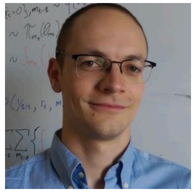
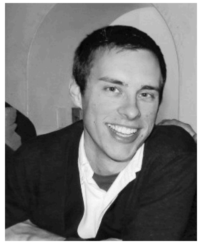
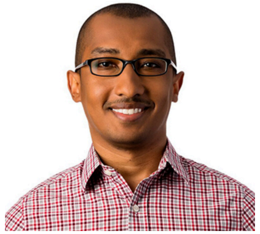
Stein's method as a computational tool



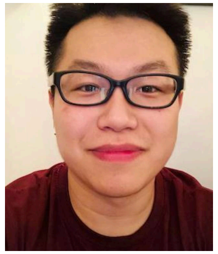











[+ many more who don't fit on slide…]

Dr François-Xavier Briol
Department of Statistical Science
University College London

A summary of this course

Statistical Science
2023, Vol. 38, No. 1, 120–139
https://doi.org/10.1214/22-STS863

Stein’s Method Meets Computational Statistics: A Review of Some Recent Developments

Andreas Anastasiou, Alessandro Barp, François-Xavier Briol, Bruno Ebner, Robert E. Gaunt, Fatemeh Ghaderinezhad, Jackson Gorham, Arthur Gretton, Christophe Ley, Qiang Liu, Lester Mackey, Chris J. Oates, Gesine Reinert and Yvik Swan

A summary of this course

Statistical Science
2023, Vol. 38, No. 1, 120–139
https://doi.org/10.1214/22-STS863

Stein's Method Meets Computational Statistics: A Review of Some Recent Developments

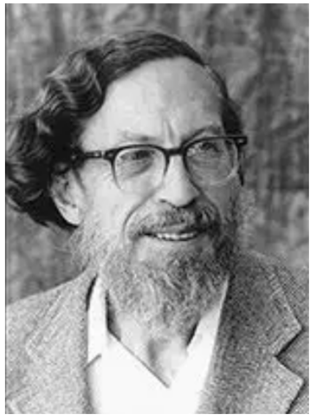
Andreas Anastasiou, Alessandro Barp, François-Xavier Briol, Bruno Ebner, Robert E. Gaunt, Fatemeh Ghaderinezhad, Jackson Gorham, Arthur Gretton, Christophe Ley, Qiang Liu, Lester Mackey, Chris J. Oates, Gesine Reinert and Yvik Swan

We started with two authors back in 2018 then things got out of hands….

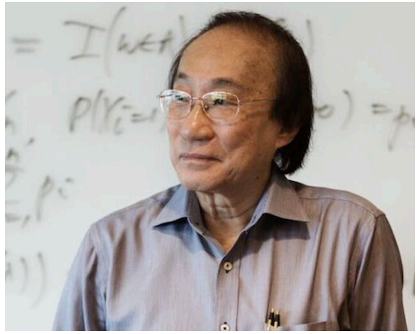
The birth of Stein's method (1972)

A BOUND FOR THE ERROR IN THE NORMAL APPROXIMATION TO THE DISTRIBUTION OF A SUM OF DEPENDENT RANDOM VARIABLES

CHARLES STEIN
STANFORD UNIVERSITY



Prof. Charles Stein (Stanford)

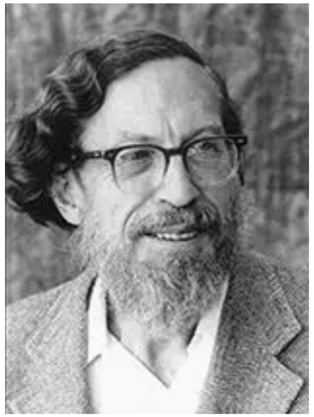


Prof. Louis Chen (NUS)

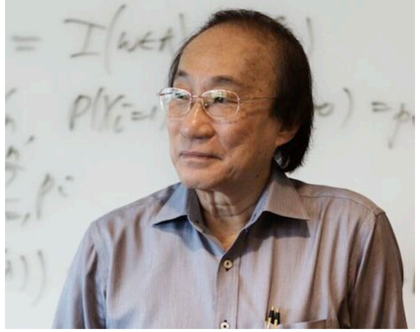
The birth of Stein's method (1972)

A BOUND FOR THE ERROR IN THE NORMAL APPROXIMATION TO THE DISTRIBUTION OF A SUM OF DEPENDENT RANDOM VARIABLES

CHARLES STEIN
STANFORD UNIVERSITY



Prof. Charles Stein (Stanford)



Prof. Louis Chen (NUS)

<u>1972-2015:</u> most of the focus is on probability theory of theoretical statistics.

Stein goes computational (2015-…)

arXiv 2014 - JRSSB 2017

J. R. Statist. Soc. B (2017)
79, *Part* 3, *pp.* 695–718

Control functionals for Monte Carlo integration

Chris J. Oates,
University of Technology Sydney, Australia

Mark Girolami
University of Warwick, Coventry, and Alan Turing Institute, London, UK

and Nicolas Chopin
Centre de Recherche en Economie et Statistique and Ecole Nationale de la Statistique et de l'Administration Economique, Paris, France

NeurIPS 2015

Measuring Sample Quality with Stein's Method

Jackson Gorham
Department of Statistics
Stanford University

Lester Mackey
Department of Statistics
Stanford University

We will now discuss why Stein characterisations are so useful…

Stein goes computational (2015-…)

arXiv 2014 - JRSSB 2017

J. R. Statist. Soc. B (2017)
79, Part 3, pp. 695–718

Control functionals for Monte Carlo integration

Chris J. Oates,
University of Technology Sydney, Australia

Mark Girolami
University of Warwick, Coventry, and Alan Turing Institute, London, UK

and Nicolas Chopin
Centre de Recherche en Economie et Statistique and Ecole Nationale de la Statistique et de l'Administration Economique, Paris, France

NeurIPS 2015

Measuring Sample Quality with Stein's Method

Jackson Gorham
Department of Statistics
Stanford University

Lester Mackey
Department of Statistics
Stanford University

ICML 2016

A Kernel Test of Goodness of Fit

Kacper Chwialkowski* KACPER.CHWIALKOWSKI@GMAIL.COM
Heiko Strathmann* HEIKO.STRATHMANN@GMAIL.COM
Arthur Gretton ARTHUR.GRETTON@GMAIL.COM
Gatsby Unit, University College London, United Kingdom

ICML 2016

A Kernelized Stein Discrepancy for Goodness-of-fit Tests

Qiang Liu QLIU@CS.DARTMOUTH.EDU
Computer Science, Dartmouth College, NH, 03755

Jason D. Lee JASONDLEE88@EECS.BERKELEY.EDU
Michael Jordan JORDAN@CS.BERKELEY.EDU
Department of Electrical Engineering and Computer Science University of California, Berkeley, CA 94709

We will now discuss why Stein characterisations are so useful…

Some exciting new algorithms!

Stein's Method

Some exciting new algorithms!

Hypothesis testing

Stein's Method

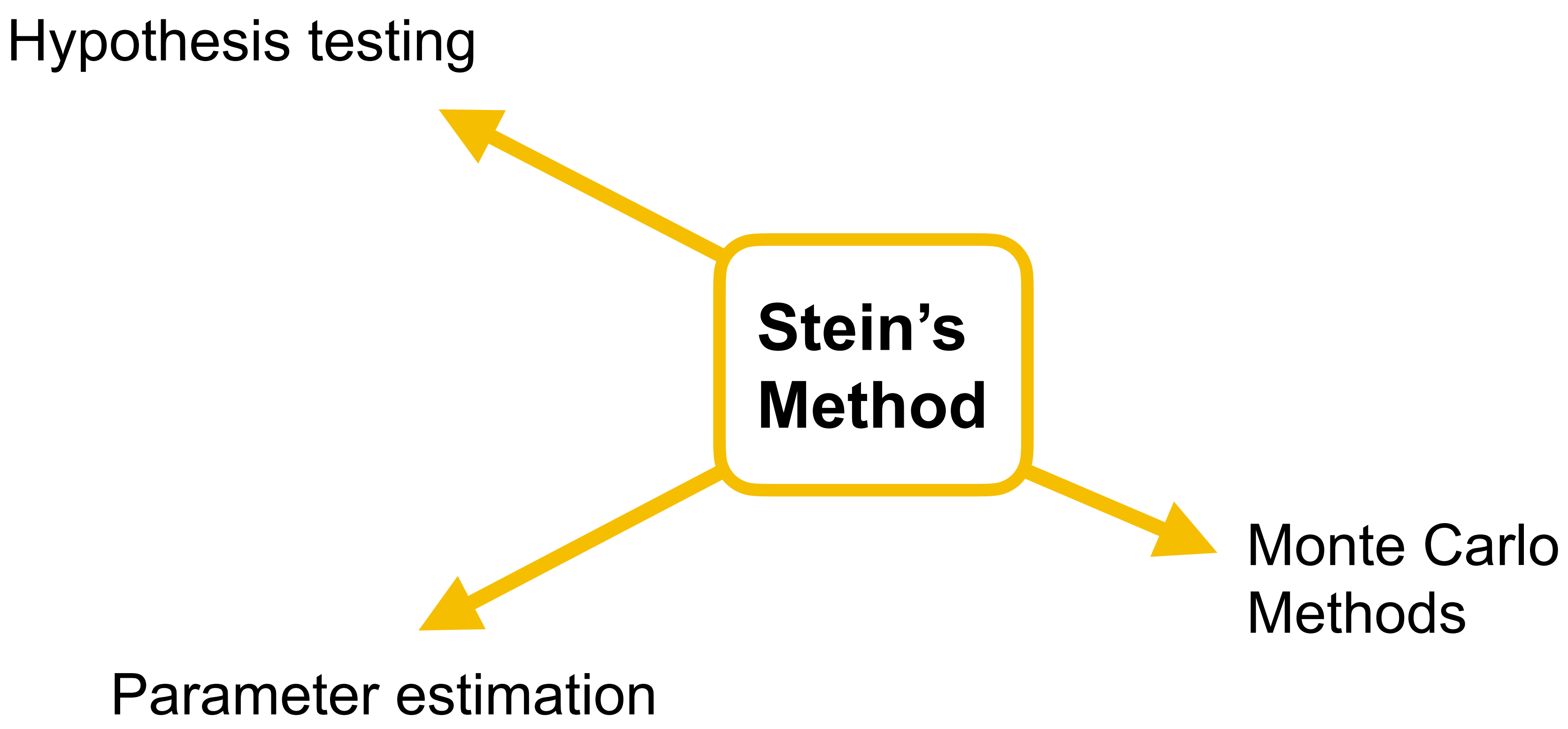
Some exciting new algorithms!

Hypothesis testing

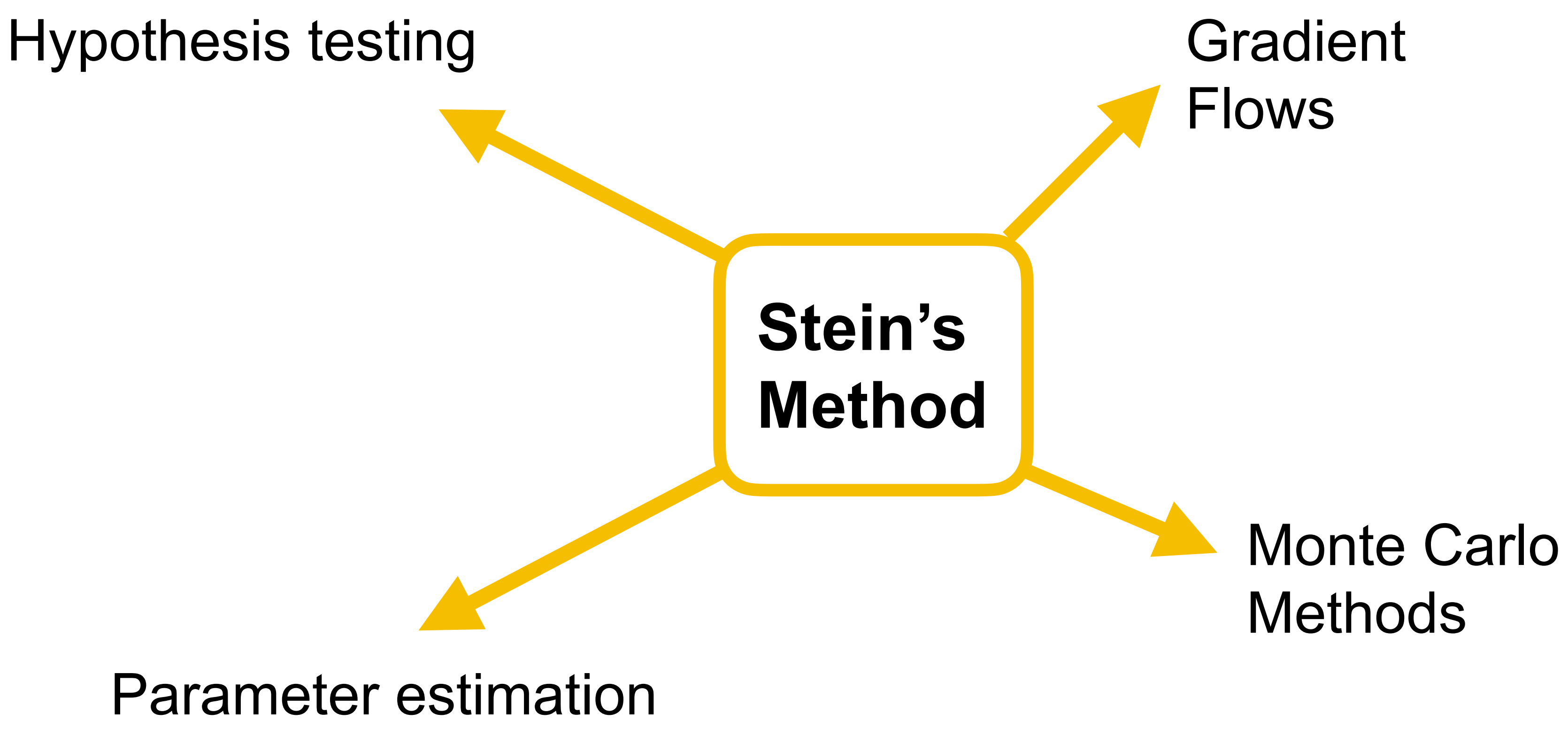
Stein's Method

Monte Carlo Methods

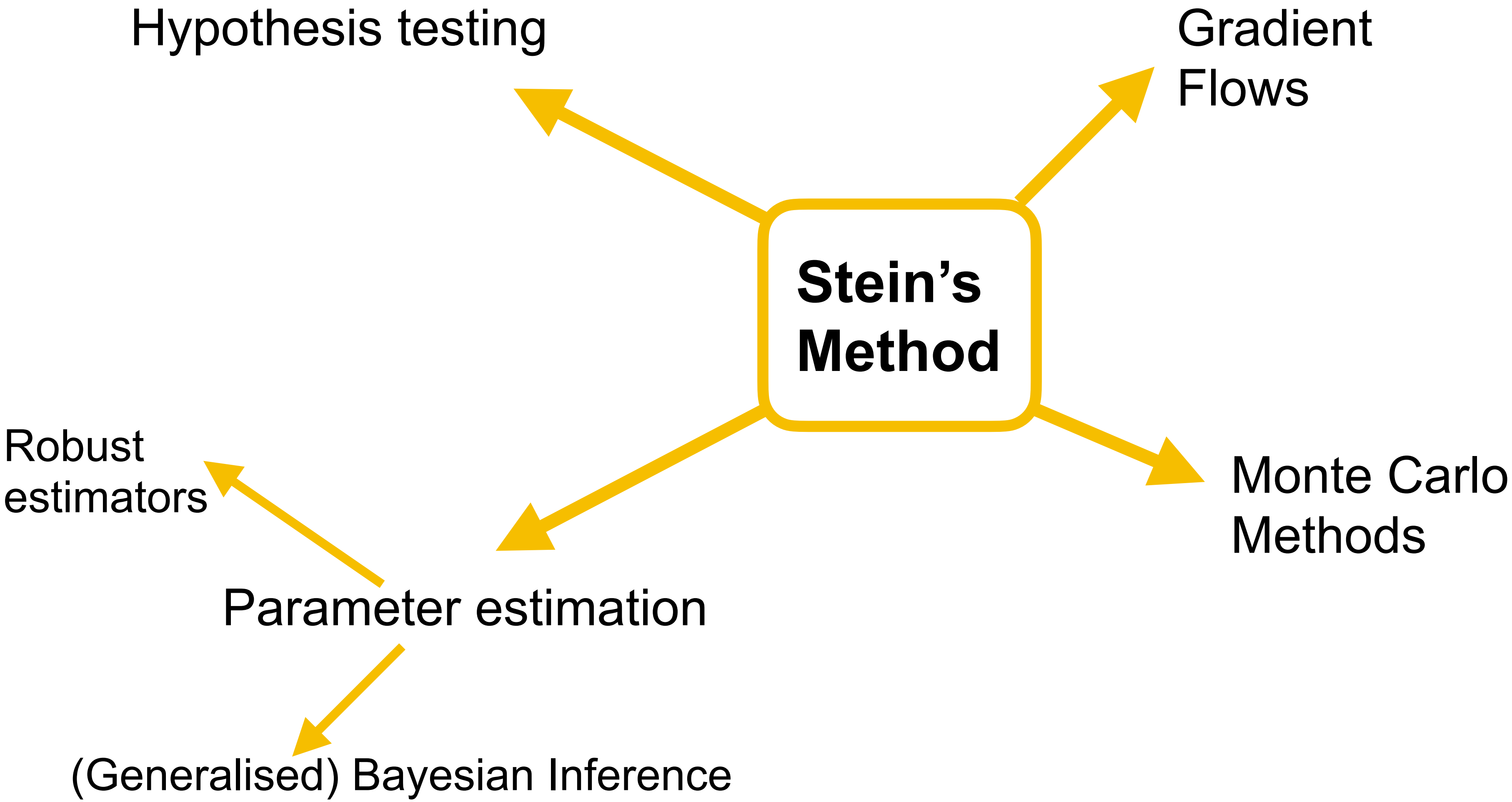
Some exciting new algorithms!



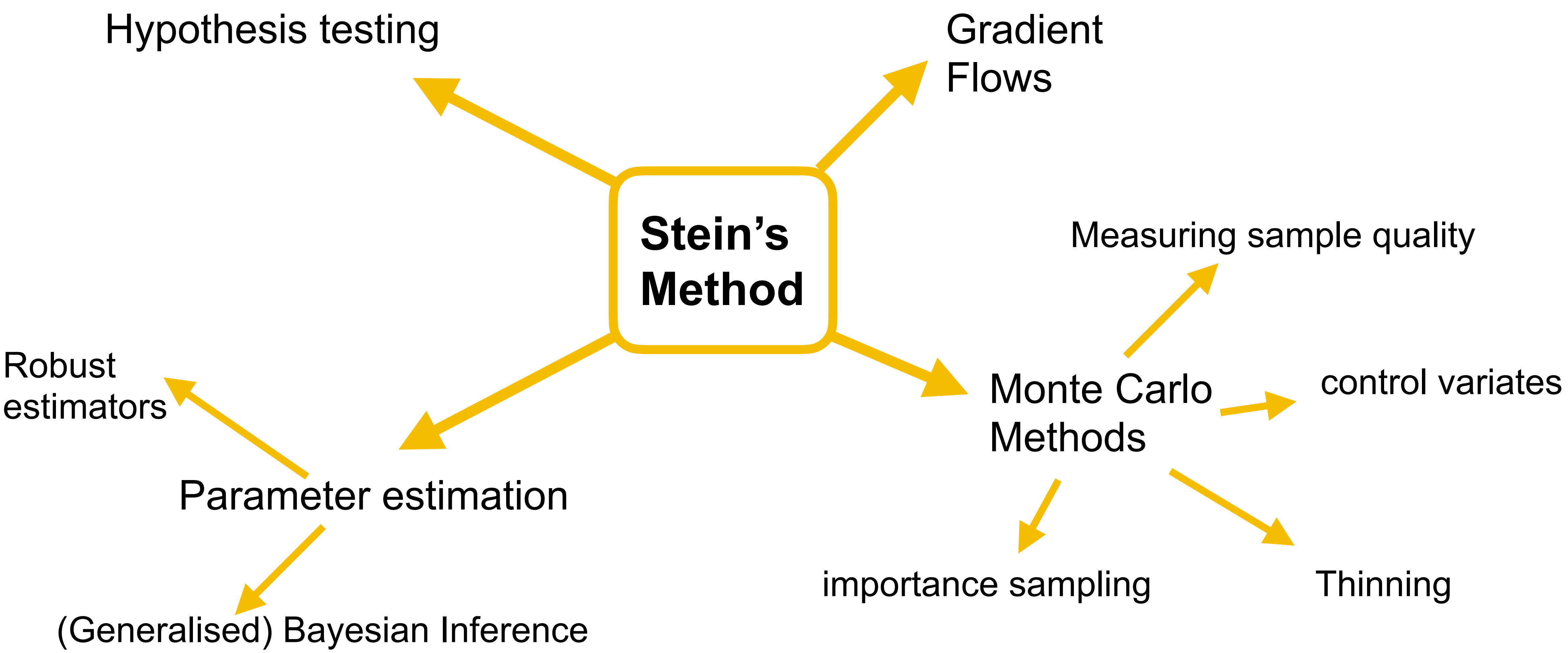
Some exciting new algorithms!



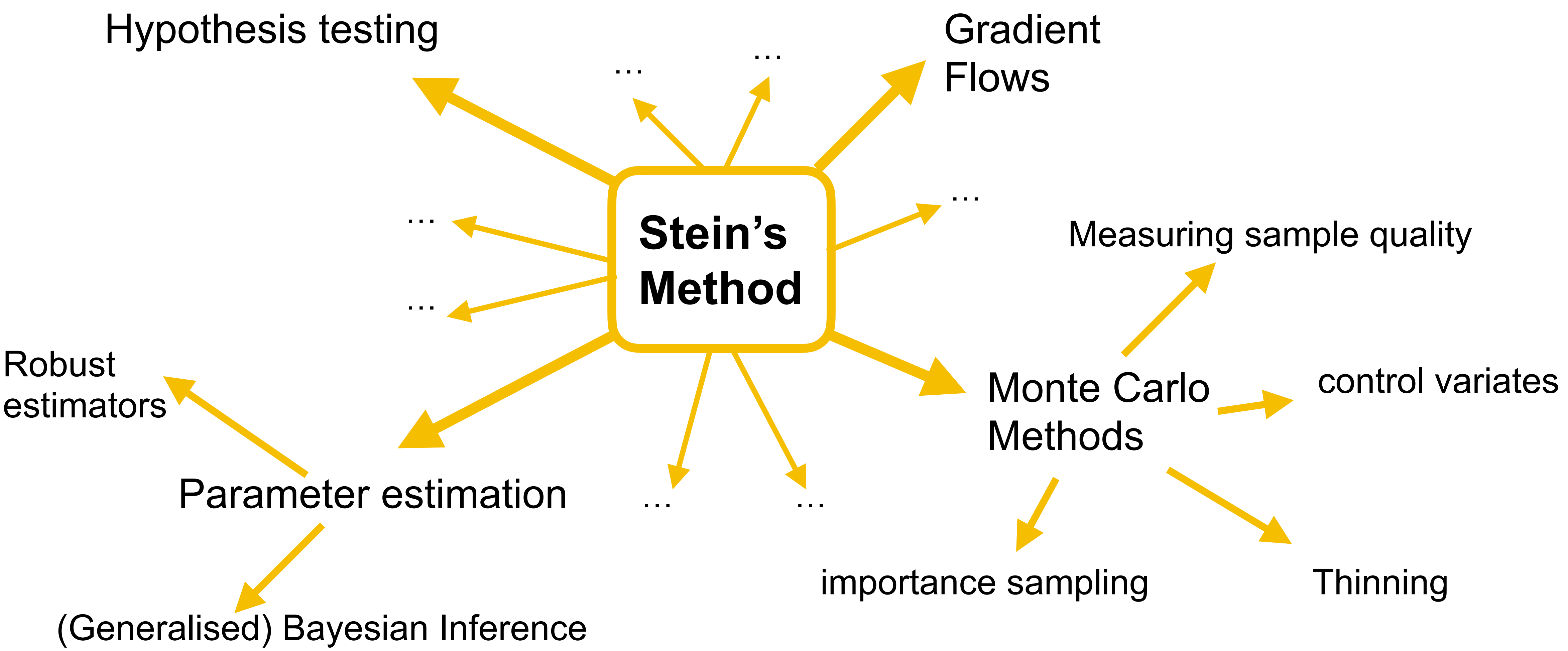
Some exciting new algorithms!



Some exciting new algorithms!



Some exciting new algorithms!



Outline

- What is Stein's method, and why should you care…
- Computational tools based on Stein's method.
- Some nice (new) algorithms!

Stein’s method as a computational tool

Stein characterisations

Uncertainty through distributions

- Our job, as statisticians, is to help make sense of the world around us by collecting and analysis data, and making conclusions from this.
- This is hard as we typically only have limited data, and we therefore need to be careful in how we **represent and communicate uncertainty**!

Uncertainty through distributions

- Our job, as statisticians, is to help make sense of the world around us by collecting and analysis data, and making conclusions from this.
- This is hard as we typically only have limited data, and we therefore need to be careful in how we **represent and communicate uncertainty**!

- In statistics, we typically represent uncertainty through **probability distributions**.

Some popular characterisations

- Our language for representing uncertainty is therefore a language for representing probability distributions.

Some popular characterisations

- Our language for representing uncertainty is therefore a language for representing probability distributions.

- We have many different ways of representing a probability distribution P:

CDF: $$F(x) = \mathbb{E}_{X \sim P}\left[1_{\{X \leq x\}}\right]$$

Some popular characterisations

- Our language for representing uncertainty is therefore a language for representing probability distributions.

- We have many different ways of representing a probability distribution P:

CDF: $$F(x) = \mathbb{E}_{X \sim P}\left[1_{\{X \leq x\}}\right]$$

PDF: $$p(x) = \frac{dF(x)}{dx}$$

Some popular characterisations

- Our language for representing uncertainty is therefore a language for representing probability distributions.

- We have many different ways of representing a probability distribution P:

CDF: $$F(x) = \mathbb{E}_{X \sim P}\left[1_{\{X \leq x\}}\right]$$

PDF: $$p(x) = \frac{dF(x)}{dx}$$

MGF: $$M(t) = \mathbb{E}_{X \sim P}[\exp(tX)]$$

Some popular characterisations

- Our language for representing uncertainty is therefore a language for representing probability distributions.

- We have many different ways of representing a probability distribution P:

CDF: $F(x) = \mathbb{E}_{X \sim P}\left[1_{\{X \leq x\}}\right]$

PDF: $p(x) = \dfrac{dF(x)}{dx}$

MGF: $M(t) = \mathbb{E}_{X \sim P}[\exp(tX)]$

CF: $\varphi(t) = \mathbb{E}_{X \sim P}[\exp(itX)]$

…

Difference between characterisations

- **Q:** *“Why do we need so many ways of describing probability distributions?”*

Difference between characterisations

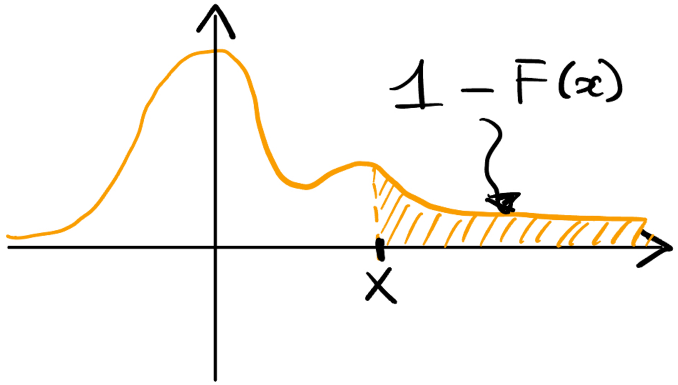
- **Q:** *"Why do we need so many ways of describing probability distributions?"*

- **A:** They each give us a **mathematical language** to work with probability distributions, and sometime expressing yourself in one language is easier than doing so with another.

