- Setup:
  - Optimisation via a gradient-free method (Bayesian optimisation with GP surrogate modelling)
  - Acceptance regions were modelled as ellipses (derived from the GP surrogate model)
- ROMC posterior matches reference posterior well.
- Effective sample size: 97% (vs. approx. 0.5% for OMC)



ROMC generalises OMC (see paper for the proof)

Theorem: Under the below assumptions, ROMC becomes equivalent to standard OMC as $\epsilon \to 0$.

Assumption 1. The distance $d(g(\boldsymbol{\theta}, \mathbf{u}), \mathbf{x}_o)$ is given by the Euclidean distance between summary statistics $||\mathbf{f}(\boldsymbol{\theta}, \mathbf{u}) - \Phi(\mathbf{x}_o)||$.

Assumption 2. The proposal distribution $q_i(\boldsymbol{\theta})$ is the uniform distribution on C^i_ϵ.

Assumption 3. The acceptance regions C^i_ϵ are approximated by the ellipsoid $C^i_\epsilon = \{\boldsymbol{\theta} : (\boldsymbol{\theta} - \boldsymbol{\theta}^*_i)^\top \mathbf{J}_i{}^\top \mathbf{J}_i(\boldsymbol{\theta} - \boldsymbol{\theta}^*_i) \leq \epsilon\}$ where $\mathbf{J}_i$ is the Jacobian matrix with columns $\partial \mathbf{f}(\boldsymbol{\theta}^*_i, \mathbf{u}_i)/\partial\theta_k$.

Assumption 4. The matrix square root $\mathbf{A}_i$ of $\mathbf{J}_i{}^\top \mathbf{J}_i$ is full rank, i.e. $\text{rank}(\mathbf{A}_i) = \dim(\boldsymbol{\theta})$.

Assumption 5. The prior $p(\boldsymbol{\theta})$ is constant on the acceptance regions C^i_ϵ.

Explanation of the failure case

Identified failure case is due to violation of Assumptions 3 and 4:

Assumption 3. The acceptance regions C_ϵ^i are approximated by the ellipsoid $C_\epsilon^i = \{\boldsymbol{\theta} : (\boldsymbol{\theta} - \boldsymbol{\theta}_i^*)^\top \mathbf{J}_i{}^\top \mathbf{J}_i (\boldsymbol{\theta} - \boldsymbol{\theta}_i^*) \leq \epsilon\}$ where $\mathbf{J}_i$ is the Jacobian matrix with columns $\partial \mathbf{f}(\boldsymbol{\theta}_i^*, \mathbf{u}_i)/\partial \theta_k$.

Assumption 4. The matrix square root $\mathbf{A}_i$ of $\mathbf{J}_i{}^\top \mathbf{J}_i$ is full rank, i.e. $\text{rank}(\mathbf{A}_i) = \dim(\boldsymbol{\theta})$.

Explanation of the failure case

Identified failure case is due to violation of Assumptions 3 and 4:

Assumption 3. The acceptance regions C_ϵ^i are approximated by the ellipsoid $C_\epsilon^i = \{\boldsymbol{\theta} : (\boldsymbol{\theta} - \boldsymbol{\theta}_i^*)^\top \mathbf{J}_i{}^\top \mathbf{J}_i(\boldsymbol{\theta} - \boldsymbol{\theta}_i^*) \leq \epsilon\}$ where $\mathbf{J}_i$ is the Jacobian matrix with columns $\partial \mathbf{f}(\boldsymbol{\theta}_i^*, \mathbf{u}_i)/\partial \theta_k$.

Assumption 4. The matrix square root $\mathbf{A}_i$ of $\mathbf{J}_i{}^\top \mathbf{J}_i$ is full rank, i.e. $\text{rank}(\mathbf{A}_i) = \dim(\boldsymbol{\theta})$.

For non-uniform priors, one then also risks violating Assumption 5:
Assumption 5. The prior $p(\boldsymbol{\theta})$ is constant on the acceptance regions C_ϵ^i.

Conclusions

- Talk was on Bayesian inference for implicit models.

Conclusions

- Talk was on Bayesian inference for implicit models.
  - Implicit models: models that are defined by a data generating process.

Conclusions

- Talk was on Bayesian inference for implicit models.
  - Implicit models: models that are defined by a data generating process.
  - Optimisation Monte Carlo (OMC): Bayesian inference method that uses optimisation to increase computational efficiency.

Conclusions

- Talk was on Bayesian inference for implicit models.
  - Implicit models: models that are defined by a data generating process.
  - Optimisation Monte Carlo (OMC): Bayesian inference method that uses optimisation to increase computational efficiency.
- We showed that OMC under-estimates posterior uncertainty by collapsing regions of near-constant likelihood into a point.

Conclusions

- Talk was on Bayesian inference for implicit models.
  - Implicit models: models that are defined by a data generating process.
  - Optimisation Monte Carlo (OMC): Bayesian inference method that uses optimisation to increase computational efficiency.
- We showed that OMC under-estimates posterior uncertainty by collapsing regions of near-constant likelihood into a point.
- We proposed a robust generalisation of OMC, robust OMC, that explains and corrects this failure mode while maintaining OMC's benefits due to optimisation.