Characterisations in probability theory

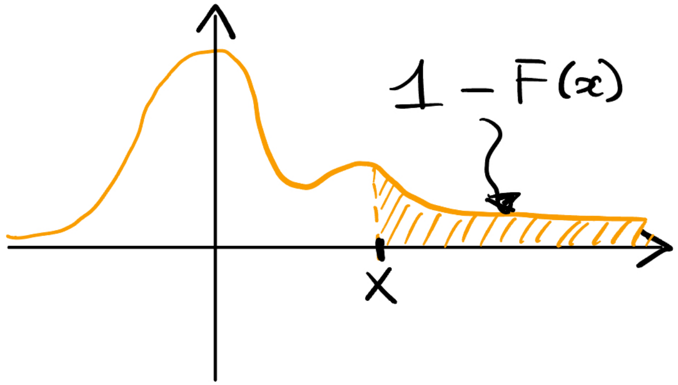
Characterisations in probability theory

- **Example 1:** Cumulative distribution functions are great for computing tail probabilities $P(X \geq x) = 1 - F(x)$, but really terrible for representing multivariate distributions!



Characterisations in probability theory

- **Example 1:** Cumulative distribution functions are great for computing tail probabilities $P(X \geq x) = 1 - F(x)$, but really terrible for representing multivariate distributions!



- **Example 2:** Characteristic functions are the expectation of a complex function and so not very interpretable, but their properties make the proof of the central limit theorem much easier!

$$\sqrt{n}\left(\frac{1}{n}\sum_{i=1}^{n} X_i - \mathbb{E}_{X\sim P}[X]\right) \to \mathcal{N}(0,\sigma^2)$$

Characterisations in statistics and machine learning

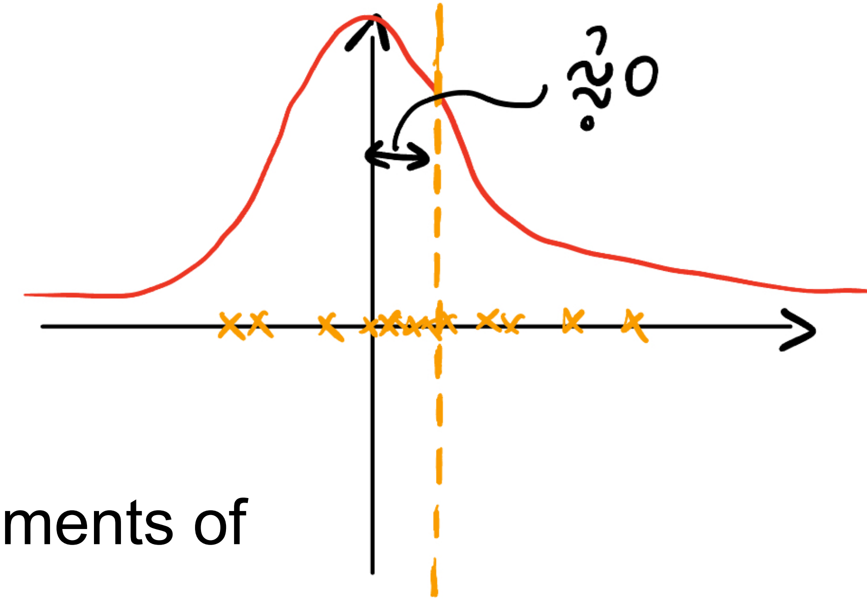
- **Example 1:** The moment generating function $M(t)$ is convenient for hypothesis testing or parameter estimation:

$$\mathbb{E}_{X\sim P}[X^n] = \left.\frac{d^n M(t)}{dt^n}\right|_{t=0}$$

Characterisations in statistics and machine learning

- **Example 1:** The moment generating function $M(t)$ is convenient for hypothesis testing or parameter estimation:

$$\mathbb{E}_{X\sim P}[X^n] = \frac{d^n M(t)}{dt^n}\bigg|_{t=0}$$

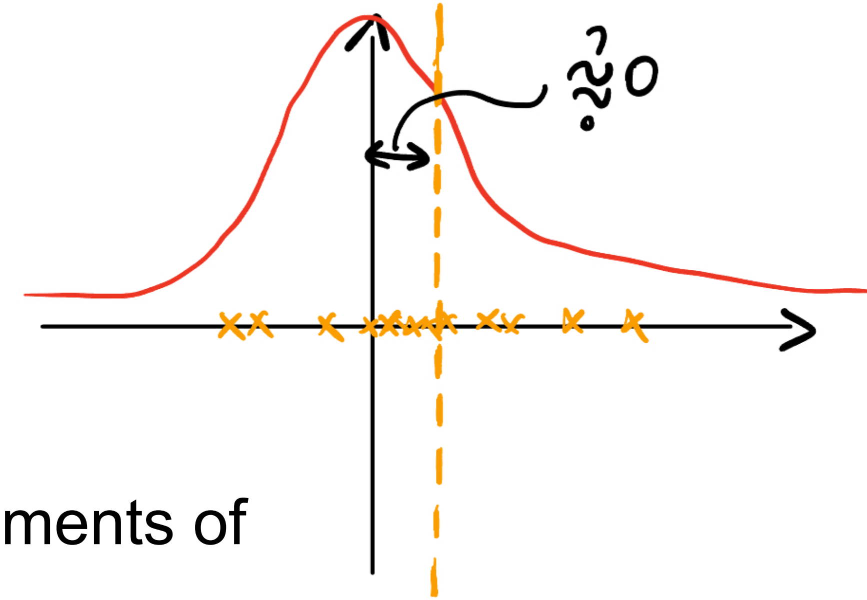


- We can check if $\{x_i\}_{i=1}^n \sim P$ by checking whether moments of $\{x_i\}_{i=1}^n$ are close to those of P!

Characterisations in statistics and machine learning

- **Example 1:** The moment generating function $M(t)$ is convenient for hypothesis testing or parameter estimation:

$$\mathbb{E}_{X\sim P}[X^n] = \left.\frac{d^n M(t)}{dt^n}\right|_{t=0}$$



- We can check if $\{x_i\}_{i=1}^n \sim P$ by checking whether moments of $\{x_i\}_{i=1}^n$ are close to those of P!

- We will now see yet another characterisation….

Stein characterisation

- A **Stein characterisation** for P is a pair $(\mathcal{S}_P, \mathcal{G}_P)$ such that

$$Q = P \quad \Leftrightarrow \quad \mathbb{E}_{X\sim Q}[\mathcal{S}_P[g](X)] = 0 \quad \forall g \in \mathcal{G}_P$$

Stein characterisation

Stein operator

Stein class

- A **Stein characterisation** for P is a pair $(\mathcal{S}_P, \mathcal{G}_P)$ such that

$$Q = P \quad \Leftrightarrow \quad \mathbb{E}_{X \sim Q}[\mathcal{S}_P[g](X)] = 0 \quad \forall g \in \mathcal{G}_P$$

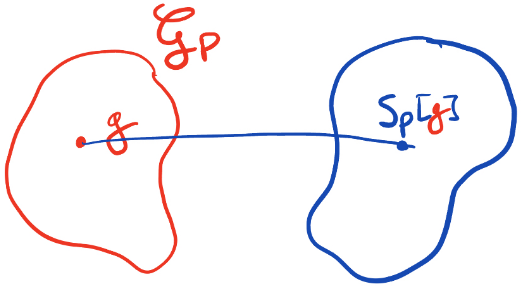
Stein characterisation

Stein operator

Stein class

- A **Stein characterisation** for P is a pair $(\mathcal{S}_P, \mathcal{G}_P)$ such that

$$Q = P \quad \Leftrightarrow \quad \mathbb{E}_{X \sim Q}[\mathcal{S}_P[g](X)] = 0 \quad \forall g \in \mathcal{G}_P$$



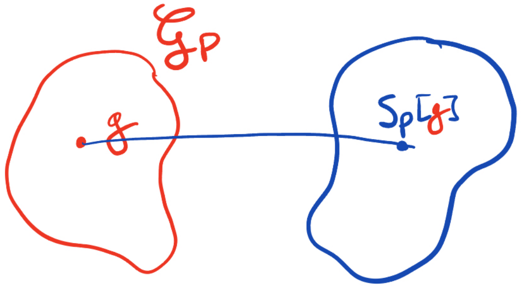
Stein characterisation

Stein operator

Stein class

- A **Stein characterisation** for P is a pair $(\mathcal{S}_P, \mathcal{G}_P)$ such that

$$Q = P \quad \Leftrightarrow \quad \mathbb{E}_{X\sim Q}[\mathcal{S}_P[g](X)] = 0 \quad \forall g \in \mathcal{G}_P$$



- In other words, you are now **representing** P **with an entire family of functions** with some peculiar property:

$$\{h : h(x) = \mathcal{S}_P[g](x), g \in \mathcal{G}_P\}$$

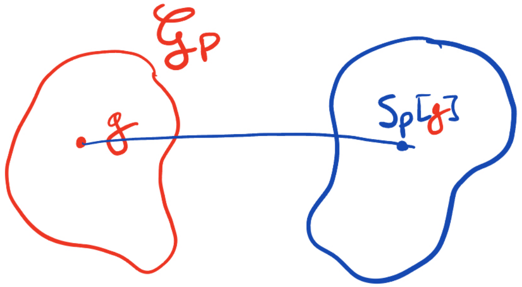
Stein characterisation

Stein operator

Stein class

- A **Stein characterisation** for P is a pair $(\mathcal{S}_P, \mathcal{G}_P)$ such that

$$Q = P \quad \Leftrightarrow \quad \mathbb{E}_{X \sim Q}[\mathcal{S}_P[g](X)] = 0 \quad \forall g \in \mathcal{G}_P$$



- In other words, you are now **representing P with an entire family of functions** with some peculiar property:

$$\left\{h : h(x) = \mathcal{S}_P[g](x), g \in \mathcal{G}_P\right\}$$

- (At this point I want to clarify I am not a sadistic mathematician…. We will see why shortly, but first lets see some examples…)

Characterising a $\mathcal{N}(0, \sigma^2)$

- At some point in your BSc/MSc/PhD, you have probably come across the many characterisations of a Gaussian:

$$p(x) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{x^2}{2\sigma^2}\right)$$

Characterising a $\mathcal{N}(0,\sigma^2)$

- At some point in your BSc/MSc/PhD, you have probably come across the many characterisations of a Gaussian:

$$p(x) = \frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{x^2}{2\sigma^2}\right) \qquad F(x) = \frac{1}{2}\left(1 + \text{Erf}\left(\frac{x}{\sqrt{2}\sigma}\right)\right)$$

Characterising a $\mathcal{N}(0,\sigma^2)$

- At some point in your BSc/MSc/PhD, you have probably come across the many characterisations of a Gaussian:

$$p(x) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{x^2}{2\sigma^2}\right) \qquad F(x) = \frac{1}{2}\left(1 + \text{Erf}\left(\frac{x}{\sqrt{2}\sigma}\right)\right)$$

$$M(t) = \exp\left(\frac{\sigma^2 t^2}{2}\right)$$

Characterising a $\mathcal{N}(0,\sigma^2)$

- At some point in your BSc/MSc/PhD, you have probably come across the many characterisations of a Gaussian:

$$p(x) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{x^2}{2\sigma^2}\right) \qquad F(x) = \frac{1}{2}\left(1 + \mathrm{Erf}\left(\frac{x}{\sqrt{2}\sigma}\right)\right)$$

$$M(t) = \exp\left(\frac{\sigma^2 t^2}{2}\right) \qquad \varphi(t) = \exp\left(-\frac{\sigma^2 t^2}{2}\right)$$

Characterising a $\mathcal{N}(0,\sigma^2)$

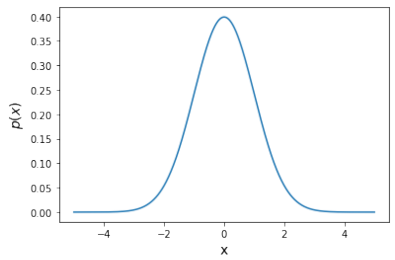
- At some point in your BSc/MSc/PhD, you have probably come across the many characterisations of a Gaussian:

$$p(x) = \frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{x^2}{2\sigma^2}\right) \qquad F(x) = \frac{1}{2}\left(1 + \text{Erf}\left(\frac{x}{\sqrt{2}\sigma}\right)\right)$$

$$M(t) = \exp\left(\frac{\sigma^2 t^2}{2}\right) \qquad \varphi(t) = \exp\left(-\frac{\sigma^2 t^2}{2}\right)$$

- We will now add a new one…

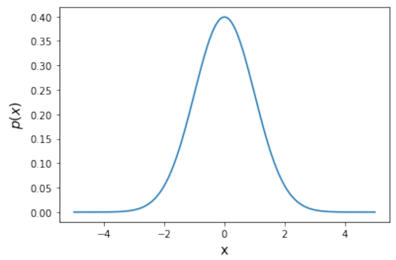
Stein characterisation for a $\mathcal{N}(0,\sigma^2)$



- Recall the general form of a Stein characterisation as a pair $(\mathcal{S}_P, \mathcal{G}_P)$:

$$Q = \mathcal{N}(0,\sigma^2) \quad \Leftrightarrow \quad \mathbb{E}_{X \sim Q}[\mathcal{S}_P[g](X)] = 0 \quad \forall g \in \mathcal{G}_P$$

Stein characterisation for a $\mathcal{N}(0,\sigma^2)$



- Recall the general form of a Stein characterisation as a pair $(\mathcal{S}_P, \mathcal{G}_P)$:

$$Q = \mathcal{N}(0,\sigma^2) \quad \Leftrightarrow \quad \mathbb{E}_{X \sim Q}[\mathcal{S}_P[g](X)] = 0 \quad \forall g \in \mathcal{G}_P$$

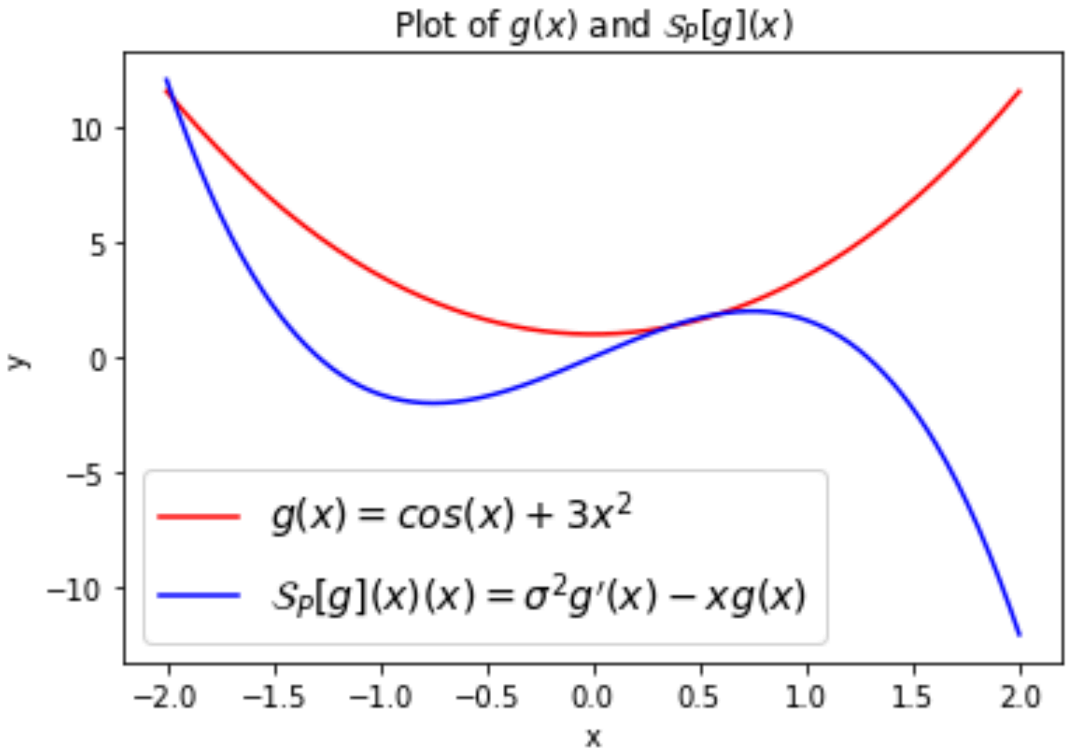
- For $P = \mathcal{N}(0,\sigma^2)$, one such pair is

$$\mathcal{G}_P := \left\{ g : \mathbb{R}^d \to \mathbb{R} \,|\, g \text{ almost diff. \& } \int |g'(x)| p(x) dx < \infty \right\}$$

$$\mathcal{S}_P[g](x) := \sigma^2 g'(x) - x g(x)$$

Illustration for a first function

$$P = \mathcal{N}(0,1)$$

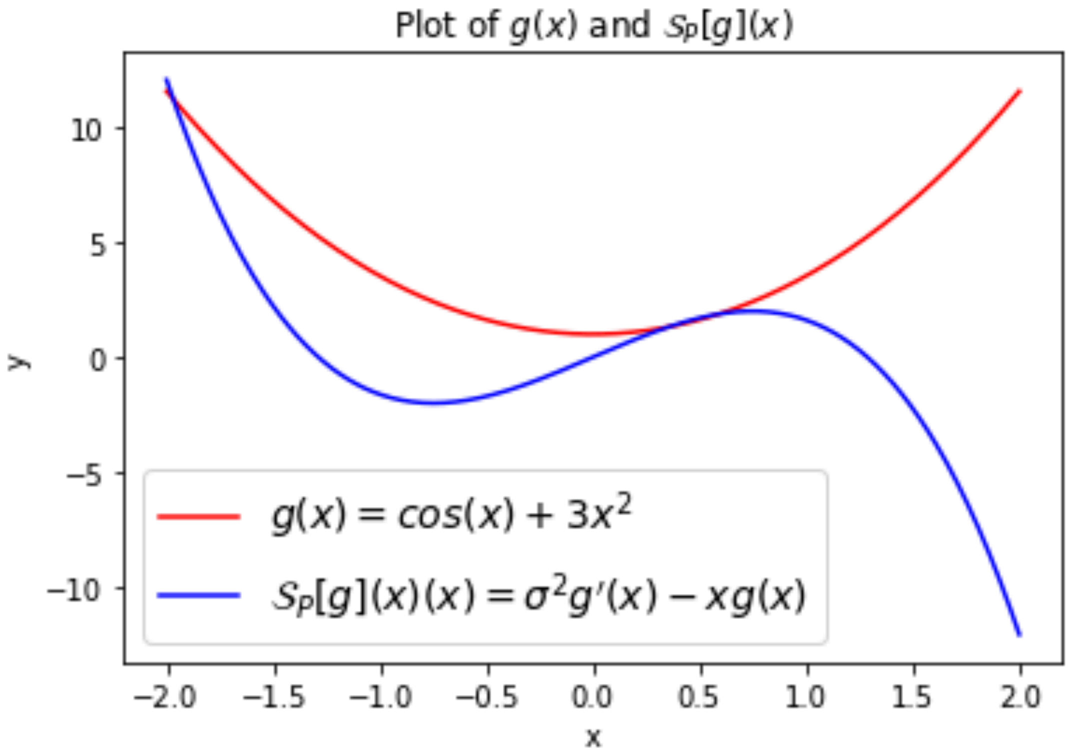


$$\sigma^2 = 1, \quad n = 10^6,$$

$$\{x_i\}_{i=1}^n \sim P = \mathcal{N}(0,\sigma^2)$$

Illustration for a first function

$$P = \mathcal{N}(0,1)$$



$$\sigma^2 = 1, \quad n = 10^6,$$

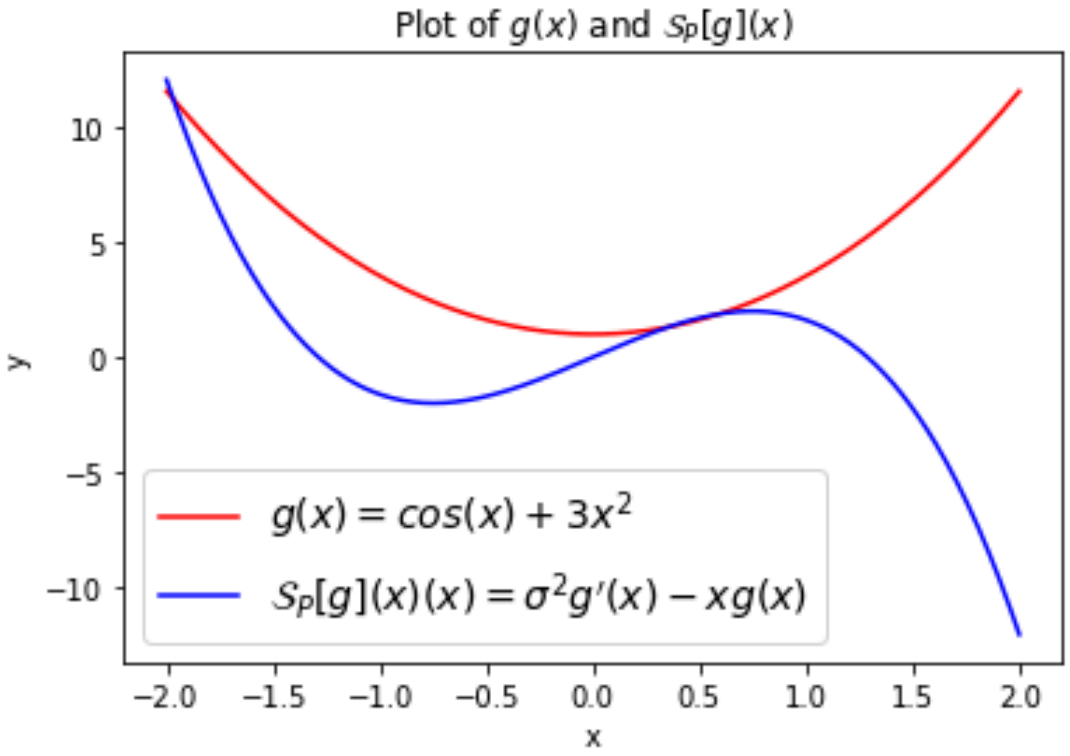
$$\{x_i\}_{i=1}^n \sim P = \mathcal{N}(0,\sigma^2)$$

$$\frac{1}{n}\sum_{i=1}^n g(x_i) \approx 3.616$$

Arbitrary function in $\mathcal{G}_P$.
Unknown mean…

Illustration for a first function

$$P = \mathcal{N}(0,1)$$



$$\sigma^2 = 1, \quad n = 10^6,$$

$$\{x_i\}_{i=1}^n \sim P = \mathcal{N}(0,\sigma^2)$$

$$\frac{1}{n}\sum_{i=1}^n g(x_i) \approx 3.616$$

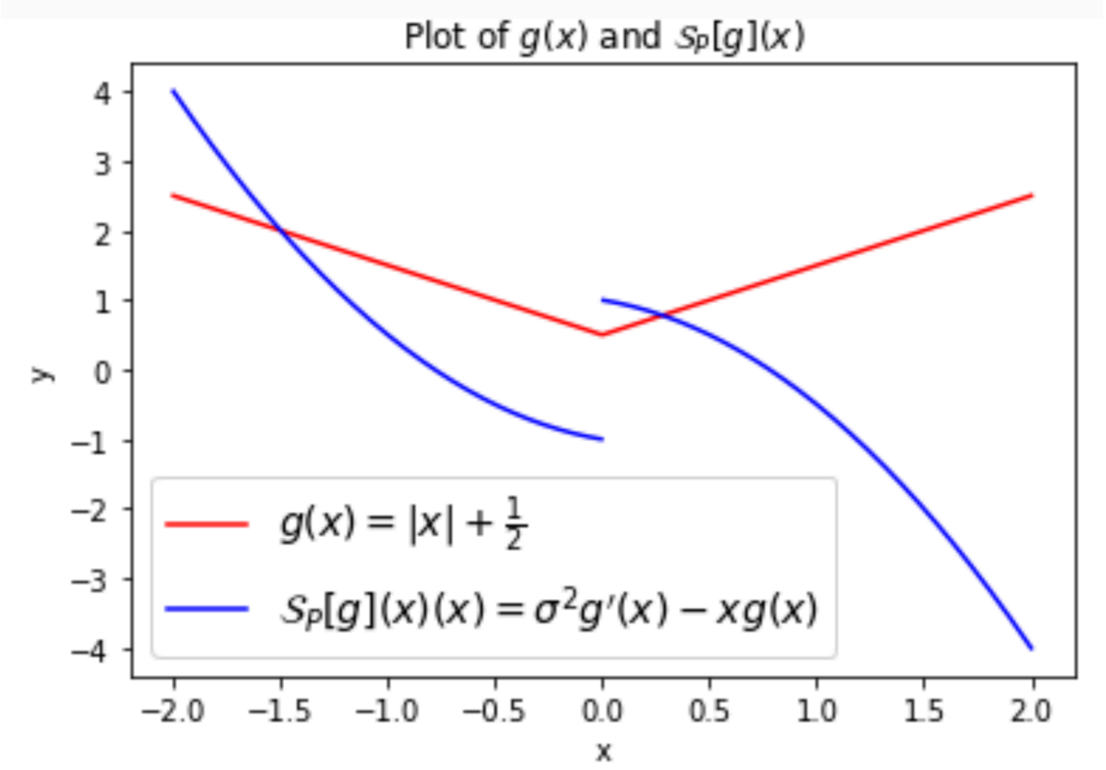
Arbitrary function in $\mathcal{G}_P$.
Unknown mean…

$$\frac{1}{n}\sum_{i=1}^n \mathcal{S}_P[g](x_i) \approx 0.007$$

Mean zero function!

Illustration for a second function

$$P = \mathcal{N}(0,1)$$

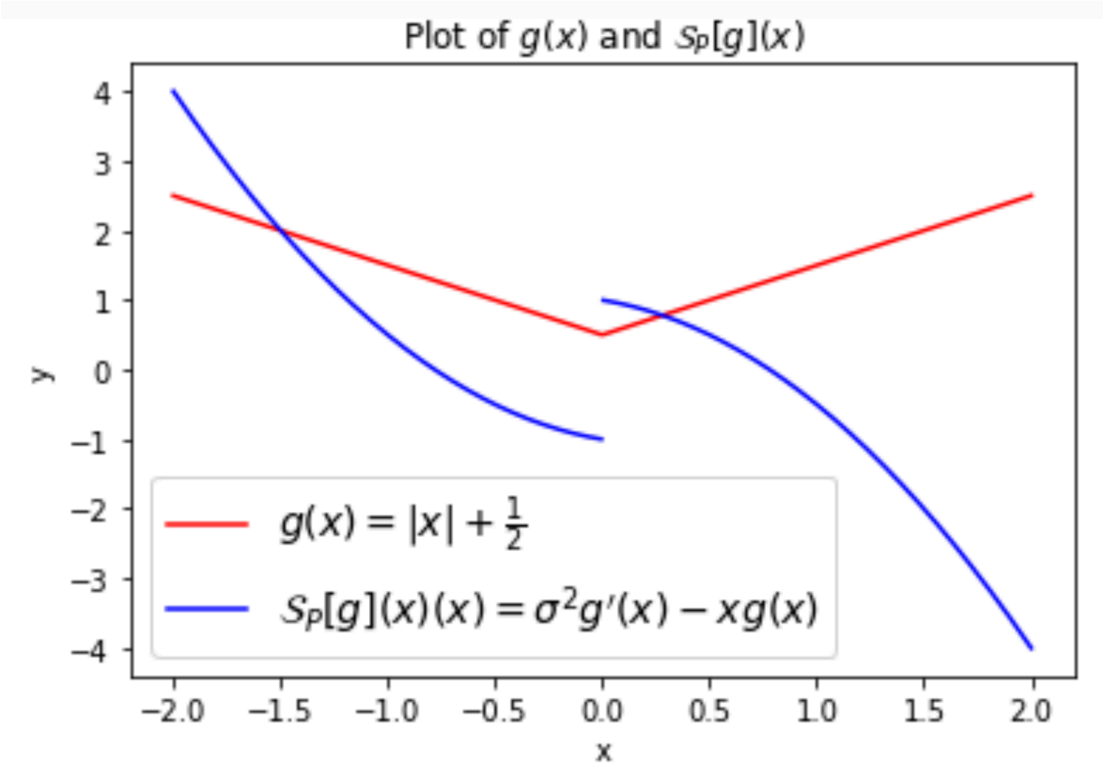


$$\sigma^2 = 1, \quad n = 10^6,$$

$$\{x_i\}_{i=1}^n \sim P = \mathcal{N}(0,\sigma^2)$$

Illustration for a second function

$$P = \mathcal{N}(0,1)$$



$$\sigma^2 = 1, \quad n = 10^6,$$

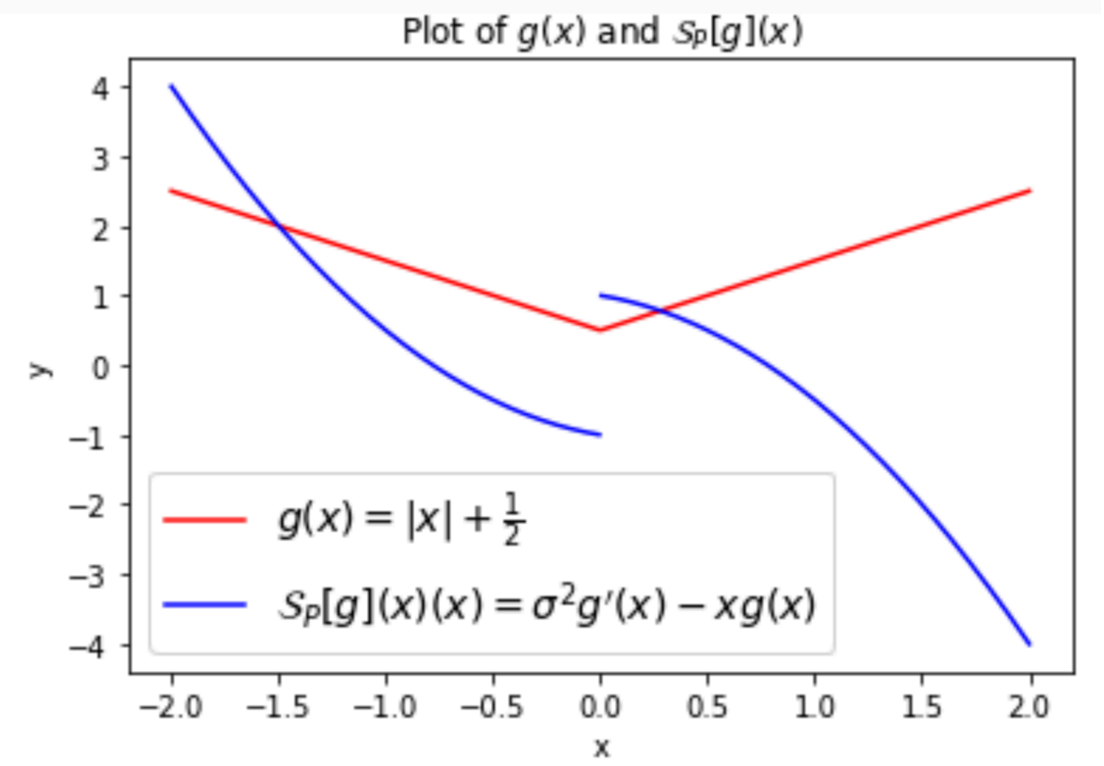
$$\{x_i\}_{i=1}^n \sim P = \mathcal{N}(0,\sigma^2)$$

$$\frac{1}{n}\sum_{i=1}^n g(x_i) \approx 1.298$$

Arbitrary function in $\mathcal{G}_P$.
Unknown mean…

Illustration for a second function

$$P = \mathcal{N}(0,1)$$



$$\sigma^2 = 1, \quad n = 10^6,$$

$$\{x_i\}_{i=1}^n \sim P = \mathcal{N}(0,\sigma^2)$$

$$\frac{1}{n}\sum_{i=1}^n g(x_i) \approx 1.298$$

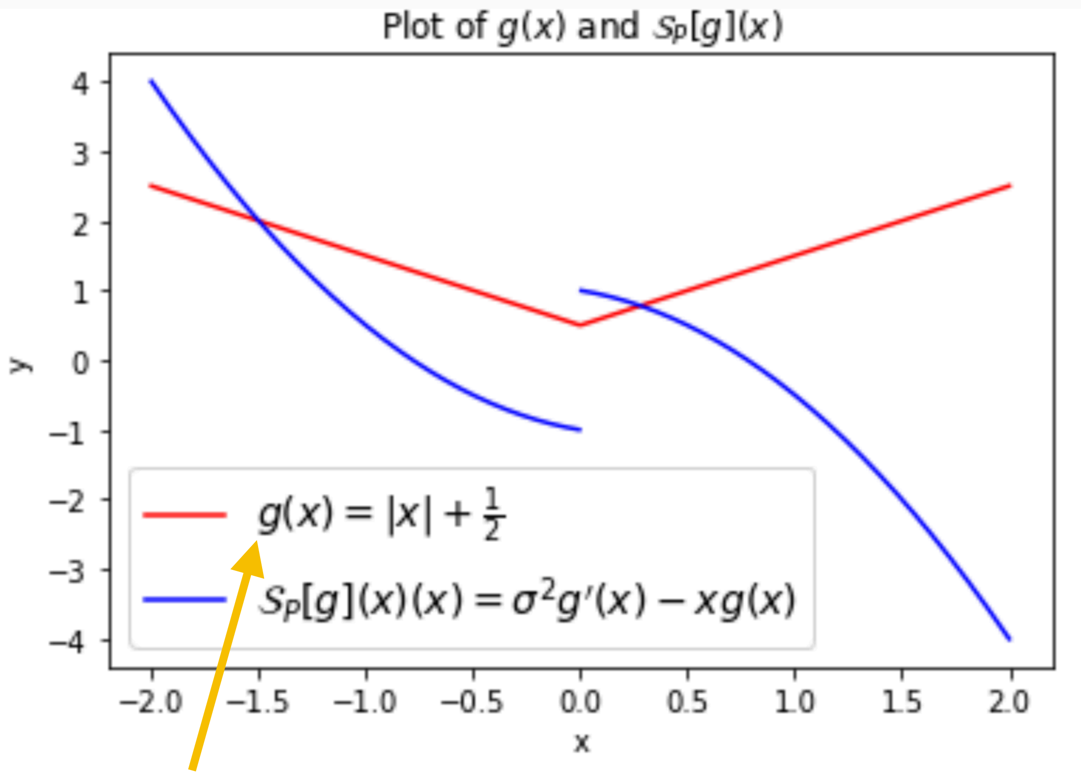
Arbitrary function in $\mathcal{G}_P$.
Unknown mean…

$$\frac{1}{n}\sum_{i=1}^n \mathcal{S}_P[g](x_i) \approx 0.002$$

Mean zero function!

Illustration for a second function

$$P = \mathcal{N}(0,1)$$



Not differentiable but almost differentiable…
We can make many many functions have mean zero!

$$\sigma^2 = 1, \quad n = 10^6,$$

$$\{x_i\}_{i=1}^n \sim P = \mathcal{N}(0,\sigma^2)$$

$$\frac{1}{n}\sum_{i=1}^n g(x_i) \approx 1.298$$

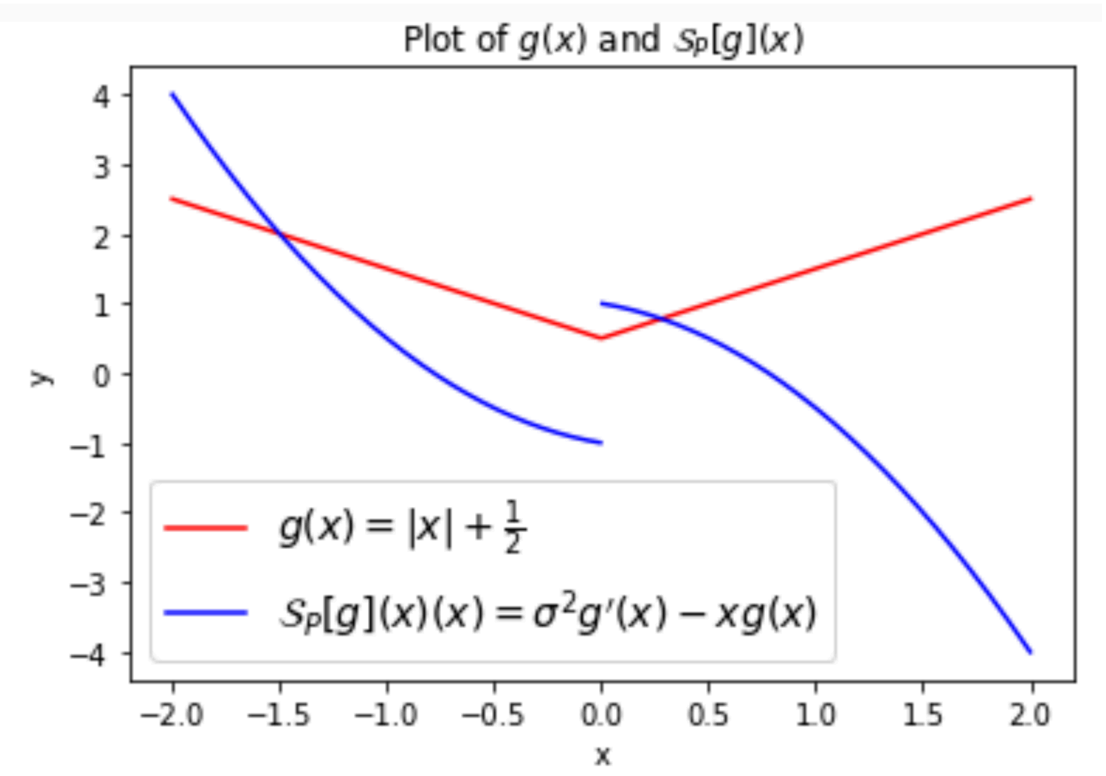
Arbitrary function in $\mathcal{G}_P$.
Unknown mean…

$$\frac{1}{n}\sum_{i=1}^n \mathcal{S}_P[g](x_i) \approx 0.002$$

Mean zero function!

Mean zero only against P though!

$$P = \mathcal{N}(0,1)$$

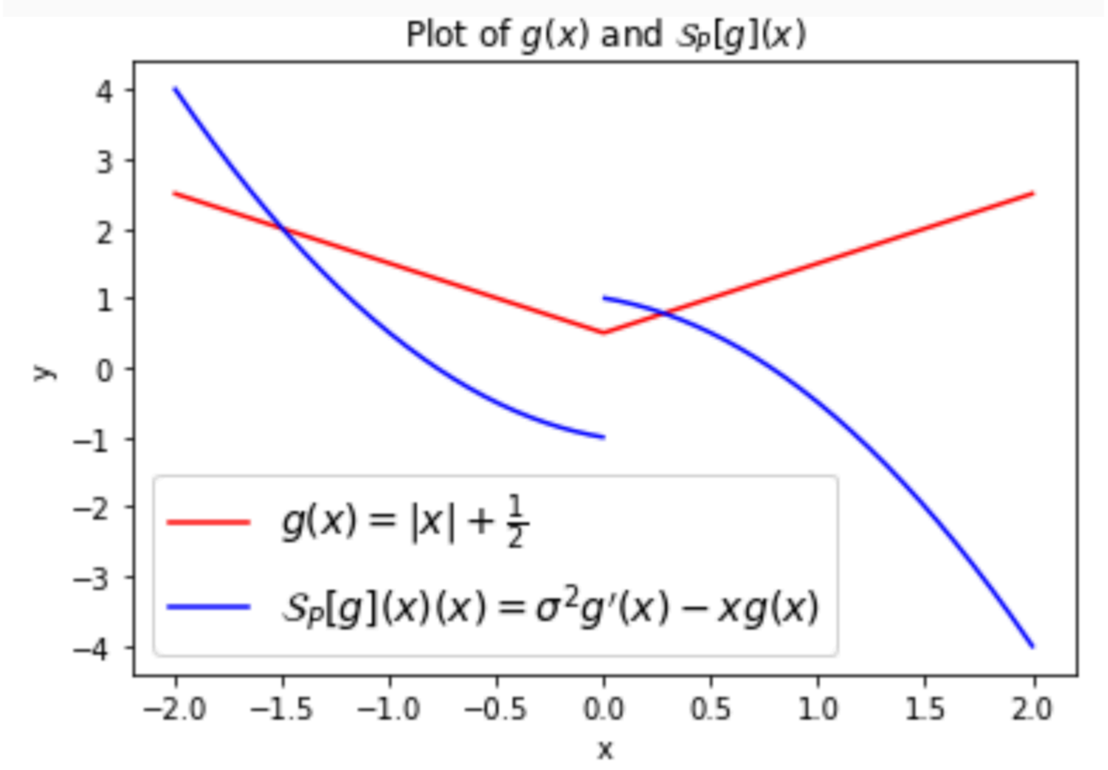


$$\{x_i\}_{i=1}^n \sim Q = \mathcal{N}(1,1) \neq P$$

$$\frac{1}{n}\sum_{i=1}^n \mathcal{S}_P[g](x_i) \boxed{\approx -1.661}$$

Mean zero only against P though!

$$P = \mathcal{N}(0,1)$$



$$\{x_i\}_{i=1}^n \sim Q = \mathcal{N}(1,1) \neq P$$

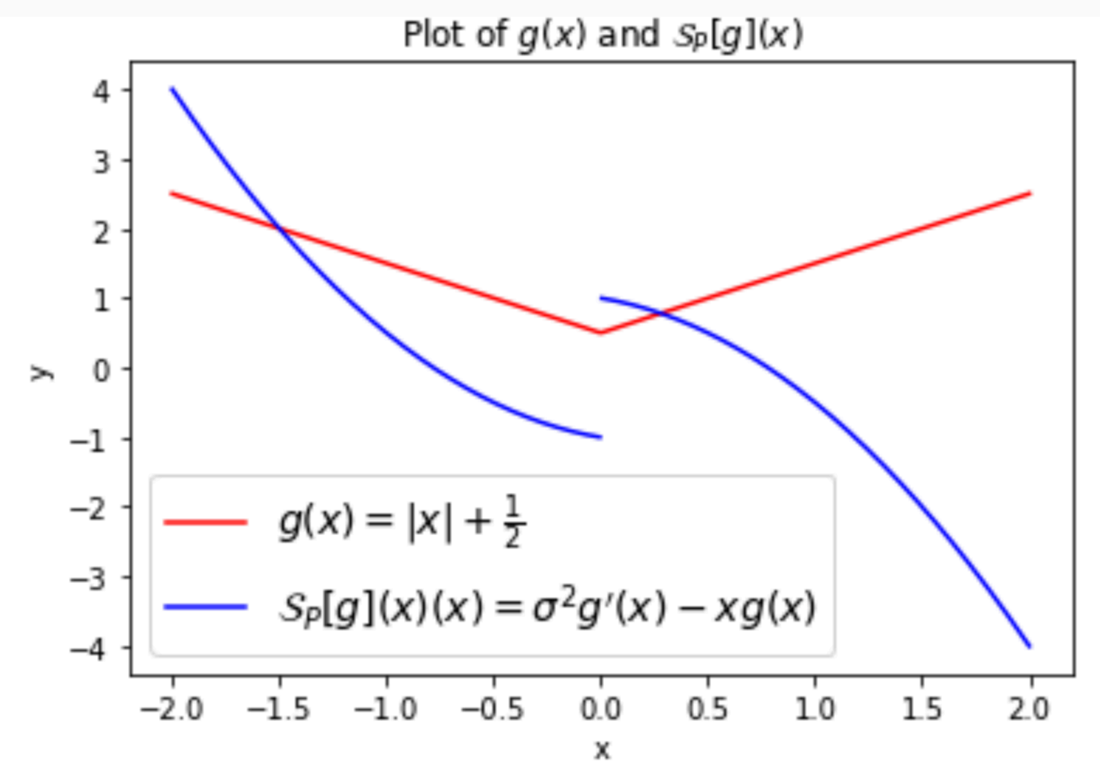
$$\frac{1}{n}\sum_{i=1}^n \mathcal{S}_P[g](x_i) \boxed{\approx -1.661}$$

$$\{x_i\}_{i=1}^n \sim Q = \mathcal{N}(0,9) \neq P$$

$$\frac{1}{n}\sum_{i=1}^n \mathcal{S}_P[g](x_i) \boxed{\approx -2.170}$$

Mean zero only against P though!

$$P = \mathcal{N}(0,1)$$



$$\{x_i\}_{i=1}^n \sim Q = \mathcal{N}(1,1) \neq P$$

$$\frac{1}{n}\sum_{i=1}^n \mathcal{S}_P[g](x_i) \boxed{\approx -1.661}$$

$$\{x_i\}_{i=1}^n \sim Q = \mathcal{N}(0,9) \neq P$$

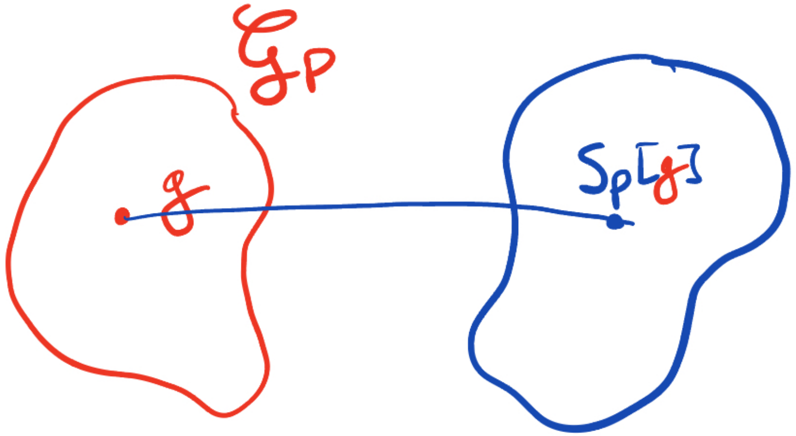
$$\frac{1}{n}\sum_{i=1}^n \mathcal{S}_P[g](x_i) \boxed{\approx -2.170}$$

We have a characterisation!

Stein characterisation

- One last time to make sure you remember it, a **Stein characterisation** for P is a pair $(\mathcal{S}_P, \mathcal{G}_P)$ such that

$$Q = P \quad \Leftrightarrow \quad \mathbb{E}_{X \sim Q}[\mathcal{S}_P[g](X)] = 0 \quad \forall g \in \mathcal{G}_P$$

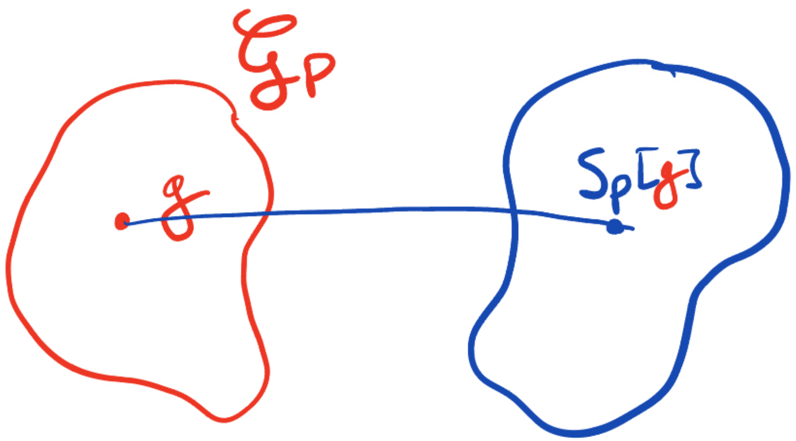


Stein characterisation

- One last time to make sure you remember it, a **Stein characterisation** for P is a pair $(\mathcal{S}_P, \mathcal{G}_P)$ such that

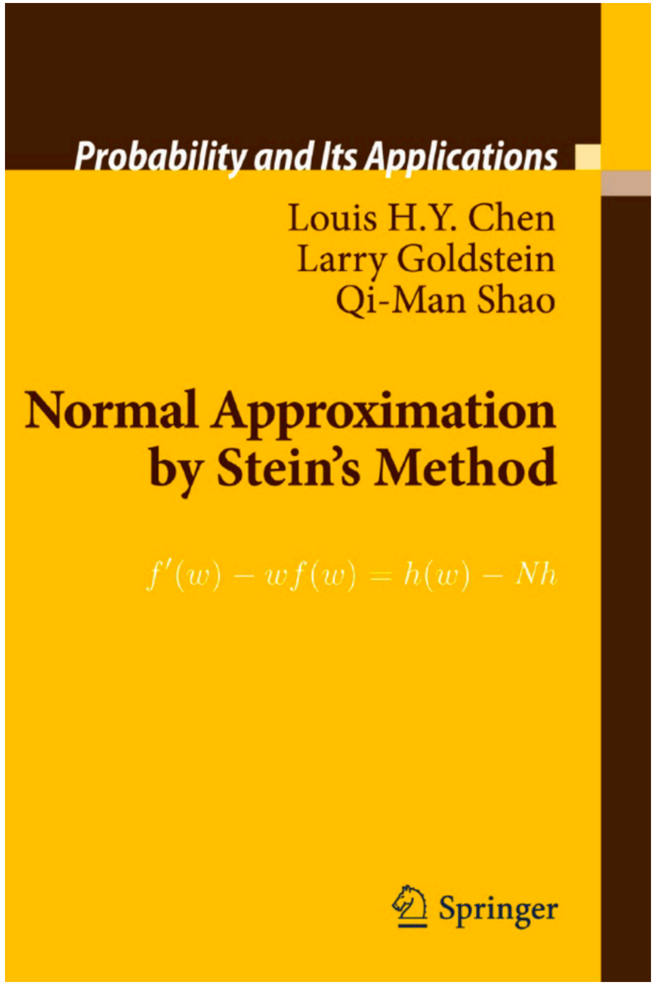
$$Q = P \quad \Leftrightarrow \quad \mathbb{E}_{X \sim Q}[\mathcal{S}_P[g](X)] = 0 \quad \forall g \in \mathcal{G}_P$$

Any questions??



Stein characterisations are not unique!

- Some might be more computationally convenient than others.
- Some might be easier to manipulate…
- This book has 350+ pages on characterising Gaussian distributions in different ways, and how this can help for theory…



Stein characterisations for other distributions

Surveys

- Arratia, R., Goldstein, L., & Gordon, L. (1990). Poisson approximation and the Chen-Stein method. *Statistical Science*, 403-424.
- Arratia, R., Goldstein, L., & Kochman, F. (2013). Size bias for one and all. *arXiv preprint arXiv:1308.2729*.
- Barbour, A. D., & Chen, L. H. (2014). Steins (magic) method. *arXiv preprint arXiv:1411.1179*.
- Chatterjee, S. (2014). A short survey of Stein's method. *arXiv preprint arXiv:1404.1392*.
- Chen, L. H. (1998). Stein's method: some perspectives with applications. In *Probability towards 2000* (pp. 97-122). Springer, New York, NY.
- Chen, L. H., & Shao, Q. M. (2005). Stein's method for normal approximation. *An introduction to Stein's method*, *4*, 1-59.
- Lachieze-Rey, R. (2018). Using Stein's method in random geometry.
- Ley, C., Reinert, G., & Swan, Y. (2017). Stein's method for comparison of univariate distributions. *Probability Surveys*, *14*, 1-52.
- Nourdin, I., & Peccati, G. (2010). Stein's method meets Malliavin calculus: a short survey with new estimates. In *Recent development in stochastic dynamics and stochastic analysis* (pp. 207-236).Reinert, G. (2011). A short introduction to Stein's method. *Lecture Notes*.
- Reinert, G. (2005). Three general approaches to Stein's method. *An introduction to Stein's method*, *4*, 183-221.
- Reinert, G. (2011). Gaussian approximation of functionals: Malliavin calculus and Stein's method. *Surveys in Stochastic Processes*, *4*, 107.
- Rinott, Y., & Rotar, V. (2000). Normal approximations by Stein's method. *Decisions in Economics and Finance*, *23*(1), 15-29.
- Ross, N. (2011). Fundamentals of Stein's method. *Probability Surveys*, *8*, 210-293.
- Upadhye, N. S., Čekanavičius, V., & Vellaisamy, P. (2017). On Stein operators for discrete approximations. *Bernoulli*, *23*(4A), 2828-2859.
- Xia, A. (2005). Stein's method and Poisson process approximation. *An introduction to Stein's method*, *4*, 115-181.

[https://sites.google.com/site/steinsmethod]

Prof. Yvik Swan (ULB)

Stein’s method as a computational tool

Why Stein? Part I: Intractable integrals

Why Stein characterisations?

- At this point, we have a new characterisation (i.e. mathematical language!) to represent distributions.

Why Stein characterisations?

- At this point, we have a new characterisation (i.e. mathematical language!) to represent distributions.
- BUT it is seemingly **much more complicated**!!
- Instead of a single function, we now have (infinitely) many…..

Why Stein characterisations?

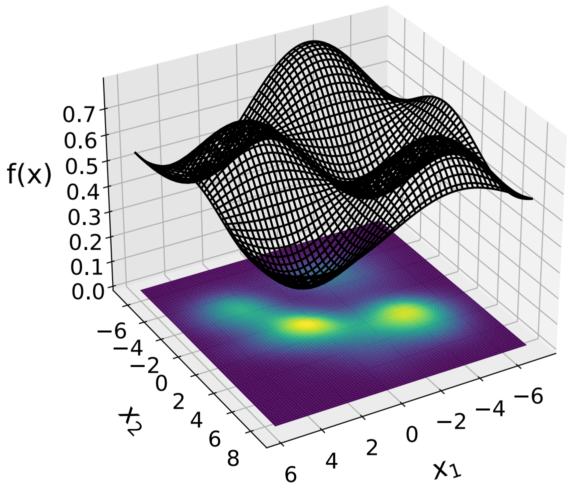
- At this point, we have a new characterisation (i.e. mathematical language!) to represent distributions.
- BUT it is seemingly **much more complicated**!!
- Instead of a single function, we now have (infinitely) many…..

The key point is that **all of these functions have mean zero under a distribution of interest**, which is super useful from a computational viewpoint!

A key challenge in computational statistics

- Let $\mathcal{X} \subseteq \mathbb{R}^d$. One of the main computational challenges encountered in statistics and machine learning is to have to compute:

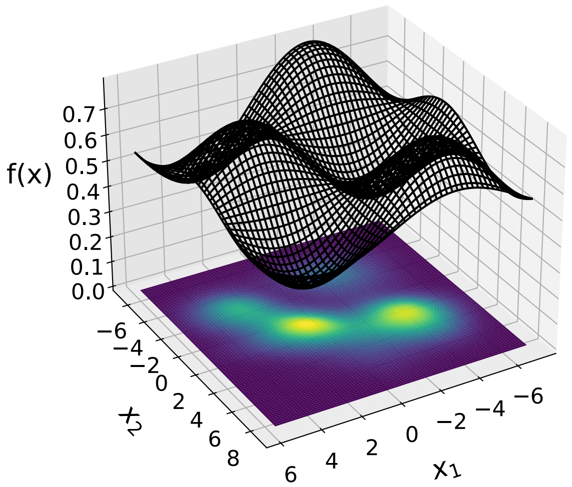
$$\mathbb{E}_{X \sim P}[f(X)] = \int_{\mathcal{X}} f(x)p(x)dx = ??$$



A key challenge in computational statistics

- Let $\mathcal{X} \subseteq \mathbb{R}^d$. One of the main computational challenges encountered in statistics and machine learning is to have to compute:

$$\mathbb{E}_{X \sim P}[f(X)] = \int_{\mathcal{X}} f(x)p(x)dx = ??$$



- This is a really **hard problem** when:
  - The problem is high-dimensional (i.e. d is large).
  - The function $f : \mathcal{X} \to \mathbb{R}$ is complicated and/or expensive.
  - The distribution P is complex/multi-modal and/or $p(x)$ cannot be evaluated point wise.

Examples in Bayesian statistics

$$\mathbb{E}_{X \sim P}[f(X)] = \int_{\mathcal{X}} f(x) p(x) dx = ??$$

Examples in Bayesian statistics

$$\mathbb{E}_{X \sim P}[f(X)] = \int_{\mathcal{X}} f(x)p(x)dx = ??$$

1. **Posterior moments:** x is some unknown parameter of our model. $f(x) = x^l$ for some $l \in \mathbb{N}$, $p(x)$ is a posterior density.

Examples in Bayesian statistics

$$\mathbb{E}_{X\sim P}[f(X)] = \int_{\mathcal{X}} f(x)p(x)dx = ??$$

1. **Posterior moments:** x is some unknown parameter of our model. $f(x) = x^l$ for some $l \in \mathbb{N}$, $p(x)$ is a posterior density.

2. **Model evidence:** x is some unknown parameter of our model. $f(x)$ is the likelihood, $p(x)$ is a prior density.

Examples in frequentist statistics

$$\mathbb{E}_{X\sim P}[f(X)] = \int_{\mathcal{X}} f(x)p(x)dx = ??$$

Examples in frequentist statistics

$$\mathbb{E}_{X\sim P}[f(X)] = \int_{\mathcal{X}} f(x)p(x)dx = ??$$

1. **Marginalisation:** Our likelihood could be based on some unobserved variables (nuisance parameters) which need to be integrated out.

Examples in frequentist statistics

$$\mathbb{E}_{X\sim P}[f(X)] = \int_{\mathcal{X}} f(x)p(x)dx = ??$$

1. **Marginalisation:** Our likelihood could be based on some unobserved variables (nuisance parameters) which need to be integrated out.

2. **Unnormalised likelihoods:** Sometimes we only have access to a likelihood up to a normalisation constant, which is the integral of the unnormalised part (e.g. graphical models, models on manifolds, deep exponential family models).

Why Stein characterisations: Intractable integrals

- Clearly if we have that f can be written as

$$f(x) = \mathcal{S}_P[g](x) + C \quad \text{for some } \mathcal{S}_P, g \in \mathcal{G}_P, C \in \mathbb{R}$$

Why Stein characterisations: Intractable integrals

- Clearly if we have that f can be written as

$$f(x) = \mathcal{S}_P[g](x) + C \quad \text{for some } \mathcal{S}_P, g \in \mathcal{G}_P, C \in \mathbb{R}$$

- Then we can compute this integral/expectation in closed form:

$$\mathbb{E}_{X \sim P}[f(X)] = \mathbb{E}_{X \sim P}[\mathcal{S}_P[g](x)] + \mathbb{E}_{X \sim P}[C] = C$$

Why Stein characterisations: Intractable integrals

- Clearly if we have that f can be written as

$$f(x) = \mathcal{S}_P[g](x) + C \quad \text{for some } \mathcal{S}_P, g \in \mathcal{G}_P, C \in \mathbb{R}$$

- Then we can compute this integral/expectation in closed form:

$$\mathbb{E}_{X\sim P}[f(X)] = \mathbb{E}_{X\sim P}[\mathcal{S}_P[g](x)] + \mathbb{E}_{X\sim P}[C] = C$$

Known!

Why Stein characterisations: Intractable integrals

- Clearly if we have that f can be written as

$$f(x) = \mathcal{S}_P[g](x) + C \quad \text{for some } \mathcal{S}_P, g \in \mathcal{G}_P, C \in \mathbb{R}$$

- Then we can compute this integral/expectation in closed form:

$$\mathbb{E}_{X \sim P}[f(X)] = \mathbb{E}_{X \sim P}[\mathcal{S}_P[g](x)] + \mathbb{E}_{X \sim P}[C] = C \longleftarrow \text{Known!}$$

- The flexibility in $\mathcal{S}_P, g, C$ makes this not too unlikely!
- A key trick is therefore to replace our intractable integrals with integrals that we can compute exactly.

Stein’s method as a computational tool

Why Stein? Part II: Intractable densities

Why Stein characterisations: Complex probability distributions

- Our first motivation for Stein characterisations was for calculating intractable integrals.
- But equally important is the case where our distribution is not very tractable in the sense that we only know its unnormalised density:

$$p(x) = \frac{\tilde{p}(x)}{C}$$

Why Stein characterisations: Complex probability distributions

- Our first motivation for Stein characterisations was for calculating intractable integrals.
- But equally important is the case where our distribution is not very tractable in the sense that we only know its unnormalised density:

Can evaluate

$$p(x) = \frac{\tilde{p}(x)}{C}$$

Why Stein characterisations: Complex probability distributions

- Our first motivation for Stein characterisations was for calculating intractable integrals.
- But equally important is the case where our distribution is not very tractable in the sense that we only know its unnormalised density:

Can evaluate

$$p(x) = \frac{\tilde{p}(x)}{C}$$

Cannot evaluate…

Why Stein characterisations: Complex probability distributions

- Our first motivation for Stein characterisations was for calculating intractable integrals.
- But equally important is the case where our distribution is not very tractable in the sense that we only know its unnormalised density:

Can evaluate

$$p(x) = \frac{\tilde{p}(x)}{C}$$

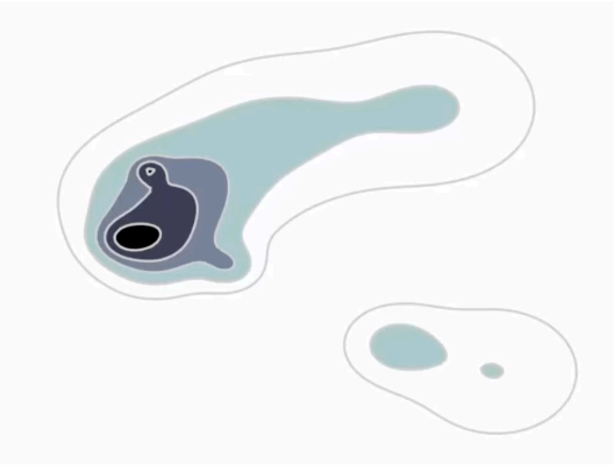
Cannot evaluate…

Cannot evaluate…

Characterisation for a complicated posterior distribution?

- Suppose we have a prior $p(\theta)$, and n iid observations from a distribution with density $p(x \mid \theta)$. Then the posterior is:

$$p(\theta \mid x_1, \ldots, x_n) = \frac{1}{C} \prod_{i=1}^{n} p(x_i \mid \theta) p(\theta)$$

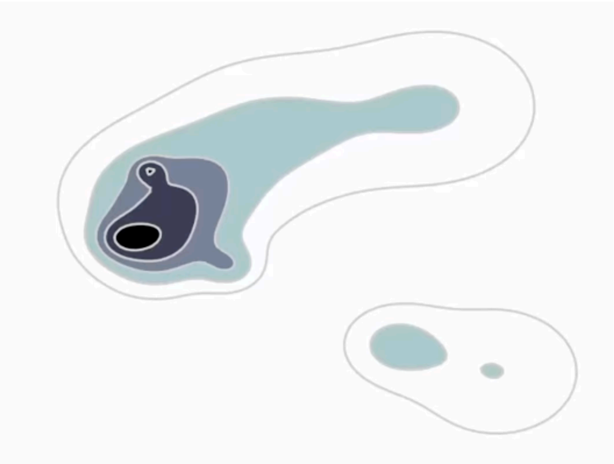


Characterisation for a complicated posterior distribution?

- Suppose we have a prior $p(\theta)$, and n iid observations from a distribution with density $p(x \mid \theta)$. Then the posterior is:

$$p(\theta \mid x_1, \ldots, x_n) = \frac{1}{C} \prod_{i=1}^{n} p(x_i \mid \theta) p(\theta)$$

$$C = \int_{\Theta} \prod_{i=1}^{n} p(x_i \mid \theta) p(\theta) d\theta$$

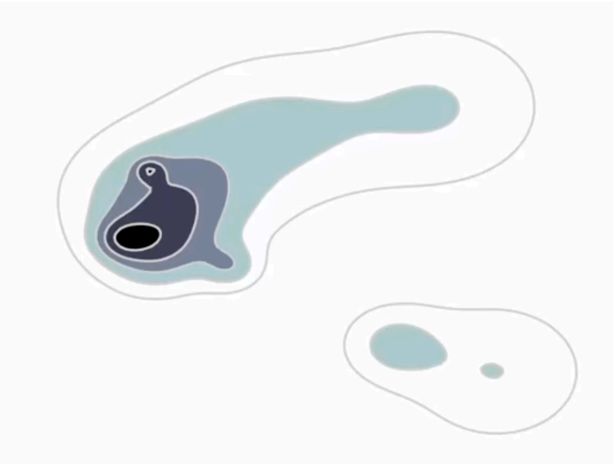


Characterisation for a complicated posterior distribution?

- Suppose we have a prior $p(\theta)$, and n iid observations from a distribution with density $p(x\,|\,\theta)$. Then the posterior is:

$$p(\theta\,|\,x_1,\ldots,x_n) = \frac{1}{C}\prod_{i=1}^{n} p(x_i\,|\,\theta)p(\theta)$$

$$C = \int_\Theta \prod_{i=1}^{n} p(x_i\,|\,\theta)p(\theta)d\theta$$



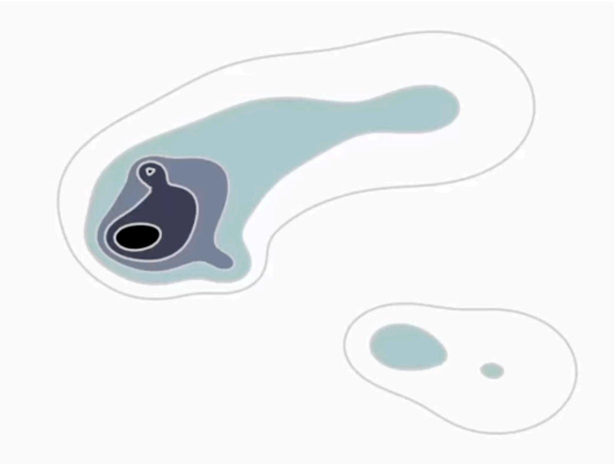
$F(x)$?

Characterisation for a complicated posterior distribution?

- Suppose we have a prior $p(\theta)$, and n iid observations from a distribution with density $p(x \mid \theta)$. Then the posterior is:

$$p(\theta \mid x_1, \ldots, x_n) = \frac{1}{C} \prod_{i=1}^{n} p(x_i \mid \theta) p(\theta)$$

$$C = \int_{\Theta} \prod_{i=1}^{n} p(x_i \mid \theta) p(\theta) d\theta$$



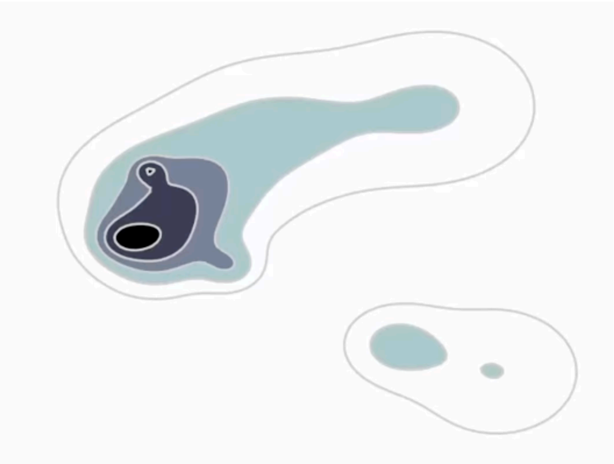
$F(x)$? $\qquad M(t)$?

Characterisation for a complicated posterior distribution?

- Suppose we have a prior $p(\theta)$, and n iid observations from a distribution with density $p(x \mid \theta)$. Then the posterior is:

$$p(\theta \mid x_1, \ldots, x_n) = \frac{1}{C} \prod_{i=1}^{n} p(x_i \mid \theta) p(\theta)$$

$$C = \int_\Theta \prod_{i=1}^{n} p(x_i \mid \theta) p(\theta) d\theta$$



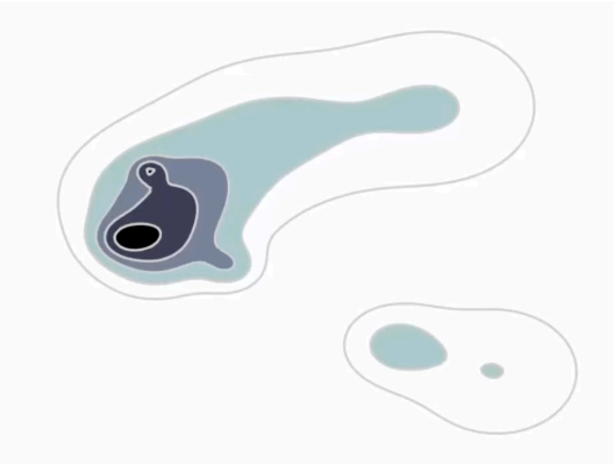
$F(x)$? $M(t)$? $\varphi(t)$?

Characterisation for a complicated posterior distribution?

- Suppose we have a prior $p(\theta)$, and n iid observations from a distribution with density $p(x \mid \theta)$. Then the posterior is:

$$p(\theta \mid x_1, \ldots, x_n) = \frac{1}{C} \prod_{i=1}^{n} p(x_i \mid \theta) p(\theta)$$

$$C = \int_{\Theta} \prod_{i=1}^{n} p(x_i \mid \theta) p(\theta) d\theta$$



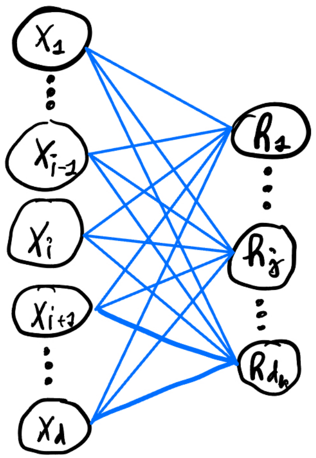
$F(x)$? $M(t)$? $\varphi(t)$?

All characterisations are intractable!

Characterisation for machine learning models?

- **Restricted Boltzmann Machine** (i.e. 'simple and shallow' ML model):

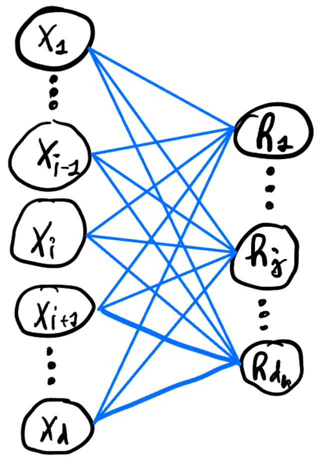
$$p(x) = \frac{1}{C} \sum_{h \in \{-1,1\}^{d_h}} \exp\left(x^\top B h + b^\top h + c^\top x - \frac{1}{2}\|x\|_2^2 \right)$$



Characterisation for machine learning models?

- **Restricted Boltzmann Machine** (i.e. 'simple and shallow' ML model):

$$p(x) = \frac{1}{C} \sum_{h \in \{-1,1\}^{d_h}} \exp\left(x^\top B h + b^\top h + c^\top x - \frac{1}{2}\|x\|_2^2 \right)$$

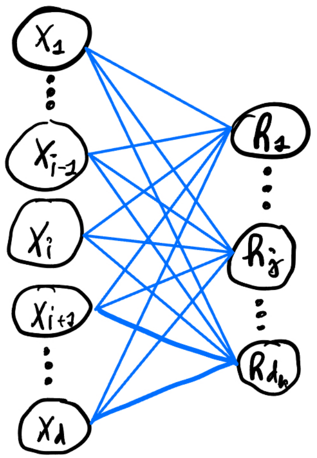


$F(x)$?

Characterisation for machine learning models?

- **Restricted Boltzmann Machine** (i.e. ‘simple and shallow’ ML model):

$$p(x) = \frac{1}{C} \sum_{h \in \{-1,1\}^{d_h}} \exp\left(x^\top B h + b^\top h + c^\top x - \frac{1}{2}\|x\|_2^2 \right)$$

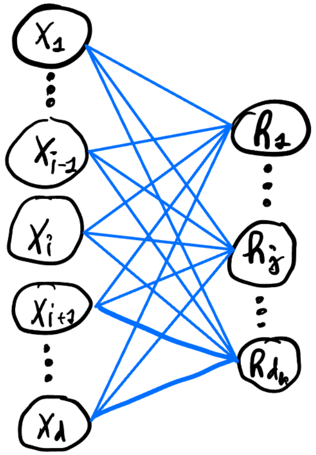


$F(x)$? $M(t)$?

Characterisation for machine learning models?

- **Restricted Boltzmann Machine** (i.e. ‘simple and shallow’ ML model):

$$p(x) = \frac{1}{C} \sum_{h \in \{-1,1\}^{d_h}} \exp\left(x^\top B h + b^\top h + c^\top x - \frac{1}{2}\|x\|_2^2 \right)$$

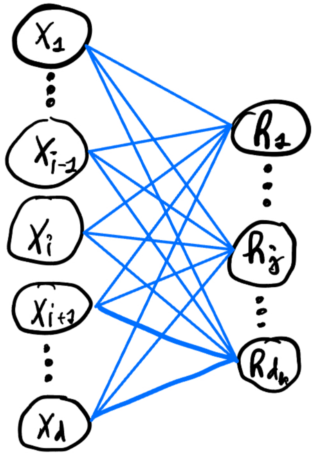


$F(x)$? $M(t)$? $\varphi(t)$?

Characterisation for machine learning models?

- **Restricted Boltzmann Machine** (i.e. 'simple and shallow' ML model):

$$p(x) = \frac{1}{C} \sum_{h \in \{-1,1\}^{d_h}} \exp\left(x^\top B h + b^\top h + c^\top x - \frac{1}{2} \|x\|_2^2 \right)$$



$F(x)$? $M(t)$? $\varphi(t)$?

All characterisations are intractable!

Implications

- Since we are **not able to characterise these distributions in a computationally tractable way**, we cannot answer most basic questions of interest to statisticians!

Implications

- Since we are **not able to characterise these distributions in a computationally tractable way**, we cannot answer most basic questions of interest to statisticians!

“Is P a good model for our data?”

Implications

- Since we are **not able to characterise these distributions in a computationally tractable way**, we cannot answer most basic questions of interest to statisticians!

"Is P a good model for our data?"

"What is the probability of observing an extreme event?"

Implications

- Since we are **not able to characterise these distributions in a computationally tractable way**, we cannot answer most basic questions of interest to statisticians!

"Is P a good model for our data?"

"What is the probability of observing an extreme event?"

"What is the expected value of the important summary statistic $f(x)$ under P?"

Implications

- Since we are **not able to characterise these distributions in a computationally tractable way**, we cannot answer most basic questions of interest to statisticians!

"Is P a good model for our data?"

"What is the probability of observing an extreme event?"

"What is the expected value of the important summary statistic $f(x)$ under P?"

Thankfully, this is another case where Stein characterisations shine!

The main reason is that $\mathcal{S}_P$ and $\mathcal{G}_P$ can be obtained without knowledge of normalisation constants (more on this shortly).

Stein’s method as a computational tool

The generator approach to Stein operators

Finding Stein characterisations

- So far, I have shown you a simple Stein characterisation for the $\mathcal{N}(0,\sigma^2)$ distribution.

Finding Stein characterisations

- So far, I have shown you a simple Stein characterisation for the $\mathcal{N}(0,\sigma^2)$ distribution.

- That's cute, but what about these more complex models…?

Finding Stein characterisations

- So far, I have shown you a simple Stein characterisation for the $\mathcal{N}(0,\sigma^2)$ distribution.
- That's cute, but what about these more complex models…?
- One of the main aim of theoretical research on Stein's method is to find a **convenient recipe to discover Stein characterisations**.

Finding Stein characterisations

- So far, I have shown you a simple Stein characterisation for the $\mathcal{N}(0,\sigma^2)$ distribution.
- That's cute, but what about these more complex models…?
- One of the main aim of theoretical research on Stein's method is to find a **convenient recipe to discover Stein characterisations**.

Your favourite distribution P $\xrightarrow{???}$ Stein characterisation

Finding Stein characterisations

- So far, I have shown you a simple Stein characterisation for the $\mathcal{N}(0,\sigma^2)$ distribution.

- That's cute, but what about these more complex models…?

- One of the main aim of theoretical research on Stein's method is to find a **convenient recipe to discover Stein characterisations**.

Your favourite distribution P $\xrightarrow{???}$ Stein characterisation

- Since this is hard, we will just follow what serious mathematicians have previously proposed…

The Langevin Stein operator

- Suppose g is a (sufficiently regular) vector-valued function $g : \mathbb{R}^d \to \mathbb{R}^d$. The **Langevin Stein operator** is given by:

$$\mathcal{T}[g](x) := \langle \nabla_x \log p(x), g(x) \rangle + \langle \nabla, g(x) \rangle$$

The Langevin Stein operator

- Suppose g is a (sufficiently regular) vector-valued function $g : \mathbb{R}^d \to \mathbb{R}^d$. The **Langevin Stein operator** is given by:

$$\mathcal{T}[g](x) := \langle \nabla_x \log p(x), g(x) \rangle + \langle \nabla, g(x) \rangle$$

$$g(x) = \begin{bmatrix} g_1(x) \\ g_2(x) \\ \vdots \\ g_d(x) \end{bmatrix}$$

The Langevin Stein operator

- Suppose g is a (sufficiently regular) vector-valued function $g : \mathbb{R}^d \to \mathbb{R}^d$. The **Langevin Stein operator** is given by:

$$\mathcal{T}[g](x) := \langle \nabla_x \log p(x), g(x) \rangle + \langle \nabla, g(x) \rangle$$

$$g(x) = \begin{bmatrix} g_1(x) \\ g_2(x) \\ \vdots \\ g_d(x) \end{bmatrix} \qquad \nabla \log p(x) = \begin{bmatrix} \frac{\partial \log p(x)}{\partial x_1} \\ \vdots \\ \frac{\partial \log p(x)}{\partial x_d} \end{bmatrix}$$

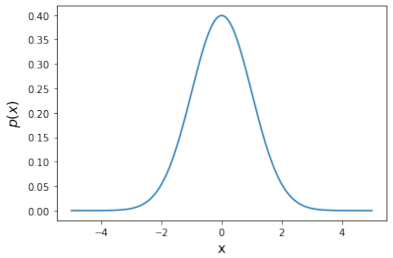
The Langevin Stein operator

- Suppose g is a (sufficiently regular) vector-valued function $g : \mathbb{R}^d \to \mathbb{R}^d$. The **Langevin Stein operator** is given by:

$$\mathcal{T}[g](x) := \langle \nabla_x \log p(x), g(x) \rangle + \langle \nabla, g(x) \rangle$$

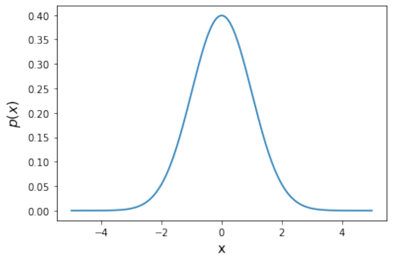
$$g(x) = \begin{bmatrix} g_1(x) \\ g_2(x) \\ \vdots \\ g_d(x) \end{bmatrix} \qquad \nabla \log p(x) = \begin{bmatrix} \frac{\partial \log p(x)}{\partial x_1} \\ \vdots \\ \frac{\partial \log p(x)}{\partial x_d} \end{bmatrix} \qquad \sum_{j=1}^{d} \frac{\partial g_j(x)}{\partial x_j}$$

Recovering our operator for $N(0,\sigma^2)$



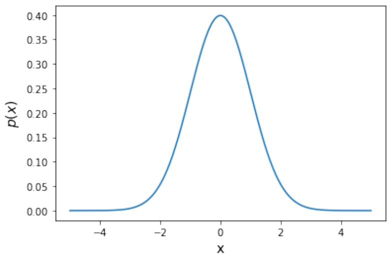
- Take $d = 1$ and $p(x) = \dfrac{1}{\sqrt{2\pi\sigma}} \exp\left(-\dfrac{x^2}{2\sigma^2}\right)$

Recovering our operator for $N(0,\sigma^2)$



- Take $d = 1$ and $p(x) = \frac{1}{\sqrt{2\pi\sigma}} \exp\left(-\frac{x^2}{2\sigma^2}\right) \rightarrow \nabla_x \log p(x) = -\frac{x}{\sigma^2}$

Recovering our operator for $N(0,\sigma^2)$

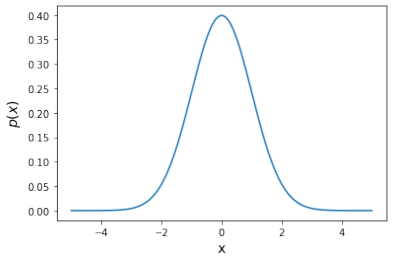


- Take $d = 1$ and $p(x) = \frac{1}{\sqrt{2\pi\sigma}} \exp\left(-\frac{x^2}{2\sigma^2}\right)$ $\longrightarrow$ $\nabla_x \log p(x) = -\frac{x}{\sigma^2}$

- Hence:

$$\mathcal{T}[g](x) := \langle \nabla_x \log p(x), g(x) \rangle + \langle \nabla, g(x) \rangle = -\frac{x}{\sigma^2} g(x) + g'(x)$$

Recovering our operator for $N(0,\sigma^2)$



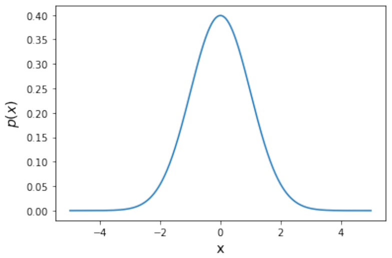
- Take $d = 1$ and $p(x) = \frac{1}{\sqrt{2\pi\sigma}} \exp\left(-\frac{x^2}{2\sigma^2}\right)$ $\rightarrow$ $\nabla_x \log p(x) = -\frac{x}{\sigma^2}$

- Hence:

$$\mathcal{T}[g](x) := \langle \nabla_x \log p(x), g(x) \rangle + \langle \nabla, g(x) \rangle = -\frac{x}{\sigma^2} g(x) + g'(x)$$

- Before, we had… $\mathcal{S}_P[g](x) := \sigma^2 g'(x) - xg(x)$.

Recovering our operator for $N(0,\sigma^2)$



- Take $d = 1$ and $p(x) = \frac{1}{\sqrt{2\pi\sigma}} \exp\left(-\frac{x^2}{2\sigma^2}\right)$ → $\nabla_x \log p(x) = -\frac{x}{\sigma^2}$

- Hence:

$$\mathcal{T}[g](x) := \langle \nabla_x \log p(x), g(x) \rangle + \langle \nabla, g(x) \rangle = -\frac{x}{\sigma^2} g(x) + g'(x)$$

- Before, we had… $\mathcal{S}_P[g](x) := \sigma^2 g'(x) - x g(x)$.

→ i.e. $\mathcal{S}_P[g](x) = \sigma^2 \mathcal{T}[g](x)$!!

Why this operator?

- Recall the problem of unnormalised densities:

$$p(x) = \frac{\tilde{p}(x)}{C}$$

Why this operator?

- Recall the problem of unnormalised densities:

$$p(x) = \frac{\tilde{p}(x)}{C}$$

- This is not a problem for score functions…

Why this operator?

- Recall the problem of unnormalised densities:

$$p(x) = \frac{\tilde{p}(x)}{C}$$

- This is not a problem for score functions…

$$\nabla_x \log p(x) = \nabla_x \log \left(\frac{\tilde{p}(x)}{C} \right)$$

Why this operator?

- Recall the problem of unnormalised densities:

$$p(x) = \frac{\tilde{p}(x)}{C}$$

- This is not a problem for score functions…

$$\nabla_x \log p(x) = \nabla_x \log \left(\frac{\tilde{p}(x)}{C} \right)$$

$$= \nabla_x \log \tilde{p}(x) - \nabla_x \log C$$

Why this operator?

- Recall the problem of unnormalised densities:

$$p(x) = \frac{\tilde{p}(x)}{C}$$

- This is not a problem for score functions…

$$\nabla_x \log p(x) = \nabla_x \log \left(\frac{\tilde{p}(x)}{C} \right)$$

$$= \nabla_x \log \tilde{p}(x) - \nabla_x \log C \quad = \nabla_x \log \tilde{p}(x)$$

Why this operator?

- Recall the problem of unnormalised densities:

$$p(x) = \frac{\tilde{p}(x)}{C}$$

- This is not a problem for score functions…

$$\boxed{\nabla_x \log p(x)} = \nabla_x \log \left(\frac{\tilde{p}(x)}{C} \right)$$

Intractable!

$$= \nabla_x \log \tilde{p}(x) - \nabla_x \log C \quad = \boxed{\nabla_x \log \tilde{p}(x)}$$

Tractable!!

Operators based on the score

- The Langevin operator is therefore ideal for unnormalised densities:

$$\mathcal{T}[g](x) := \langle \nabla_x \log p(x), g(x) \rangle + \langle \nabla, g(x) \rangle$$
$$= \langle \nabla_x \log \tilde{p}(x), g(x) \rangle + \langle \nabla, g(x) \rangle$$

Operators based on the score

- The Langevin operator is therefore ideal for unnormalised densities:

$$\mathcal{T}[g](x) := \langle \nabla_x \log p(x), g(x) \rangle + \langle \nabla, g(x) \rangle$$
$$= \langle \nabla_x \log \tilde{p}(x), g(x) \rangle + \langle \nabla, g(x) \rangle$$

- It is however not the only Stein operator based on score functions (recall that Stein characterisations are not unique!).

The generator approach

Prof. A. Barbour
(U. Zurich)

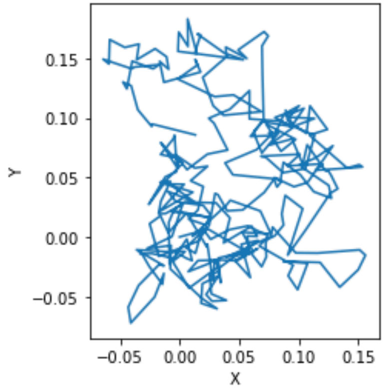
- The Langevin Stein operator is an example of Stein operator derived through the **generator approach**.

The generator approach

Prof. A. Barbour
(U. Zurich)

- The Langevin Stein operator is an example of Stein operator derived through the **generator approach**.

- **High-level idea:** Construct a Markov chain/process with invariant distribution the distribution P you would like to characterise.

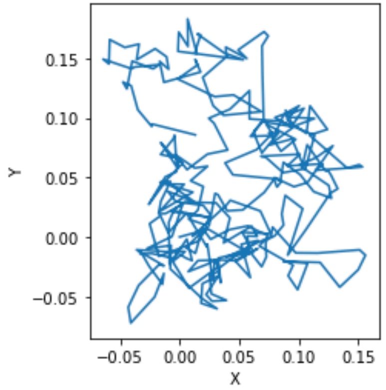


The generator approach

Prof. A. Barbour
(U. Zurich)

- The Langevin Stein operator is an example of Stein operator derived through the **generator approach**.

- **High-level idea:** Construct a Markov chain/process with invariant distribution the distribution P you would like to characterise.



- One representation of a Markov chain is through its infinitesimal generator.

→ Infinitesimal generator = Stein operator

The diffusion Stein operator

- Suppose g is a (sufficiently regular) vector-valued function $g : \mathbb{R}^d \to \mathbb{R}^d$ and m is a (nice) matrix-valued function $m : \mathbb{R}^d \to \mathbb{R}^{d\times d}$. The **diffusion Stein operator** is given by:

$$\mathcal{T}_{\text{diff}}[g](x) := \langle m(x)^\top \nabla_x \log p(x), g(x)\rangle + \langle \nabla, m(x)g(x)\rangle$$

The diffusion Stein operator

- Suppose g is a (sufficiently regular) vector-valued function $g : \mathbb{R}^d \to \mathbb{R}^d$ and m is a (nice) matrix-valued function $m : \mathbb{R}^d \to \mathbb{R}^{d \times d}$. The **diffusion Stein operator** is given by:

$$\mathcal{T}_{\text{diff}}[g](x) := \langle m(x)^\top \nabla_x \log p(x), g(x) \rangle + \langle \nabla, m(x) g(x) \rangle$$

→ Generator for pre-conditioned Langevin diffusions!

The diffusion Stein operator

- Suppose g is a (sufficiently regular) vector-valued function $g : \mathbb{R}^d \to \mathbb{R}^d$ and m is a (nice) matrix-valued function $m : \mathbb{R}^d \to \mathbb{R}^{d\times d}$. The **diffusion Stein operator** is given by:

$$\mathcal{T}_{\text{diff}}[g](x) := \langle m(x)^\top \nabla_x \log p(x), g(x)\rangle + \langle \nabla, m(x)g(x)\rangle$$

→ Generator for pre-conditioned Langevin diffusions!

- We recover the Langevin Stein operator when $m(x) = I_d$:

$$\mathcal{T}[g](x) := \langle \nabla_x \log p(x), g(x)\rangle + \langle \nabla, g(x)\rangle$$

Summary

- We have a new mathematical language (i.e. characterisation) to work with probability distributions.

Summary

- We have a new mathematical language (i.e. characterisation) to work with probability distributions.
- This new characterisation is quite a bit more complicated than what we are used to as it is represented through a pair.

$$(\mathcal{S}_P, \mathcal{G}_P)$$

Summary

- We have a new mathematical language (i.e. characterisation) to work with probability distributions.
- This new characterisation is quite a bit more complicated than what we are used to as it is represented through a pair.

$$(\mathcal{S}_P, \mathcal{G}_P)$$

However, it is easy to find such operators/characterisation for very complex distribution (including posteriors or complex ML models)!

Summary

- We have a new mathematical language (i.e. characterisation) to work with probability distributions.
- This new characterisation is quite a bit more complicated than what we are used to as it is represented through a pair.

$$(\mathcal{S}_P, \mathcal{G}_P)$$

→ However, it is easy to find such operators/characterisation for very complex distribution (including posteriors or complex ML models)!

- What should we do with our new tool?

Outline (updated)

- What is Stein's method, and why should you care…
- Computational tools based on Stein's method.
- Some nice (new) algorithms!

Stein’s method as a computational tool

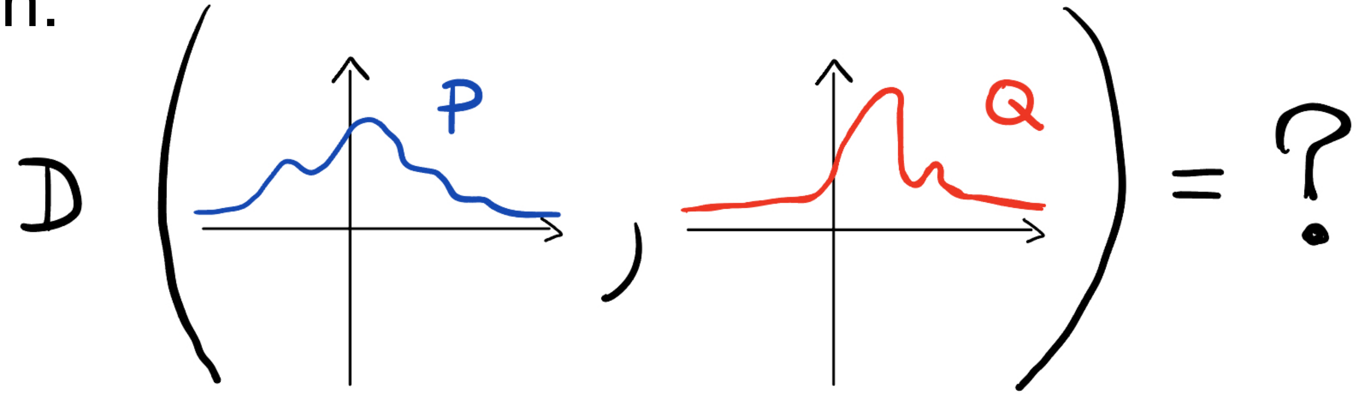
Stein discrepancies

Discrepancies

- One thing we might want to use our tool for is comparing distributions.

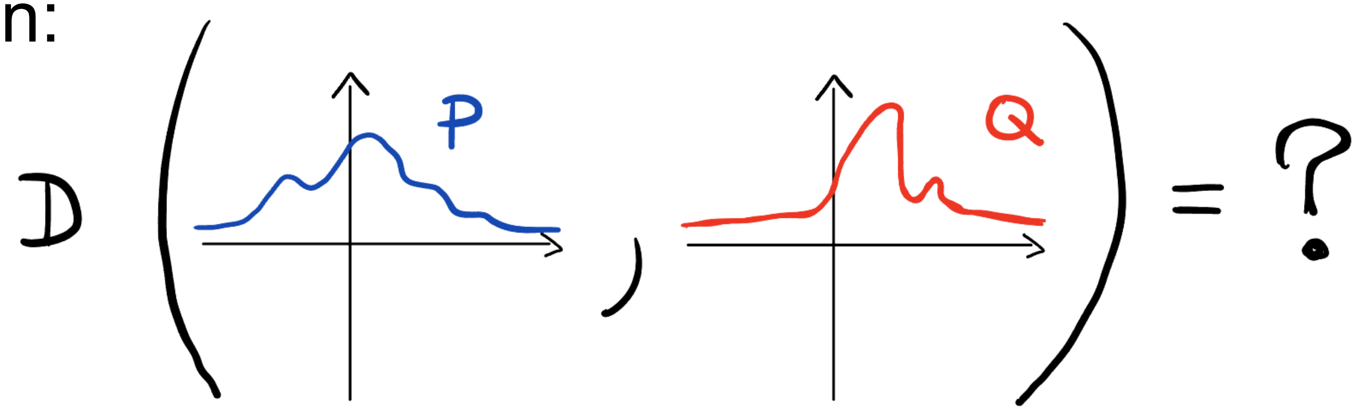
Discrepancies

- One thing we might want to use our tool for is comparing distributions.
- One idea would be to construct some notion of **dissimilarity/ discrepancy** between two distributions P, Q based on our characterisation:



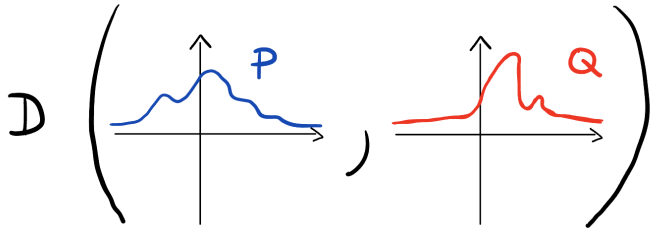
Discrepancies

- One thing we might want to use our tool for is comparing distributions.
- One idea would be to construct some notion of **dissimilarity/ discrepancy** between two distributions P, Q based on our characterisation:



- One limitation of most existing discrepancies is stats/ML is that they are **not computable** for complex P, Q.

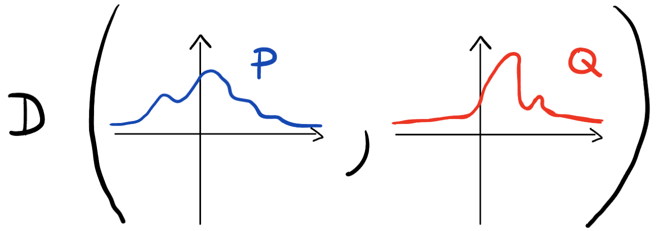
Integral probability metrics



- A very popular class of discrepancies in statistics and ML are **integral probability metrics (IPMs)**:

$$D(P, Q) = \sup_{h \in \mathcal{H}} |\, \mathbb{E}_{X \sim P}[h(X)] - \mathbb{E}_{X \sim Q}[h(X)] \,|$$

Integral probability metrics

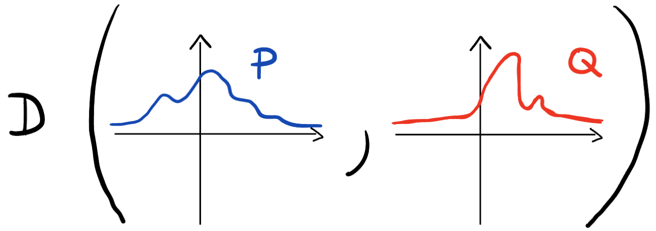


- A very popular class of discrepancies in statistics and ML are **integral probability metrics (IPMs)**:

$$D(P, Q) = \sup_{h \in \mathscr{H}} |\, \mathbb{E}_{X \sim P}[h(X)] - \mathbb{E}_{X \sim Q}[h(X)] \,|$$

- If $\mathscr{H} = \{h(x) = x\}$, then we are just comparing the means of P & Q.

Integral probability metrics

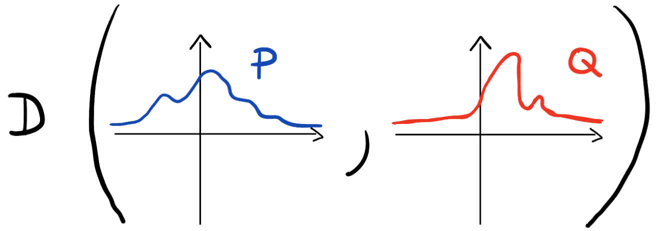


- A very popular class of discrepancies in statistics and ML are **integral probability metrics (IPMs)**:

$$D(P, Q) = \sup_{h \in \mathscr{H}} |\, \mathbb{E}_{X \sim P}[h(X)] - \mathbb{E}_{X \sim Q}[h(X)] \,|$$

  - If $\mathscr{H} = \{h(x) = x\}$, then we are just comparing the means of P & Q.
  - If $\mathscr{H}$ are all functions with Lipschitz constant less than 1, we recover the 1-**Wasserstein distance**

Integral probability metrics

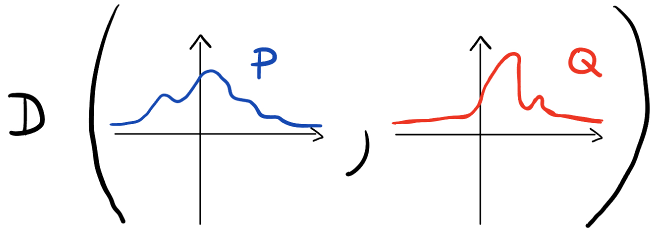


- A very popular class of discrepancies in statistics and ML are **integral probability metrics (IPMs)**:

$$D(P, Q) = \sup_{h \in \mathscr{H}} |\mathbb{E}_{X\sim P}[h(X)] - \mathbb{E}_{X\sim Q}[h(X)]|$$

  - If $\mathscr{H} = \{h(x) = x\}$, then we are just comparing the means of P & Q.
  - If $\mathscr{H}$ are all functions with Lipschitz constant less than 1, we recover the 1-**Wasserstein distance**
  - If $\mathscr{H}$ are all bounded functions with maximum at 1, we recover the **total variation distance.**

Integral probability metrics



- A very popular class of discrepancies in statistics and ML are **integral probability metrics (IPMs)**:

$$D(P, Q) = \sup_{h \in \mathscr{H}} |\, \mathbb{E}_{X\sim P}[h(X)] - \mathbb{E}_{X\sim Q}[h(X)] \,|$$

Hard to compute!

  - If $\mathscr{H} = \{h(x) = x\}$, then we are just comparing the means of P & Q.
  - If $\mathscr{H}$ are all functions with Lipschitz constant less than 1, we recover the 1-**Wasserstein distance**
  - If $\mathscr{H}$ are all bounded functions with maximum at 1, we recover the **total variation distance.**

A new class of IPMs from Stein

- Suppose we now want to consider functions of the form:

$$\mathscr{H} = \left\{ h : h(x) = \mathscr{S}_P[g](x), g \in \mathscr{G}_P \right\}$$

A new class of IPMs from Stein

- Suppose we now want to consider functions of the form:

$$\mathcal{H} = \left\{h : h(x) = \mathcal{S}_P[g](x), g \in \mathcal{G}_P\right\}$$

- Then the expression simplifies:

$$D(P, Q) = \sup_{h \in \mathcal{H}} |\mathbb{E}_{X \sim P}[h(X)] - \mathbb{E}_{X \sim Q}[h(X)]|$$

A new class of IPMs from Stein

- Suppose we now want to consider functions of the form:

$$\mathcal{H} = \left\{h : h(x) = \mathcal{S}_P[g](x), g \in \mathcal{G}_P\right\}$$

- Then the expression simplifies:

$$\begin{aligned} D(P, Q) &= \sup_{h \in \mathcal{H}} |\mathbb{E}_{X \sim P}[h(X)] - \mathbb{E}_{X \sim Q}[h(X)]| \\ &= \sup_{g \in \mathcal{G}_P} |\mathbb{E}_{X \sim P}[\mathcal{S}_P[g](X)] - \mathbb{E}_{X \sim Q}[\mathcal{S}_P[g](X)]| \end{aligned}$$

A new class of IPMs from Stein

- Suppose we now want to consider functions of the form:

$$\mathscr{H} = \left\{h : h(x) = \mathcal{S}_P[g](x), g \in \mathcal{G}_P\right\}$$

- Then the expression simplifies:

$$\begin{aligned} D(P,Q) &= \sup_{h \in \mathscr{H}} |\mathbb{E}_{X \sim P}[h(X)] - \mathbb{E}_{X \sim Q}[h(X)]| \\ &= \sup_{g \in \mathcal{G}_P} |\mathbb{E}_{X \sim P}[\mathcal{S}_P[g](X)] - \mathbb{E}_{X \sim Q}[\mathcal{S}_P[g](X)]| \\ &= \sup_{g \in \mathcal{G}_P} |\mathbb{E}_{X \sim Q}[\mathcal{S}_P[g](X)]| \end{aligned}$$

We use our key property that our functions integrate to zero under P

Stein discrepancy

- A **Stein discrepancy (SD)** is a measure of dissimilarity between P and Q:

$$\mathrm{SD}(P\,||\,Q) = \sup_{g \in \mathcal{G}} |\,\mathbb{E}_{X \sim Q}[\mathcal{S}_P[g](X)]\,|$$

Stein discrepancy

- A **Stein discrepancy (SD)** is a measure of dissimilarity between P and Q:

$$\mathrm{SD}(P||Q) = \sup_{g \in \mathcal{G}} |\mathbb{E}_{X\sim Q}[\mathcal{S}_P[g](X)]|$$

→ We do not need $\mathcal{G}$ to be the whole of $\mathcal{G}_P$, and we will often take it to only be a subset: $\mathcal{G} \subseteq \mathcal{G}_P$.

Stein discrepancy

- A **Stein discrepancy (SD)** is a measure of dissimilarity between P and Q:

$$\mathrm{SD}(P\,||\,Q) = \sup_{g \in \mathcal{G}} |\,\mathbb{E}_{X \sim Q}[\mathcal{S}_P[g](X)]\,|$$

We do not need $\mathcal{G}$ to be the whole of $\mathcal{G}_P$, and we will often take it to only be a subset: $\mathcal{G} \subseteq \mathcal{G}_P$.

If we find that at least one $g \in \mathcal{G}$ such that $\mathbb{E}_{X \sim Q}[\mathcal{S}_P[g](X)] \neq 0$, then we know $Q \neq P$!

Stein discrepancy

$$\mathrm{SD}(P||Q) = \sup_{g \in \mathcal{G}} |\mathbb{E}_{X \sim Q}[\mathcal{S}_P[g](X)]|$$

- **Question 1:** What properties does this measure of dissimilarity have?
- **Question 2:** When can we actually compute this?
- **Question 3:** What can we use this measure of dissimilarity for?

Q1: What properties does this have?

$$\mathrm{SD}(P||Q) = \sup_{g \in \mathcal{G}} |\mathbb{E}_{X \sim Q}[\mathcal{S}_P[g](X)]|$$

Q1: What properties does this have?

$$\mathrm{SD}(P\,||\,Q) = \sup_{g \in \mathcal{G}} |\,\mathbb{E}_{X \sim Q}[\mathcal{S}_P[g](X)]\,|$$

If $\mathcal{G}$ is large enough and $\mathrm{SD}(P\,||\,Q) = 0$, then we know that $Q = P$ (i.e. it is a statistical divergence)

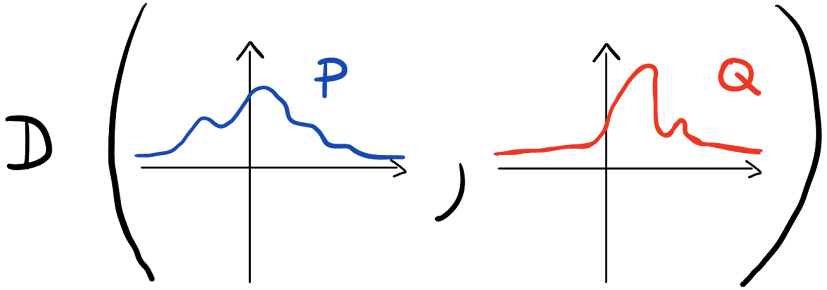
Q1: What properties does this have?

$$\mathrm{SD}(P\,||\,Q) = \sup_{g \in \mathcal{G}} |\,\mathbb{E}_{X \sim Q}[\mathcal{S}_P[g](X)]\,|$$

- If $\mathcal{G}$ is large enough and $\mathrm{SD}(P\,||\,Q) = 0$, then we know that $Q = P$ (i.e. it is a statistical divergence)
- The magnitude of $\mathrm{SD}(P\,||\,Q)$ tells us something about how far Q is from P.

Q2: When can we compute it?

$$\mathrm{SD}(P\,||\,Q) = \sup_{g \in \mathcal{G}} |\mathbb{E}_{X\sim Q}[\mathcal{S}_P[g](X)]|$$

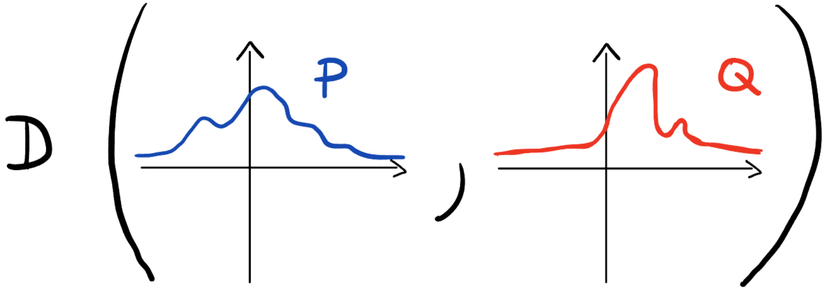


= ??

$$\mathbb{E}_{X\sim Q}[\mathcal{S}_P[g](X)] = ?$$

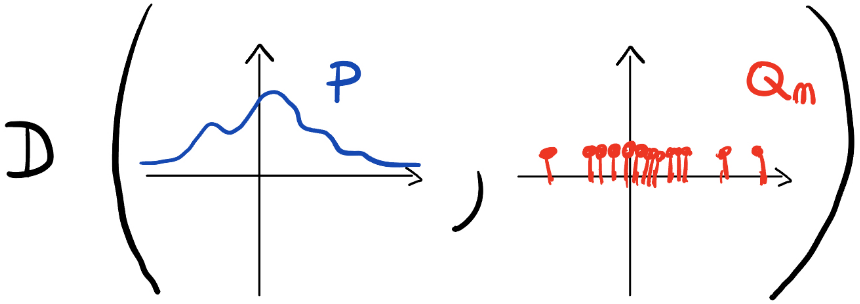
Q2: When can we compute it?

$$\text{SD}(P\,||\,Q) = \sup_{g \in \mathcal{G}} |\mathbb{E}_{X \sim Q}[\mathcal{S}_P[g](X)]|$$



= ??

$$\mathbb{E}_{X \sim Q}[\mathcal{S}_P[g](X)] = ?$$

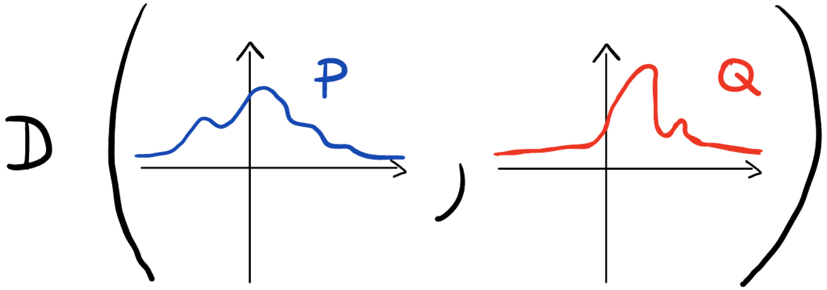


=

$$\mathbb{E}_{X \sim Q_n}[\mathcal{S}_P[g](X)] = \frac{1}{n}\sum_{i=1}^{n} \mathcal{S}_P[g](x_i)$$

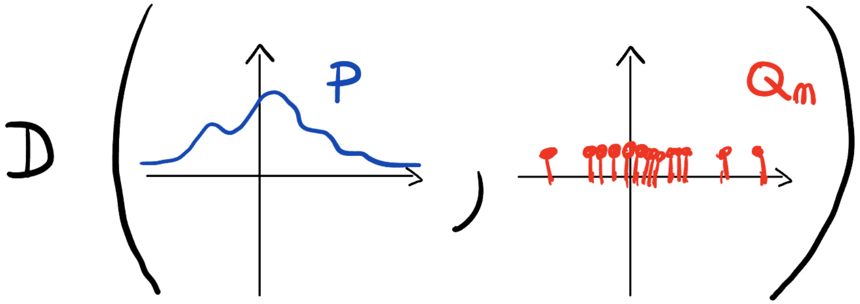
Q2: When can we compute it?

$$\mathrm{SD}(P||Q) = \sup_{g \in \mathcal{G}} |\mathbb{E}_{X\sim Q}[\mathcal{S}_P[g](X)]|$$



= ??

$$\mathbb{E}_{X\sim Q}[\mathcal{S}_P[g](X)] = ?$$



=

$$\mathbb{E}_{X\sim Q_n}[\mathcal{S}_P[g](X)] = \frac{1}{n}\sum_{i=1}^{n} \mathcal{S}_P[g](x_i)$$

Answer 1: Compare to an empirical measure/dataset! $Q_n = \frac{1}{n}\sum_{i=1}^{n} \delta_{x_i}$

Q2: When can we compute it?

$$\text{SD}\left(P \,\middle|\middle|\, \frac{1}{n}\sum_{i=1}^{n}\delta_{x_i}\right) = \sup_{g\in\mathcal{G}}\left|\frac{1}{n}\sum_{i=1}^{n}\mathcal{S}_P[g](X)\right|$$

Answer 2: When $\mathcal{G}$ is not too large, so as to make this supremum tractable.

Q2: When can we compute it?

$$\text{SD}\left(P \middle|\middle| \frac{1}{n}\sum_{i=1}^{n} \delta_{x_i}\right) = \sup_{g \in \mathcal{G}} \left| \frac{1}{n}\sum_{i=1}^{n} \mathcal{S}_P[g](X) \right|$$

Answer 2: When $\mathcal{G}$ is not too large, so as to make this supremum tractable.

- We do however need to make sure $\mathcal{G}$ is not too small either, as otherwise the measure of similarity is not useful for anything.

Q2: When can we compute it?

$$\text{SD}\left(P \middle|\middle| \frac{1}{n}\sum_{i=1}^{n}\delta_{x_i}\right) = \sup_{g\in\mathcal{G}}\left|\frac{1}{n}\sum_{i=1}^{n}\mathcal{S}_P[g](X)\right|$$

Answer 2: When $\mathcal{G}$ is not too large, so as to make this supremum tractable.

- We do however need to make sure $\mathcal{G}$ is not too small either, as otherwise the measure of similarity is not useful for anything.

Goal: Choose $\mathcal{G}$ the largest possible such that SD is still tractable!

Example: SD for $N(0,\sigma^2)$

$$\mathrm{SD}(P\,||\,Q_n) = \sup_{g\in\mathcal{G}} \left| \frac{1}{n}\sum_{i=1}^{n} \mathcal{S}_P[g](x_i) \right|$$

$$= \sup_{g \text{ almost diff.}} \left| \frac{1}{n}\sum_{i=1}^{n} \sigma^2 g'(x_i) - x_i g(x_i)] \right|$$

Example: SD for $N(0,\sigma^2)$

$$\mathrm{SD}(P\,||\,Q_n) = \sup_{g\in\mathcal{G}} \left| \frac{1}{n}\sum_{i=1}^{n} \mathcal{S}_P[g](x_i) \right|$$

$$= \sup_{g \text{ almost diff.}} \left| \frac{1}{n}\sum_{i=1}^{n} \sigma^2 g'(x_i) - x_i g(x_i)] \right| = ??$$

Example: SD for $N(0,\sigma^2)$

$$\text{SD}(P\,||\,Q_n) = \sup_{g \in \mathcal{G}} \left| \frac{1}{n} \sum_{i=1}^{n} \mathcal{S}_P[g](x_i) \right|$$

$$= \sup_{g \text{ almost diff.}} \left| \frac{1}{n} \sum_{i=1}^{n} \sigma^2 g'(x_i) - x_i g(x_i)] \right| = ??$$

The Stein class of almost differentiable functions is way too large for us to be able to find this supremum. Not so helpful as a computational tool….

Example 1: Graph-Stein discrepancies

$$\mathrm{GSD}(P \,||\, Q) = \sup_{g \in \mathcal{G}} \left\| \mathbb{E}_{X \sim P}[\mathcal{S}_P[g](X)] \right\|$$

$$\mathcal{G} = \left\{ g : \max \left(\|g(v)\|_\infty, \|\nabla g(v)\|_\infty, \frac{\|g(x) - g(y)\|_\infty}{\|x - y\|_1}, \frac{\|\nabla g(x) - \nabla g(y)\|_\infty}{\|x - y\|_1} \right) \leq 1, \right.$$
$$\left. \frac{\|g(x) - g(y) - \nabla g(x)(x-y)\|_\infty}{\frac{1}{2}\|x - y\|_1^2} \leq 1, \frac{\|g(x) - g(y) - \nabla g(y)(x-y)\|_\infty}{\frac{1}{2}\|x - y\|_1^2} \leq 1, \quad \forall x, y \in E, v \in \{x_i\}_{i=1}^n \right\}$$

$\mathcal{S}_P$ is the Langevin Stein operator.

Gorham, J., & Mackey, L. (2015). Measuring sample quality with Stein's method. *Advances in Neural Information Processing Systems*, 226–234.

Example 1: Graph-Stein discrepancies

$$\text{GSD}(P\,||\,Q) = \sup_{g\in\mathcal{G}} \left\| \mathbb{E}_{X\sim P}[\mathcal{S}_P[g](X)] \right\|$$

$$\mathcal{G} = \Bigg\{ g : \text{max}\left(\|g(v)\|_\infty, \|\nabla g(v)\|_\infty, \frac{\|g(x)-g(y)\|_\infty}{\|x-y\|_1}, \frac{\|\nabla g(x)-\nabla g(y)\|_\infty}{\|x-y\|_1} \right) \leq 1,$$
$$\frac{\|g(x)-g(y)-\nabla g(x)(x-y)\|_\infty}{\frac{1}{2}\|x-y\|_1^2} \leq 1, \frac{\|g(x)-g(y)-\nabla g(y)(x-y)\|_\infty}{\frac{1}{2}\|x-y\|_1^2} \leq 1, \quad \forall x, y \in E, v \in \{x_i\}_{i=1}^n \Bigg\}$$

$\mathcal{S}_P$ is the Langevin Stein operator.

The class is small enough that we can find the maximum through linear programming!

Gorham, J., & Mackey, L. (2015). Measuring sample quality with Stein's method. *Advances in Neural Information Processing Systems*, 226–234.

Example 2: Hyvarinen divergence

$$\mathrm{SM}(P\,||\,Q) = \sup_{g \in \mathcal{G}} \|\mathbb{E}_{X \sim Q}[\mathcal{S}_P[g](X)]\|$$

$$\mathcal{G} = \left\{ g = (g_1, \ldots, g_d) \in C^2(\mathcal{X}, \mathbb{R}^d) \cap L^2(\mathcal{X}; \mathbb{Q}) : \|g\|_{L^2(\mathcal{X};\mathbb{Q})} \leq 1 \right\}$$

$\mathcal{S}_P$ is the Langevin Stein operator.

Barp, A., Briol, F.-X., Duncan, A. B., Girolami, M., & Mackey, L. (2019). Minimum Stein discrepancy estimators. *Neural Information Processing Systems*, 12964–12976.

Example 2: Hyvarinen divergence

$$\text{SM}(P\,||\,Q) = \sup_{g \in \mathcal{G}} \|\mathbb{E}_{X \sim Q}[\mathcal{S}_P[g](X)]\|$$

$$\mathcal{G} = \left\{ g = (g_1, \ldots, g_d) \in C^2(\mathcal{X}, \mathbb{R}^d) \cap L^2(\mathcal{X}; \mathbb{Q}) : \|g\|_{L^2(\mathcal{X};\mathbb{Q})} \leq 1 \right\}$$

$\mathcal{S}_P$ is the Langevin Stein operator.

The class is small enough that we can attain the maximum!

Barp, A., Briol, F.-X., Duncan, A. B., Girolami, M., & Mackey, L. (2019). Minimum Stein discrepancy estimators. *Neural Information Processing Systems*, 12964–12976.

Example 2: Hyvarinen divergence

$$\text{SM}(P\,||\,Q) = \sup_{g\in\mathcal{G}} \|\mathbb{E}_{X\sim Q}[\mathcal{S}_P[g](X)]\|$$

$$= \mathbb{E}_{X\sim Q}\left[\|\nabla_x \log p(X) - \nabla_x \log q(X)\|_2^2\right]$$

Example 2: Hyvarinen divergence

$$\text{SM}(P \,||\, Q) = \sup_{g \in \mathcal{G}} \|\mathbb{E}_{X \sim Q}[\mathcal{S}_P[g](X)]\|$$

$$= \mathbb{E}_{X \sim Q}\left[\|\nabla_x \log p(X) - \nabla_x \log q(X)\|_2^2\right]$$

- We are comparing the score functions for P and Q, and so this is often called the **score-matching divergence**.

Example 2: Hyvarinen divergence

$$\text{SM}(P \,||\, Q) = \sup_{g \in \mathcal{G}} \|\mathbb{E}_{X \sim Q}[\mathcal{S}_P[g](X)]\|$$

$$= \mathbb{E}_{X \sim Q}\left[\|\nabla_x \log p(X) - \nabla_x \log q(X)\|_2^2\right]$$

- We are comparing the score functions for P and Q, and so this is often called the **score-matching divergence**.
- This is the method which powers many modern generative models such as diffusion models.

Example 2: Hyvarinen divergence

$$\mathrm{SM}(P\,||\,Q) = \sup_{g \in \mathcal{G}} \|\mathbb{E}_{X\sim Q}[\mathcal{S}_P[g](X)]\|$$

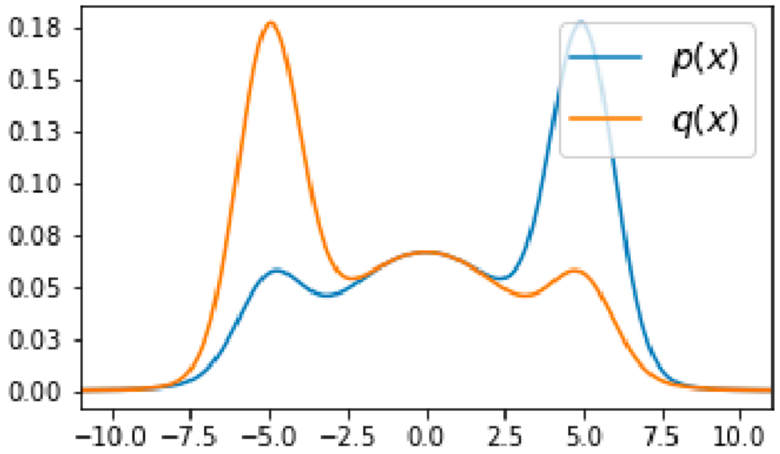
$$= \mathbb{E}_{X\sim Q}\left[\|\nabla_x \log p(X) - \nabla_x \log q(X)\|_2^2\right]$$

- We are comparing the score functions for P and Q, and so this is often called the **score-matching divergence**.
- This is the method which powers many modern generative models such as diffusion models.

Hyvärinen, A. (2006). Estimation of non-normalized statistical models by score matching. Journal of Machine Learning Research, 6, 695–708.

Example 2: Hyvarinen divergence

$$\text{SM}(P\,||\,Q) = \mathbb{E}_{X\sim Q}\left[\|\nabla_x \log p(X) - \nabla_x \log q(X)\|_2^2\right]$$

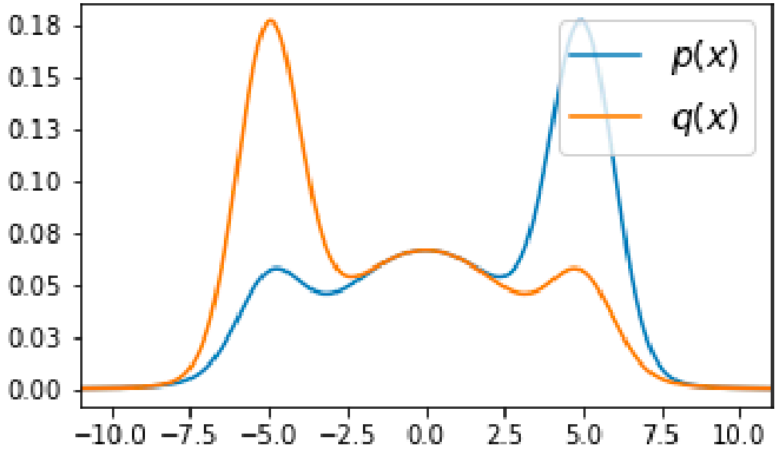


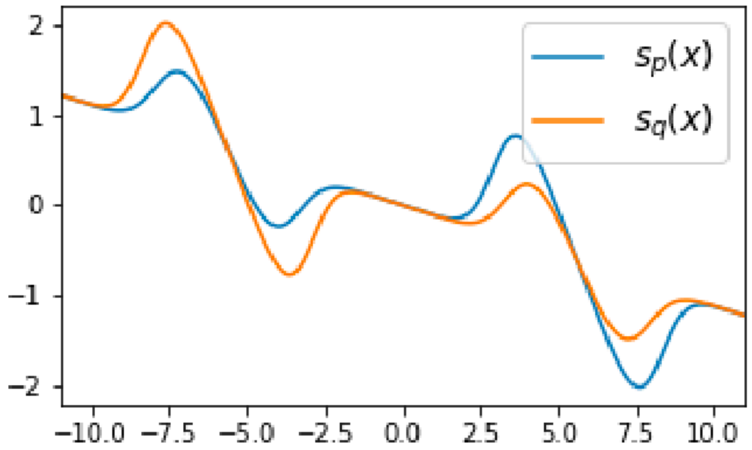
Cannot use this as we typically don't know the densities exactly….

$$p(x) = \frac{\tilde{p}(x)}{C}$$

Example 2: Hyvarinen divergence

$$\mathrm{SM}(P||Q) = \mathbb{E}_{X\sim Q}\left[\|\nabla_x \log p(X) - \nabla_x \log q(X)\|_2^2\right]$$





Cannot use this as we typically don't know the densities exactly….

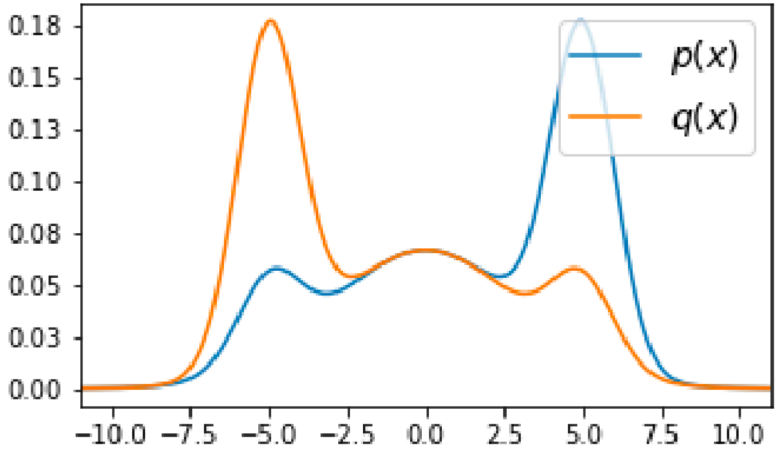
$$p(x) = \frac{\tilde{p}(x)}{C}$$

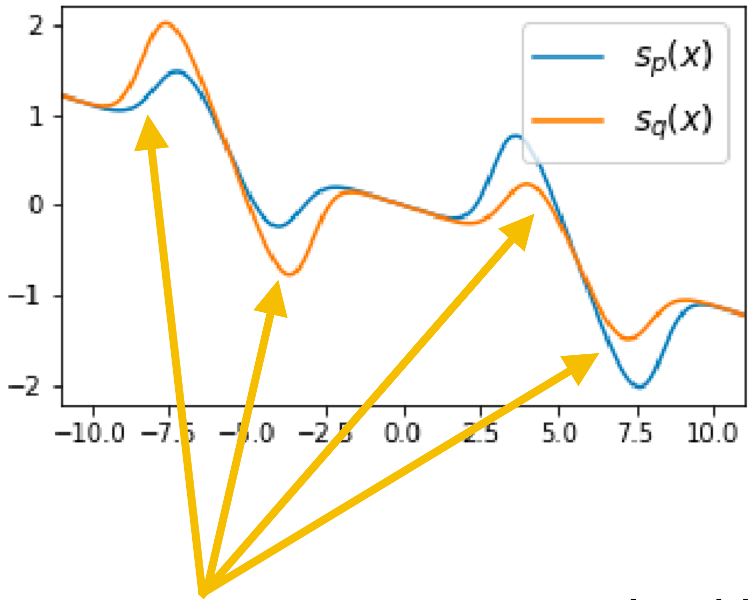
Luckily….

$$s_p(x) = \nabla_x \log p(x) = \nabla_x \log \tilde{p}(x)$$

Example 2: Hyvarinen divergence

$$\text{SM}(P\,||\,Q) = \mathbb{E}_{X\sim Q}\left[\|\nabla_x \log p(X) - \nabla_x \log q(X)\|_2^2\right]$$





Cannot use this as we typically don't know the densities exactly….

$$p(x) = \frac{\tilde{p}(x)}{C}$$

Not the same!

Luckily….

$$s_p(x) = \nabla_x \log p(x) = \nabla_x \log \tilde{p}(x)$$

Example 2: Hyvarinen divergence

$$\mathrm{SM}(P \,||\, Q) = \mathbb{E}_{X \sim Q}\left[\|\nabla_x \log p(X) - \nabla_x \log q(X)\|_2^2\right]$$

Example 2: Hyvarinen divergence

$$\text{SM}(P\,||\,Q) = \mathbb{E}_{X\sim Q}\left[\|\nabla_x \log p(X) - \nabla_x \log q(X)\|_2^2\right]$$

- Sadly we usually do not have access to $\nabla \log q$ in most applications (typically Q is some unknown data-generating process).

Example 2: Hyvarinen divergence

$$\text{SM}(P \,||\, Q) = \mathbb{E}_{X \sim Q}\left[\|\nabla_x \log p(X) - \nabla_x \log q(X)\|_2^2\right]$$

- Sadly we usually do not have access to $\nabla \log q$ in most applications (typically Q is some unknown data-generating process).

- The only type of application where this can be used is for parameter estimation/generative modelling, since we can typically still evaluate the divergence up to some additive constant.

More on this shortly….

Example 3: Kernel Stein discrepancies

$$\text{KSD}(P || Q) = \sup_{g \in \mathcal{G}} \|\mathbb{E}_{X \sim Q}[\mathcal{S}_P[g](X)]\|$$

$$\mathcal{G} = \left\{ g = (g_1, \ldots, g_d) \in \mathcal{H}_k : \|v\|_2 \leq 1 \text{ where } v_i = \|g_i\|_{\mathcal{H}_k} \right\}$$

$\mathcal{S}_P$ is the Langevin Stein operator.

Chwialkowski, K., Strathmann, H., & Gretton, A. (2016). A kernel test of goodness of fit. *International Conference on Machine Learning*, 2606–2615.

Liu, Q., Lee, J. D., & Jordan, M. I. (2016). A kernelized Stein discrepancy for goodness-of-fit tests and model evaluation. *International Conference on Machine Learning*, 276–284.

Example 3: Kernel Stein discrepancies

$$\text{KSD}(P\,||\,Q) = \sup_{g \in \mathcal{G}} ||\mathbb{E}_{X\sim Q}[\mathcal{S}_P[g](X)]||$$

$$\mathcal{G} = \left\{ g = (g_1, \ldots, g_d) \in \mathcal{H}_k : ||v||_2 \leq 1 \text{ where } v_i = ||g_i||_{\mathcal{H}_k} \right\}$$

$\mathcal{S}_P$ is the Langevin Stein operator.

→ The most practical class as it can be evaluated in closed-form!

Chwialkowski, K., Strathmann, H., & Gretton, A. (2016). A kernel test of goodness of fit. *International Conference on Machine Learning*, 2606–2615.

Liu, Q., Lee, J. D., & Jordan, M. I. (2016). A kernelized Stein discrepancy for goodness-of-fit tests and model evaluation. *International Conference on Machine Learning*, 276–284.

Stein’s method as a computational tool

Kernel Stein discrepancies

Reproducing kernels

- A reproducing kernel is any symmetric and positive-semidefinite function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$.

   1. Symmetric means that for any $x, x' \in \mathcal{X}$, $k(x, x') = k(x', x)$.

   2. Positive semi-definite means that for any $x_1, \ldots, x_n$ and $n \in \mathbb{N}$, the Gram matrix $K \in \mathbb{R}^{n \times n}$ (where $K_{ij} = k(x_i, x_j)$) must be positive semidefinite.

(In other words, it can only have nonnegative eigenvalues.)

Reproducing kernels

- A reproducing kernel is any symmetric and positive-semidefinite function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$.

    1. Symmetric means that for any $x, x' \in \mathcal{X}$, $k(x, x') = k(x', x)$.

    2. Positive semi-definite means that for any $x_1, \ldots, x_n$ and $n \in \mathbb{N}$, the Gram matrix $K \in \mathbb{R}^{n \times n}$ (where $K_{ij} = k(x_i, x_j)$) must be positive semidefinite.

        (In other words, it can only have nonnegative eigenvalues.)

One way to think about kernel is as measuring the similarity between points!

Examples of kernels

- **Example 1:** Squared exponential (or Gaussian) kernel:

$$k(x, x') = \lambda \exp\left(-\frac{\|x - x'\|_2^2}{l}\right)$$

Examples of kernels

- **Example 1:** Squared exponential (or Gaussian) kernel:

$$k(x, x') = \lambda \exp\left(-\frac{\|x - x'\|_2^2}{l}\right)$$

- **Example 2:** Inverse-multiquadric kernel:

$$k(x, x') = \lambda \left(\|x - x'\|_2^2 + c\right)^{-\frac{1}{2}}$$

Examples of kernels

- **Example 1:** Squared exponential (or Gaussian) kernel:

$$k(x, x') = \lambda \exp\left(-\frac{\|x - x'\|_2^2}{l}\right)$$

- **Example 2:** Inverse-multiquadric kernel:

$$k(x, x') = \lambda \left(\|x - x'\|_2^2 + c\right)^{-\frac{1}{2}}$$

- **Example 3:** Polynomial kernel

$$k(x, x') = \lambda(c + x^\top x')^p$$

Properties of kernels

- Many of the kernels we have seen so far only depend on x, x' through $||x - x'||$. They are therefore called **translation invariant.**

- They also all take the following form for some bounded $\phi : \mathcal{X} \to \mathbb{R}_+$, making them **radial**
$$k(x, x') = \lambda^2 \phi \left(-\frac{||x - x'||_2}{l^2} \right)$$

- All of these kernels are bounded, which is a super helpful property for most of what we will do.

Kernel hyperparameters

- The parameter λ is called the **amplitude,** whilst the parameter l is called the **lengthscale.**

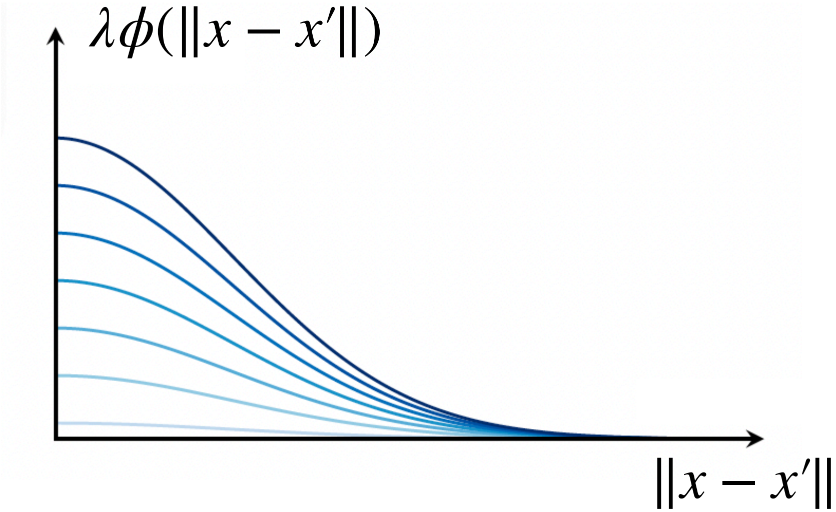
$$k(x, x') = \lambda^2 \phi \left(-\frac{\|x - x'\|_2}{l^2} \right)$$

Kernel hyperparameters

- The parameter λ is called the **amplitude,** whilst the parameter l is called the **lengthscale.**

$$k(x, x') = \lambda^2 \phi \left(-\frac{\|x - x'\|_2}{l^2} \right)$$

Varying amplitude parameter



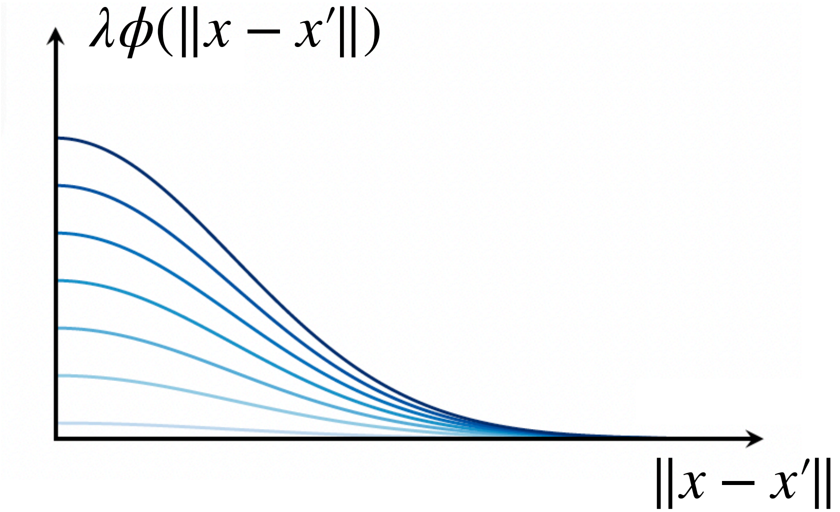
[Garnett, 2023]

Kernel hyperparameters

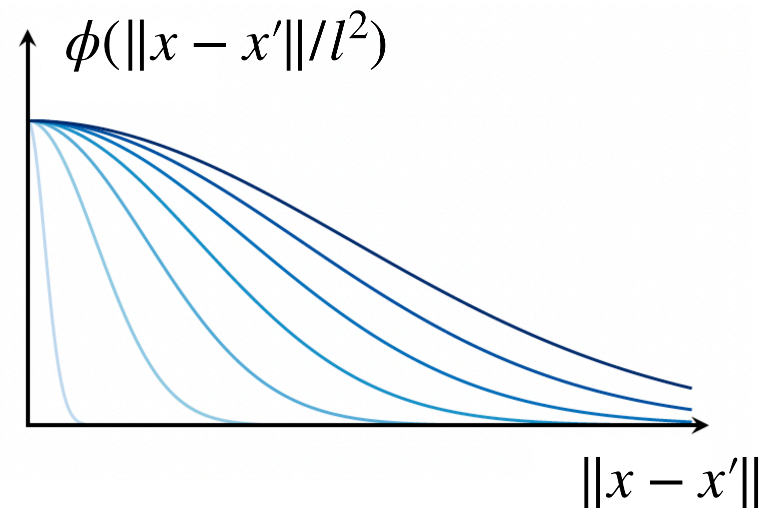
- The parameter λ is called the **amplitude,** whilst the parameter l is called the **lengthscale.**

$$k(x, x') = \lambda^2 \phi\left(-\frac{\|x - x'\|_2}{l^2}\right)$$

Varying amplitude parameter



Varying lengthscale parameter



[Garnett, 2023]

Reproducing kernel Hilbert Spaces

- Let $\mathscr{H}_k$ be a Hilbert space of functions from $\mathscr{X}$ to $\mathbb{R}$ (i.e. a complete inner-product space).
- We say that $\mathscr{H}_k$ is an RKHS if and only if it has a reproducing kernel; ie. a kernel which satisfies:
    - $\forall x \in \mathscr{X}, k(\cdot, x) \in \mathscr{H}_k$
    - $\forall x \in \mathscr{X}, \forall f \in \mathscr{H}_k, \langle f, k(\cdot, x) \rangle_{\mathscr{H}_k} = f(x)$

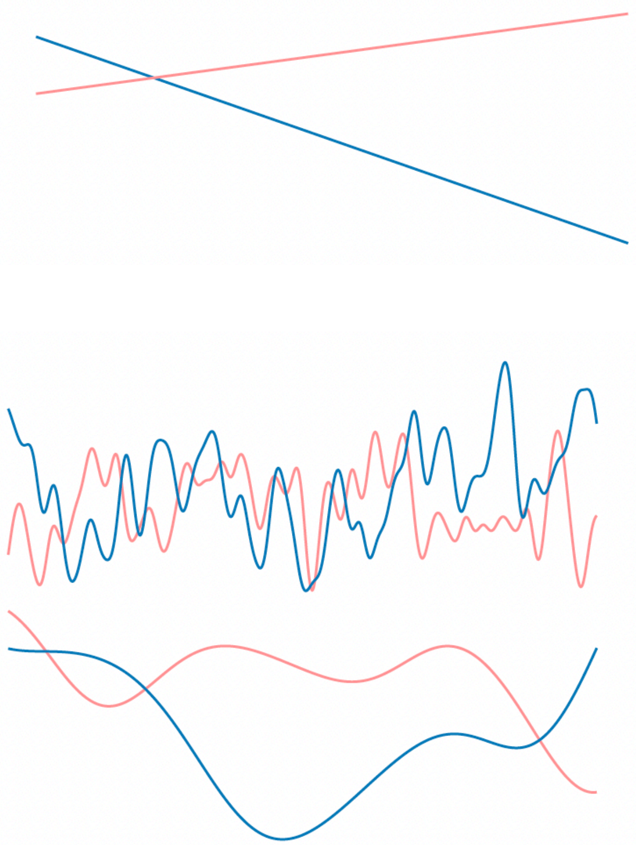
Reproducing kernel Hilbert Spaces

- Let $\mathscr{H}_k$ be a Hilbert space of functions from $\mathscr{X}$ to $\mathbb{R}$ (i.e. a complete inner-product space).
- We say that $\mathscr{H}_k$ is an RKHS if and only if it has a reproducing kernel; ie. a kernel which satisfies:
    - $\forall x \in \mathscr{X}, k(\cdot, x) \in \mathscr{H}_k$
    - $\forall x \in \mathscr{X}, \forall f \in \mathscr{H}_k, \langle f, k(\cdot, x) \rangle_{\mathscr{H}_k} = f(x)$
- **Intuition (not fully rigorous):** I like to think of RKHS functions as functions of the form:

$$f(x) = \sum_{i=1}^{n} w_i k(x, x_i)$$

Examples of RKHS

[Garnett, 2023]



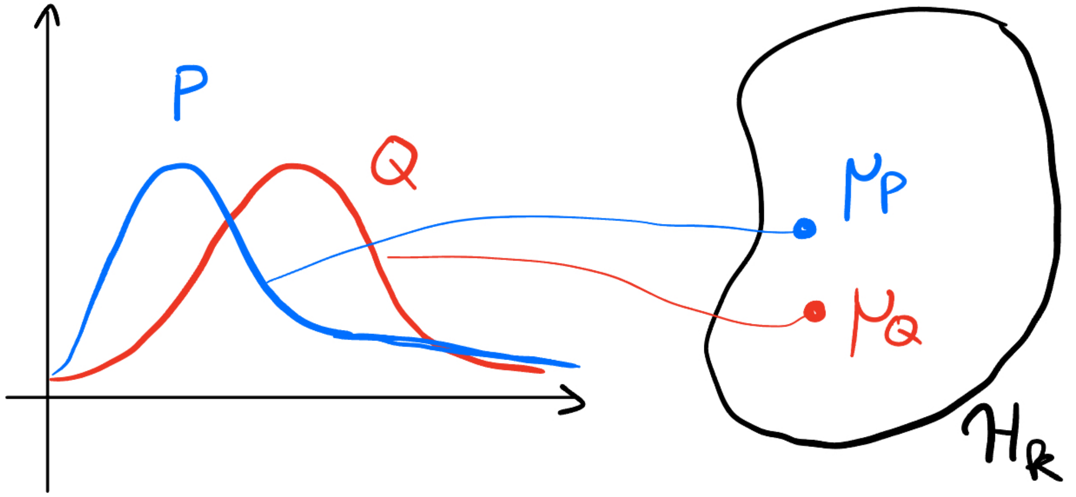
- **Example 1:** If we take an order-1 polynomial kernel, the RKHS is simply the space of straight lines!

- **Example 2:** If we take a Gaussian or inverse-multi quadric kernel, the RKHS is a space of infinitely smooth function!

Kernel mean embeddings

- Due to its nice properties, we may **want to represent probability distributions as functions** in an RKHS.
- This is achieved through the kernel mean embedding:

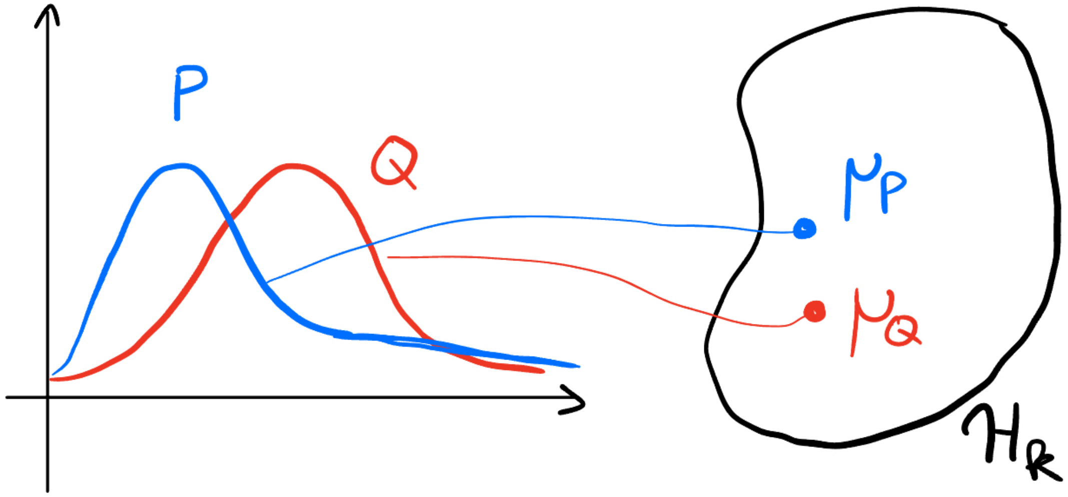
$$\mu_P(x) = \int k(x,y)p(y)dy$$



Kernel mean embeddings

- Due to its nice properties, we may **want to represent probability distributions as functions** in an RKHS.
- This is achieved through the kernel mean embedding:

$$\mu_P(x) = \int k(x,y)p(y)dy$$

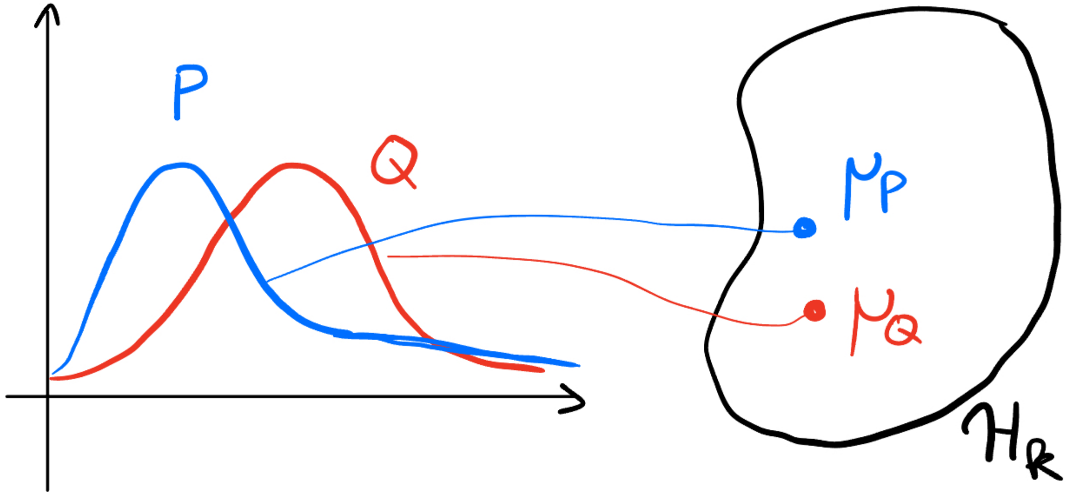


→ Working with functions is a lot easier than working with distributions… This is another convenient characterisation!!

Maximum mean discrepancy

- For example, we can just compare two distributions based on the distance between their kernel mean embeddings.
- This is called the **maximum mean discrepancy (MMD)**!

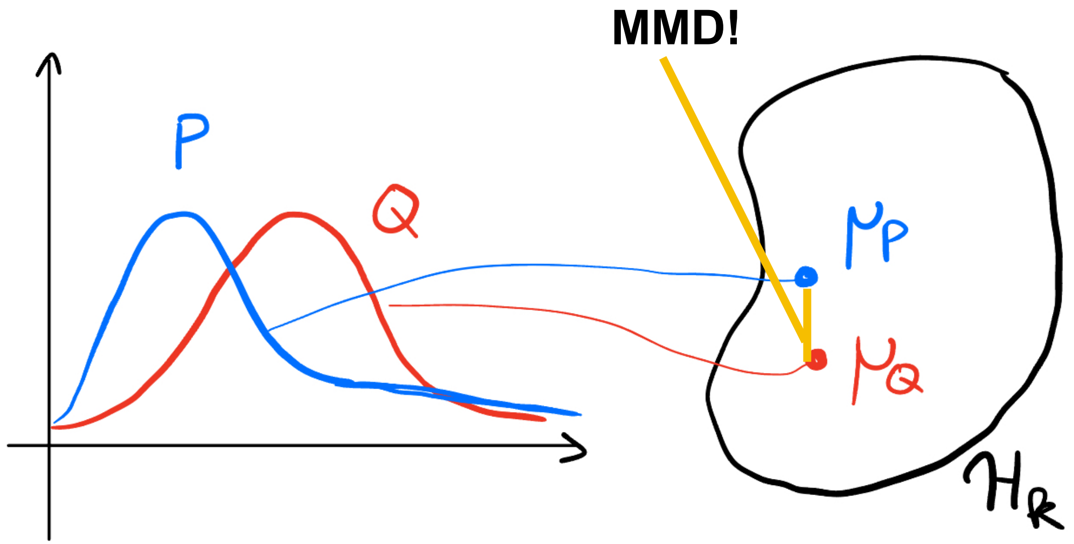
$$\text{MMD}(P\,||\,Q) = \|\mu_P - \mu_Q\|_{\mathscr{H}_k}$$



Maximum mean discrepancy

- For example, we can just compare two distributions based on the distance between their kernel mean embeddings.
- This is called the **maximum mean discrepancy (MMD)**!

$$\text{MMD}(P\,||\,Q) = \|\mu_P - \mu_Q\|_{\mathcal{H}_k}$$



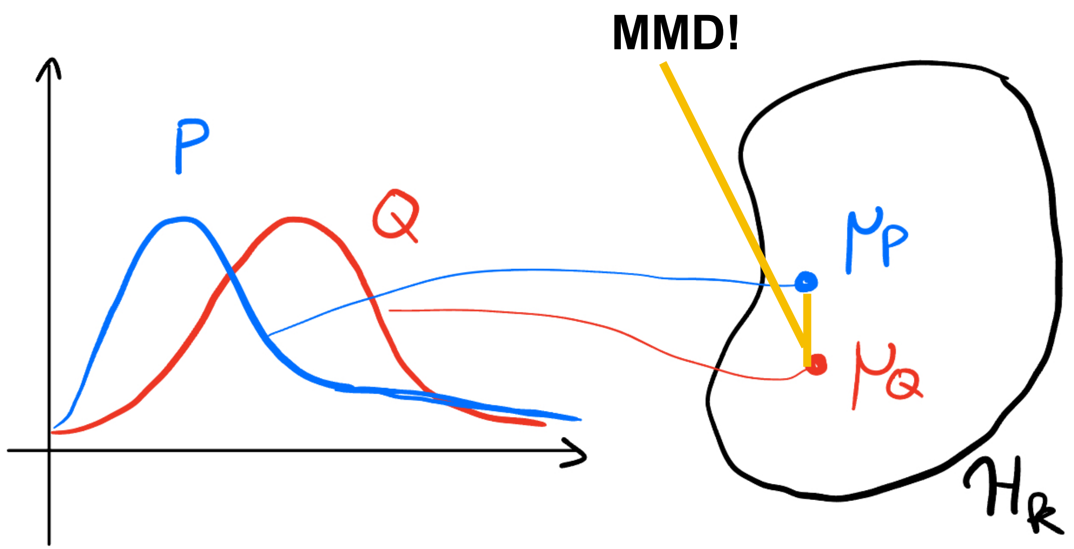
Maximum mean discrepancy

- For example, we can just compare two distributions based on the distance between their kernel mean embeddings.

- This is called the **maximum mean discrepancy (MMD)**!

$$\text{MMD}(P\,||\,Q) = ||\mu_P - \mu_Q||_{\mathscr{H}_k}$$

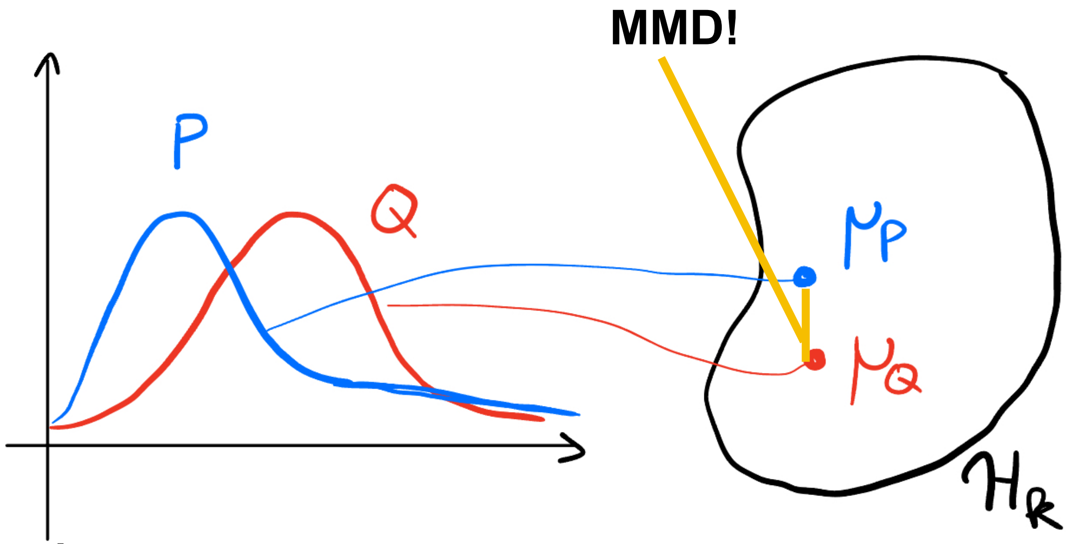
- This is actually an integral probability metric based on all functions of a fixed size in this RKHS!



Maximum mean discrepancy

- For example, we can just compare two distributions based on the distance between their kernel mean embeddings.
- This is called the **maximum mean discrepancy (MMD)**!

$$\text{MMD}(P\,||\,Q) = \|\mu_P - \mu_Q\|_{\mathcal{H}_k}$$



- This is actually an integral probability metric based on all functions of a fixed size in this RKHS!
- Of course we often can't compute the kernel mean embedding since it is an integral…

Stein RKHS

- We can use our favourite tool to make these embeddings tractable!
- Consider $g(x) = (g_1(x), \ldots, g_d(x))$ where each $g_i(x) \in \mathcal{H}_k$. Then:

$$h(x) = \mathcal{S}_P[g](x) \in \mathcal{H}_{k_p}$$

  where k_p is another reproducing kernel.

- All the functions in $\mathcal{H}_{k_p}$ have **mean zero under P by construction**, and therefore we definitely have that:

$$\mu_p(x) = \int k_p(x, y)p(y)dx = 0\,.$$

Kernel Stein discrepancies

$$\text{KSD}(P \,||\, Q) = \sup_{g \in \mathcal{G}} \|\mathbb{E}_{X \sim Q}[\mathcal{S}_P[g](X)]\|$$

- The Stein discrepancy with the RKHS $\mathcal{H}_k$ is equivalent to the the MMD with kernel k_p!

Chwialkowski, K., Strathmann, H., & Gretton, A. (2016). A kernel test of goodness of fit. *International Conference on Machine Learning*, 2606–2615.

Liu, Q., Lee, J. D., & Jordan, M. I. (2016). A kernelized Stein discrepancy for goodness-of-fit tests and model evaluation. *International Conference on Machine Learning*, 276–284.

Expression for the Langevin KSD

- The Stein discrepancy can be simplified to

$$\text{KSD}(P\,||\,Q_n) = \sqrt{\frac{1}{n^2}\sum_{i,j=1}^{n} k_P(x_i, x_j)}$$

$$k_P(x,x') = k(x,x')\langle \nabla_x \log p(x), \nabla_{x'} \log p(x')\rangle + \langle \nabla_x k(x,x'), \nabla_{x'} \log p(x')\rangle$$
$$+\langle \nabla_{x'} k(x,x'), \nabla_x \log p(x)\rangle + \text{Tr}(\nabla_x \nabla_{x'} k(x,x'))$$

- The function k_P is a **Stein reproducing kernel** (i.e. it is also a kernel!)

Expression for the Langevin KSD

- The Stein discrepancy can be simplified to

$$\mathrm{KSD}(P\,||\,Q_n) = \sqrt{\frac{1}{n^2}\sum_{i,j=1}^{n} k_P(x_i, x_j)}$$

$$k_P(x,x') = k(x,x')\langle \nabla_x \log p(x), \nabla_{x'} \log p(x')\rangle + \langle \nabla_x k(x,x'), \nabla_{x'} \log p(x')\rangle$$
$$+ \langle \nabla_{x'} k(x,x'), \nabla_x \log p(x)\rangle + \mathrm{Tr}(\nabla_x \nabla_{x'} k(x,x'))$$

→ Looks complicated but it's all straightforward to compute!

Kernel derivatives

- We can look at the example of the Gaussian kernel:

$$k(x, x') = \lambda \exp\left(-\frac{\|x - x'\|_2^2}{l}\right)$$

Kernel derivatives

- We can look at the example of the Gaussian kernel:

$$k(x, x') = \lambda \exp\left(-\frac{\|x - x'\|_2^2}{l}\right) \qquad \nabla_x k(x, x') = -\frac{2\lambda(x - x')}{l^2} \exp\left(-\frac{\|x - x'\|_2^2}{l}\right)$$

Kernel derivatives

- We can look at the example of the Gaussian kernel:

$$k(x, x') = \lambda \exp\left(-\frac{\|x - x'\|_2^2}{l}\right) \qquad \nabla_x k(x, x') = -\frac{2\lambda(x - x')}{l^2} \exp\left(-\frac{\|x - x'\|_2^2}{l}\right)$$

$$\mathrm{Tr}(\nabla_x \nabla_{x'} k(x, x')) = \frac{2\lambda\left(l^2 - 2\sum_{i=1}^{d}(x_i - x_i')^2\right)}{l^4} \exp\left(-\frac{\|x - x'\|_2^2}{l}\right)$$

Kernel derivatives

- We can look at the example of the Gaussian kernel:

$$k(x, x') = \lambda \exp\left(-\frac{\|x - x'\|_2^2}{l} \right) \qquad \nabla_x k(x, x') = -\frac{2\lambda(x - x')}{l^2} \exp\left(-\frac{\|x - x'\|_2^2}{l} \right)$$

$$\mathrm{Tr}(\nabla_x \nabla_{x'} k(x, x')) = \frac{2\lambda \left(l^2 - 2\sum_{i=1}^{d} (x_i - x_i')^2 \right)}{l^4} \exp\left(-\frac{\|x - x'\|_2^2}{l} \right)$$

→ This is indeed straightforward to compute!

Computational complexity of the KSD

$$\text{KSD}(P\,||\,Q_n) = \sqrt{\frac{1}{n^2}\sum_{i,j=1}^{n} k_P(x_i, x_j)}$$

$$k_P(x, x') = k(x, x')\langle \nabla_x \log p(x), \nabla_{x'} \log p(x')\rangle + \langle \nabla_x k(x, x'), \nabla_{x'} \log p(x')\rangle$$

$$+\langle \nabla_{x'} k(x, x'), \nabla_x \log p(x)\rangle + \text{Tr}(\nabla_x \nabla_{x'} k(x, x'))$$

Computational complexity of the KSD

$$\text{KSD}(P\,||\,Q_n) = \sqrt{\frac{1}{n^2}\sum_{i,j=1}^{n} k_P(x_i, x_j)}$$

$$k_P(x,x') = k(x,x')\langle \nabla_x \log p(x), \nabla_{x'} \log p(x')\rangle + \langle \nabla_x k(x,x'), \nabla_{x'} \log p(x')\rangle$$
$$+\langle \nabla_{x'} k(x,x'), \nabla_x \log p(x)\rangle + \text{Tr}(\nabla_x \nabla_{x'} k(x,x'))$$

- The computational complexity of each k_P evaluation is $O(d)$.

Computational complexity of the KSD

$$\text{KSD}(P\,||\,Q_n) = \sqrt{\frac{1}{n^2}\sum_{i,j=1}^{n} k_P(x_i, x_j)}$$

$$k_P(x,x') = k(x,x')\langle \nabla_x \log p(x), \nabla_{x'} \log p(x')\rangle + \langle \nabla_x k(x,x'), \nabla_{x'} \log p(x')\rangle$$
$$+ \langle \nabla_{x'} k(x,x'), \nabla_x \log p(x)\rangle + \text{Tr}(\nabla_x \nabla_{x'} k(x,x'))$$

- The computational complexity of each k_P evaluation is $O(d)$.
- There are $O(n^2)$ evaluations of k_P in the KSD expression.

Computational complexity of the KSD

$$\text{KSD}(P\,||\,Q_n) = \sqrt{\frac{1}{n^2}\sum_{i,j=1}^{n} k_P(x_i, x_j)}$$

$$k_P(x, x') = k(x, x')\langle \nabla_x \log p(x), \nabla_{x'} \log p(x')\rangle + \langle \nabla_x k(x, x'), \nabla_{x'} \log p(x')\rangle$$
$$+ \langle \nabla_{x'} k(x, x'), \nabla_x \log p(x)\rangle + \text{Tr}(\nabla_x \nabla_{x'} k(x, x'))$$

- The computational complexity of each k_P evaluation is $O(d)$.
- There are $O(n^2)$ evaluations of k_P in the KSD expression.

→ Total cost is $O(n^2 d)$!

Scalable Stein discrepancies

- It is possible to bring down the cost to linear (rather than quadratic) in n through very accurate approximations (i.e. random features).
- When P is a posterior based on a lot of data points, the cost of each score function evaluation can be prohibitive. Approximations based on stochastic estimates of the score can be used in those cases.

Jitkrittum, W., Xu, W., Szabo, Z., Fukumizu, K., & Gretton, A. (2017). A linear-time kernel goodness-of-fit test. *NeurIPS*.

Huggins, J. H., & Mackey, L. (2018). Random feature Stein discrepancies. NeurIPS.

Gorham, J., Raj, A., & Mackey, L. (2020). Stochastic Stein discrepancies. NeurIPS.

U-statistic or V-statistic

- Interestingly, this is not the only way to approximate KSD$(P||Q)$:

$$\mathrm{KSD}(P||Q_n) = \sqrt{\frac{1}{n^2}\sum_{i,j=1}^{n} k_P(x_i, x_j)} \quad \longleftarrow \quad \text{V-statistic}$$

$$\widehat{\mathrm{KSD}}(P||Q) = \sqrt{\frac{1}{n(n-1)}\sum_{i,j=1}^{n} k_P(x_i, x_j)} \quad \longleftarrow \quad \text{U-statistic}$$

U-statistic or V-statistic

- Interestingly, this is not the only way to approximate KSD$(P||Q)$:

$$\text{KSD}(P||Q_n) = \sqrt{\frac{1}{n^2}\sum_{i,j=1}^{n} k_P(x_i, x_j)}$$

V-statistic

$$\widehat{\text{KSD}}(P||Q) = \sqrt{\frac{1}{n(n-1)}\sum_{i,j=1}^{n} k_P(x_i, x_j)}$$

U-statistic

- The U-statistic is unbiased but has higher variance, whereas the V-statistic is biased but has lower variance.

Outline (updated)

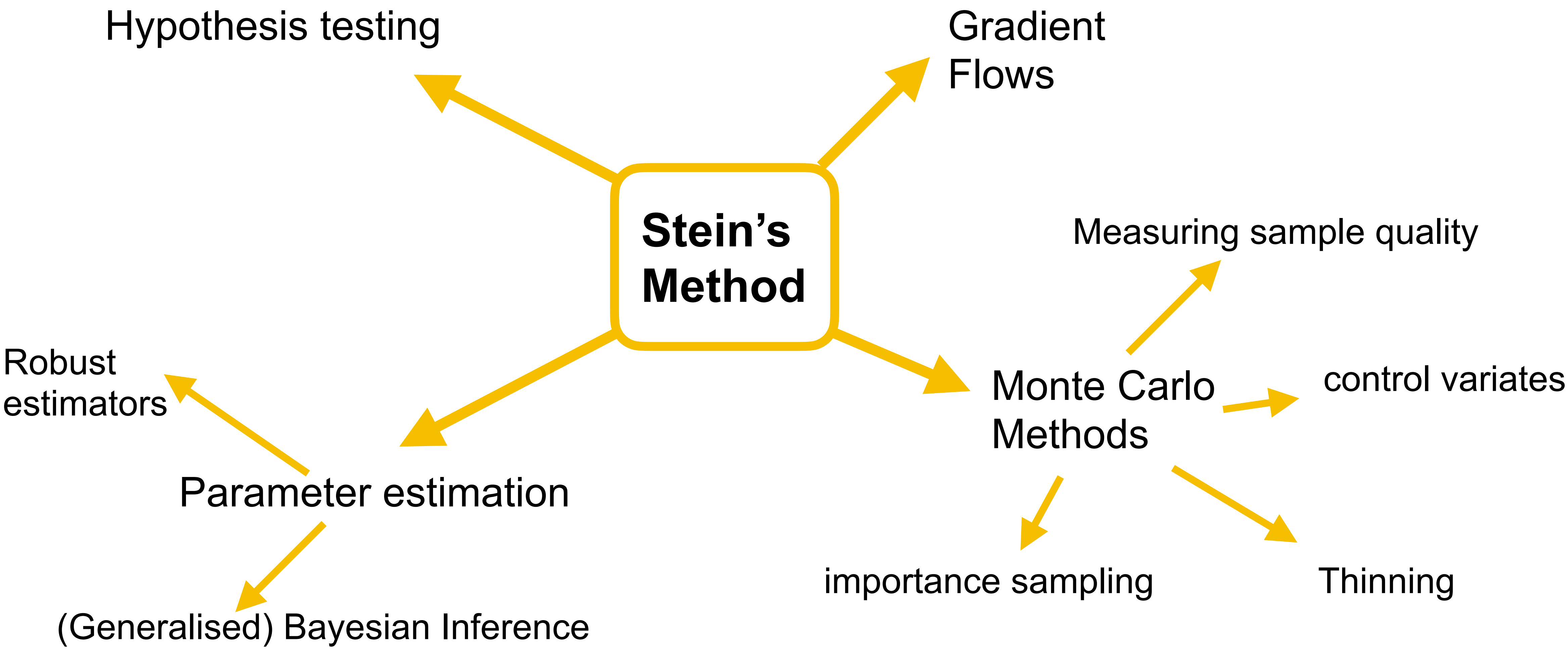
- What is Stein’s method, and why should you care…
- Computational tools based on Stein’s method.
- Some nice (new) algorithms!

Outline (updated)

- What is Stein's method, and why should you care…
- Computational tools based on Stein's method.
- Some nice (new) algorithms!

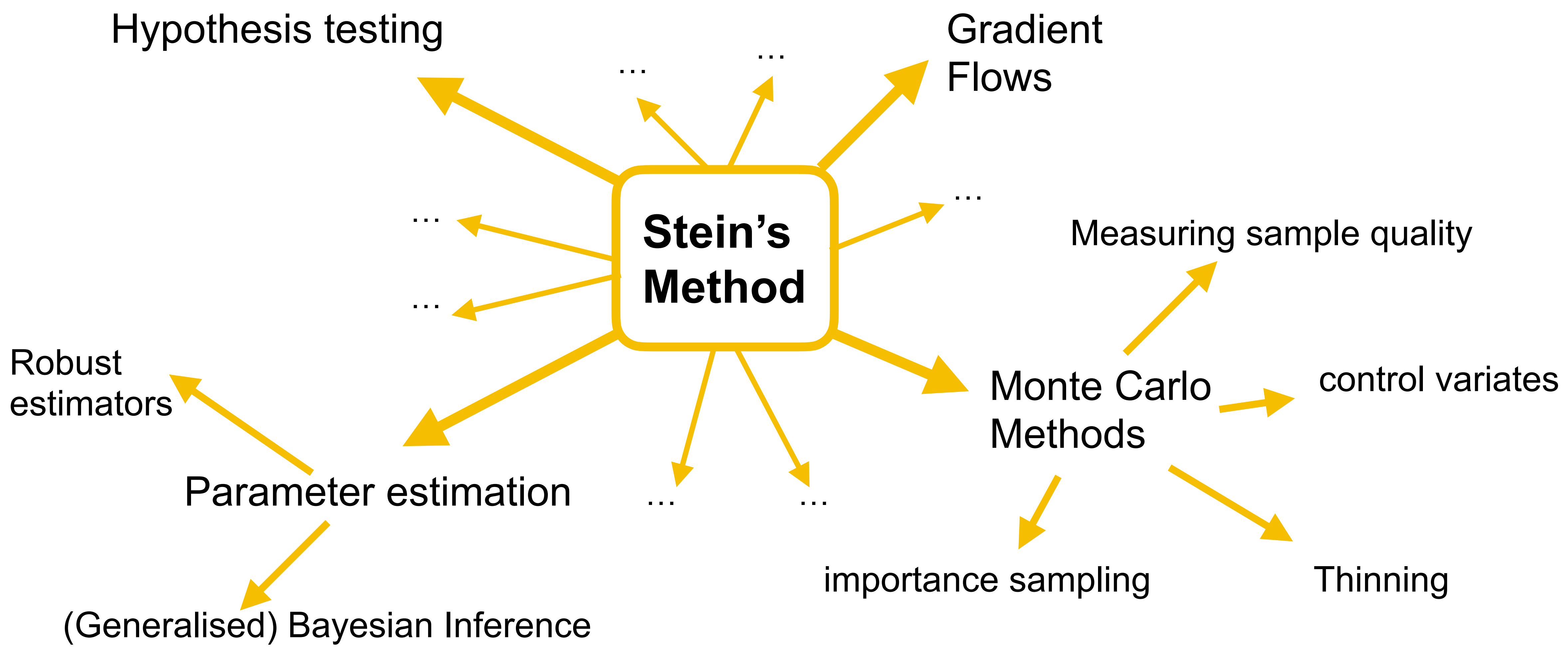
We now have an **amazing hammer** and we can use it to hit pretty much all the nails in computational statistics.

Our nails…



Hypothesis testing
Gradient Flows
Stein's Method
Measuring sample quality
Robust estimators
Monte Carlo Methods
control variates
Parameter estimation
importance sampling
Thinning
(Generalised) Bayesian Inference

Our nails…



Hypothesis testing
…
…
Gradient Flows
…
…
Stein's Method
…
Measuring sample quality
…
Robust estimators
Monte Carlo Methods
control variates
Parameter estimation
…
…
importance sampling
Thinning
(Generalised) Bayesian Inference

Focus of this course

- There are so many topics I could touch upon which I will unfortunately not have time to cover….

Focus of this course

- There are so many topics I could touch upon which I will unfortunately not have time to cover….

Algorithms ~~Theory~~ ~~Large-scale experiments~~

Focus of this course

- There are so many topics I could touch upon which I will unfortunately not have time to cover….

Algorithms ~~Theory~~ ~~Large-scale experiments~~

- My aim is simply to give you some intuition for what can be done with Stein's method, rather than an extensive guide.

Focus of this course

- There are so many topics I could touch upon which I will unfortunately not have time to cover….

Algorithms ~~Theory~~ ~~Large-scale experiments~~

- My aim is simply to give you some intuition for what can be done with Stein's method, rather than an extensive guide.

- I will be biased towards topics on which I have myself worked…

Stein’s method as a computational tool

Hypothesis testing

Goodness-of-fit testing

- In goodness-of-fit testing, we want to answer questions such as:

"Do I have a good model for my observed data?"

"Are the distributional assumptions of my analysis reasonable"

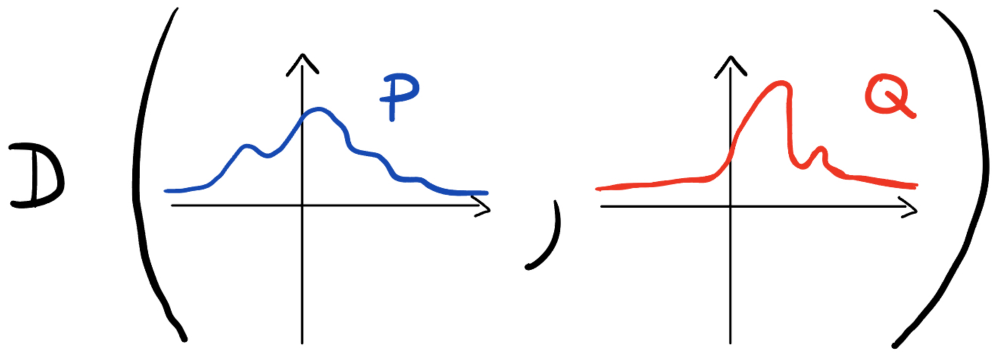
- Given a distribution P and some observed data $\{x_i\}_{i=1}^n \sim Q$, this is formalised as:

$$H_0 : P = Q$$
$$H_1 : P \neq Q$$

Testing with discrepancies

$$H_0 : P = Q$$
$$H_1 : P \neq Q$$

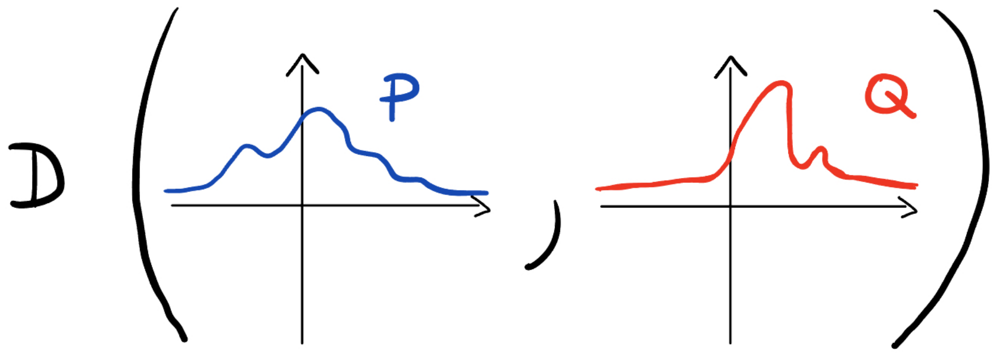
- Assume we have a "reasonable" notion of discrepancy/dissimilarity D. Then a good way to check whether H_0 holds is to compute:



Testing with discrepancies

$$H_0 : P = Q$$
$$H_1 : P \neq Q$$

- Assume we have a “reasonable” notion of discrepancy/dissimilarity D. Then a good way to check whether H_0 holds is to compute:



- If this is zero, we know that $P = Q$!
- If this is strictly greater than zero, we know that $P \neq Q$!

Existing work

- Most existing work focuses on very simple P; e.g. Gaussian, Poisson, etc..

Existing work

- Most existing work focuses on very simple P; e.g. Gaussian, Poisson, etc..

D = L^∞ distance between CDFs $\longrightarrow$ Kolmogorov-Smirnov test

Existing work

- Most existing work focuses on very simple P; e.g. Gaussian, Poisson, etc..

D = L^∞ distance between CDFs → Kolmogorov-Smirnov test

D = weighted L^2 between CDFs → Anderson-Darling test

Existing work

- Most existing work focuses on very simple P; e.g. Gaussian, Poisson, etc..

$D = L^\infty$ distance between CDFs → Kolmogorov-Smirnov test

D = weighted L^2 between CDFs → Anderson-Darling test

- The main reason that these consider only simple P is that the distance is otherwise infeasible to compute/estimate!

Goodness of fit testing with kernels

- Sadly most of these existing tests are very limited in the sense that you have to find a new test for every distribution P you care about….

Goodness of fit testing with kernels

- Sadly most of these existing tests are very limited in the sense that you have to find a new test for every distribution P you care about….

A Kernel Test of Goodness of Fit

Kacper Chwialkowski* KACPER.CHWIALKOWSKI@GMAIL.COM
Heiko Strathmann* HEIKO.STRATHMANN@GMAIL.COM
Arthur Gretton ARTHUR.GRETTON@GMAIL.COM
Gatsby Unit, University College London, United Kingdom

A Kernelized Stein Discrepancy for Goodness-of-fit Tests

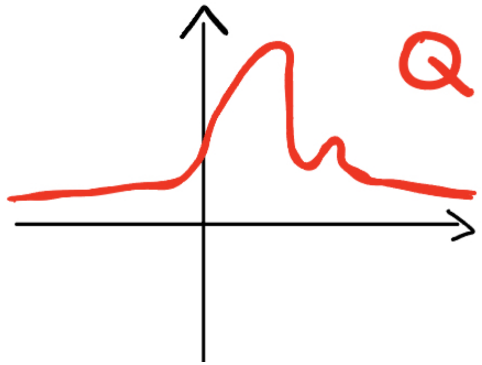
Qiang Liu QLIU@CS.DARTMOUTH.EDU
Computer Science, Dartmouth College, NH, 03755
Jason D. Lee JASONDLEE88@EECS.BERKELEY.EDU
Michael Jordan JORDAN@CS.BERKELEY.EDU
Department of Electrical Engineering and Computer Science University of California, Berkeley, CA 94709

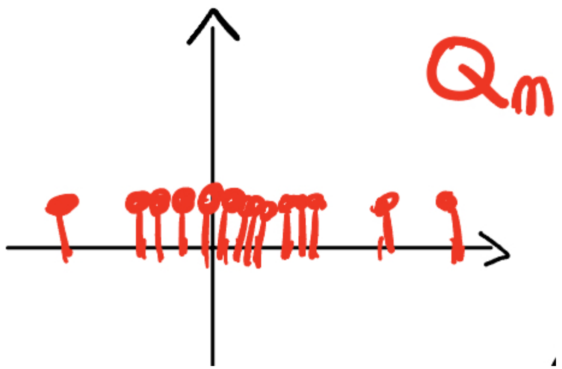
Idea: Let's use our hammer (the KSD) for goodness-of-fit testing!

Goodness-of-fit testing with KSD

$$H_0 : P = Q$$
$$H_1 : P \neq Q$$

- In practice we do not observe Q but only observe Q_n:

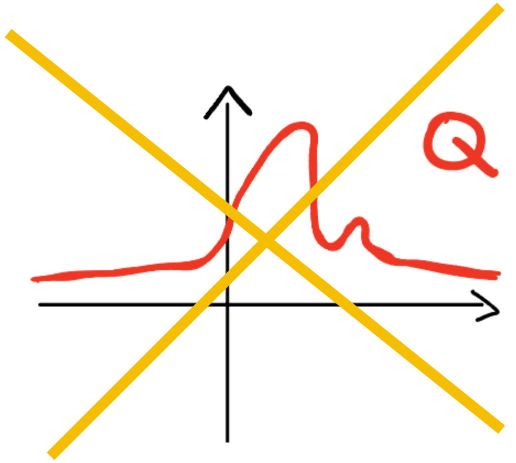


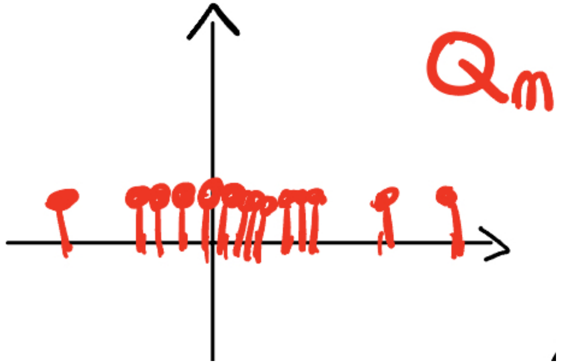


Goodness-of-fit testing with KSD

$$H_0 : P = Q$$
$$H_1 : P \neq Q$$

- In practice we do not observe Q but only observe Q_n:

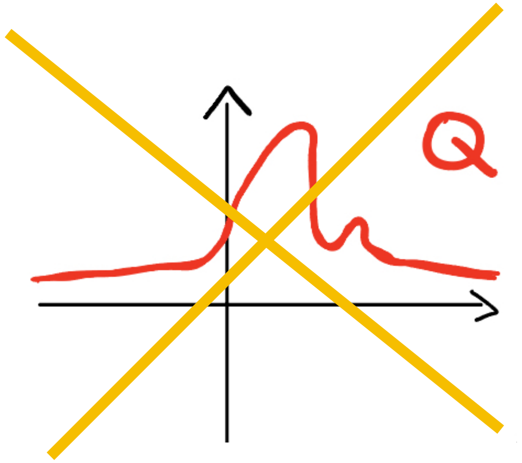


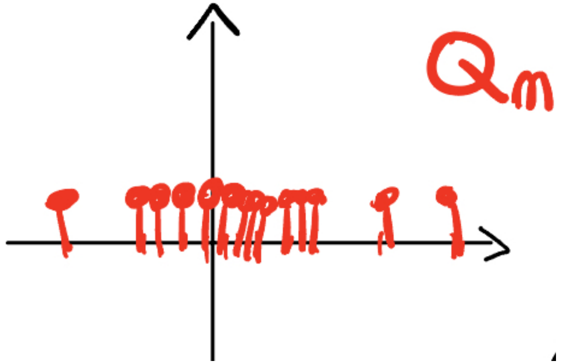


Goodness-of-fit testing with KSD

$$H_0 : P = Q$$
$$H_1 : P \neq Q$$

- In practice we do not observe Q but only observe Q_n:





- We will therefore compute KSD$(P \,||\, Q_n)$ instead of KSD$(P \,||\, Q)$, which very conveniently turns out to be exactly what we can compute!

Accounting for finite data

$$H_0 : P = Q$$
$$H_1 : P \neq Q$$

- Since we are using Q_n instead of Q, we do not have that

$$\text{KSD}(P \,||\, Q_n) \neq 0 \qquad \Rightarrow \qquad P \neq Q$$

- We must account for the fact that **we have a finite amount of data** n.

Accounting for finite data

$$H_0 : P = Q$$
$$H_1 : P \neq Q$$

- Since we are using Q_n instead of Q, we do not have that

$$\text{KSD}(P\,||\,Q_n) \neq 0 \qquad \Rightarrow \qquad P \neq Q$$

- We must account for the fact that **we have a finite amount of data** n.
- However, we would still expect that

$$\text{KSD}(P\,||\,Q_n) \approx 0 \qquad \Rightarrow \qquad P = Q$$
$$\text{KSD}(P\,||\,Q_n) \gg 0 \qquad \Rightarrow \qquad P \neq Q$$

Test statistic

$$H_0 : P = Q$$
$$H_1 : P \neq Q$$

- To construct this test, we will therefore choose:

$$\Delta = n\mathrm{KSD}(P\,||\,Q_n)^2$$

- If Δ is larger than we would expect under the null, we will reject the null hypothesis, and otherwise we will not reject.

- In practice the p-values will be computed using a Wild bootstrap algorithm which approximates the distribution of Δ under H_0:

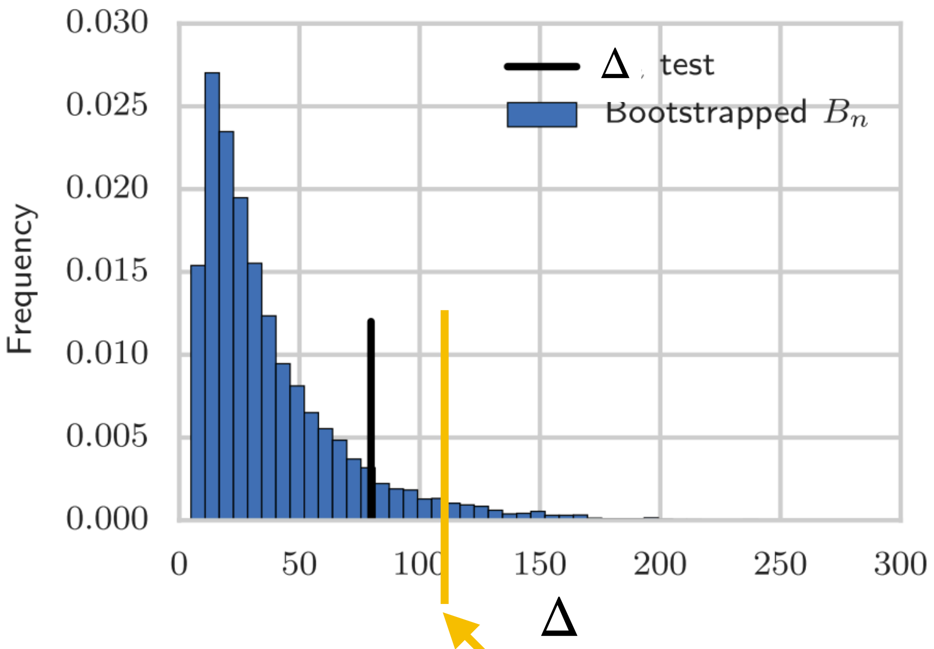
$$B = n \sum_{i,j=1}^{n} W_i W_j k_p(x_i, x_j) \qquad W_1, \ldots, W_n \sim \text{Rademacher}$$

Kernel goodness-of-fit in practice

$$H_0 : P = Q$$
$$H_1 : P \neq Q$$

- **Goodness-of-fit testing algorithm:**
  - Set level of the test to α (e.g. 0.05)
  - Calculate $\Delta = n\text{KSD}(P\,||\,Q_n)$.
  - Obtain c_α, the $(1-\alpha)$-quantile from the M bootstrap samples $B_1, \ldots, B_M$.
  - If $\Delta > c_\alpha$ then reject, otherwise do not reject.

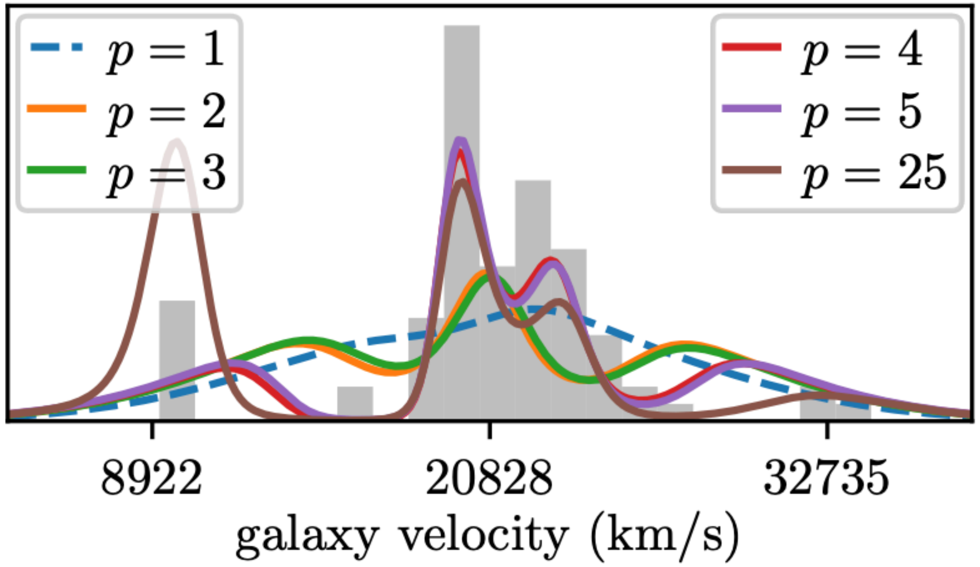
[Chwialkowski et al 2016 - slightly modified]



Quantile corresponding to our level

Composite goodness-of-fit

- Consider some parametric family of models: $\{P_\theta : \theta \in \Theta\}$



Key, O., Gretton, A., **Briol, F-X.** & Fernandez, T.. (2021). Composite goodness-of-fit tests with kernels. *arXiv:2111.10275 (under review).*

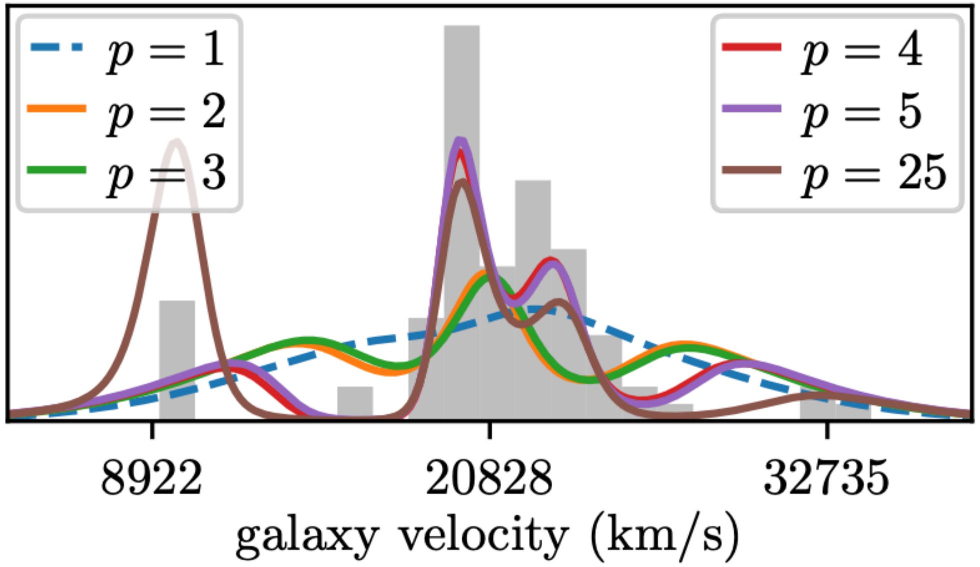
Composite goodness-of-fit

- Consider some parametric family of models: $\{P_\theta : \theta \in \Theta\}$

- An interesting question could be:

"Is my parametric model misspecified?"

$$H_0 : \exists \theta^* \text{ such that } P_{\theta^*} = Q$$
$$H_1 : \nexists \theta^* \text{ such that } P_{\theta^*} = Q$$



Key, O., Gretton, A., **Briol, F-X.** & Fernandez, T.. (2021). Composite goodness-of-fit tests with kernels. *arXiv:2111.10275 (under review).*

Overview: goodness-of-fit with Stein

- A key question in statistics is:

“Are the distributional assumptions of my analysis reasonable”

Overview: goodness-of-fit with Stein

- A key question in statistics is:

“Are the distributional assumptions of my analysis reasonable”

- Sadly classical statistical tests cannot answer this question beyond very simple distributions P such as Gaussians or uniforms.

Overview: goodness-of-fit with Stein

- A key question in statistics is:

“Are the distributional assumptions of my analysis reasonable”

- Sadly classical statistical tests cannot answer this question beyond very simple distributions P such as Gaussians or uniforms.

Stein characterisations allow us to design goodness-of-fit tests for a very wide variety of models so long as $\nabla_x \log p(x)$ is tractable!

Stein’s method as a computational tool

Parameter estimation and gen-Bayes

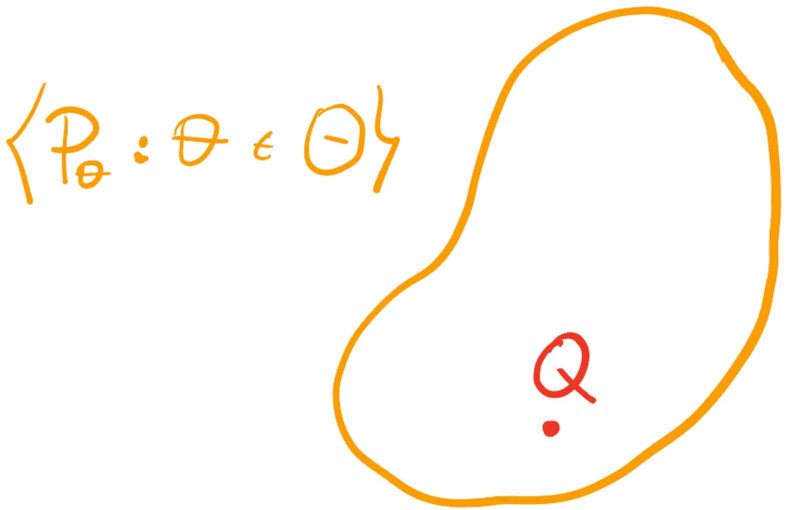
Minimum distance estimators

- In parameter estimation, we typically have a parametric family of distributions:

$$\{P_\theta : \theta \in \Theta\}$$

- Given some data $x_1, \ldots, x_n \sim Q$, we would like to find

$$\theta^* \text{ such that } P_{\theta^*} = Q$$



Why discrepancies?

- We already have plenty of good ways to estimate parameters, including maximum likelihood estimation and Bayes:

$$\arg\max_{\theta \in \Theta} \log\left(\prod_{i=1}^{n} p_\theta(x_i)\right) \qquad \pi(\theta \,|\, x_1, \ldots, x_n) \propto \prod_{i=1}^{n} p_\theta(x_i)\pi(\theta)$$

Why discrepancies?

- We already have plenty of good ways to estimate parameters, including maximum likelihood estimation and Bayes:

$$\arg\max_{\theta\in\Theta} \log\left(\prod_{i=1}^{n} p_\theta(x_i)\right) \qquad \pi(\theta\,|\,x_1,\ldots,x_n) \propto \prod_{i=1}^{n} p_\theta(x_i)\pi(\theta)$$

- These are even known to be optimal in some ways, but….

“What if the model/likelihood is misspecified?”

“What if these approaches are computationally intractable?”

Minimum distance estimators

- A natural approach is to use a minimum distance estimator:

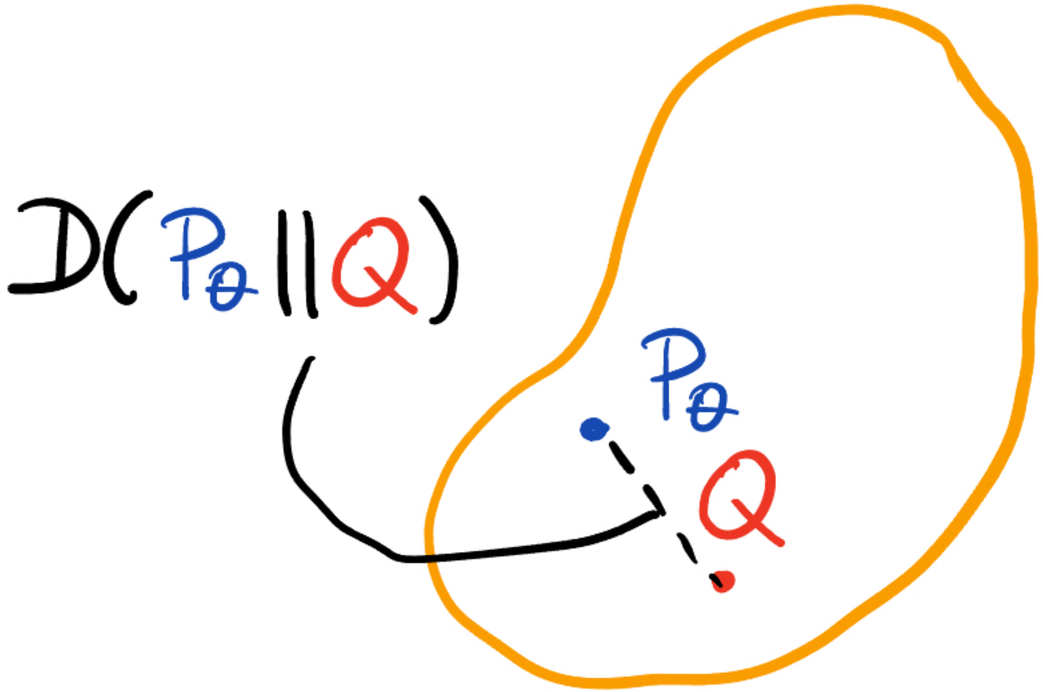
$$\arg\min_{\theta \in \Theta} D(P_\theta || Q)$$

Minimum distance estimators

- A natural approach is to use a minimum distance estimator:

$$\arg\min_{\theta \in \Theta} D(P_\theta || Q)$$

- We are simply asking for the model P_θ and the true data generating process Q to be the same, or as similar as possible.



Existing methods

- Of course, we do not have access to Q, but we have access to Q_n

$$\cancel{\arg\min_{\theta\in\Theta} D(P_\theta || Q)} \qquad \arg\min_{\theta\in\Theta} D(P_\theta || Q_n)$$

Existing methods

- Of course, we do not have access to Q, but we have access to Q_n

$$\cancel{\arg\min_{\theta\in\Theta} D(P_\theta \,||\, Q)} \qquad\qquad \arg\min_{\theta\in\Theta} D(P_\theta \,||\, Q_n)$$

- **Examples:**

D compares moments $\longrightarrow$ Method of moments

Existing methods

- Of course, we do not have access to Q, but we have access to Q_n

$$\cancel{\arg\min_{\theta \in \Theta} D(P_\theta || Q)} \qquad \arg\min_{\theta \in \Theta} D(P_\theta || Q_n)$$

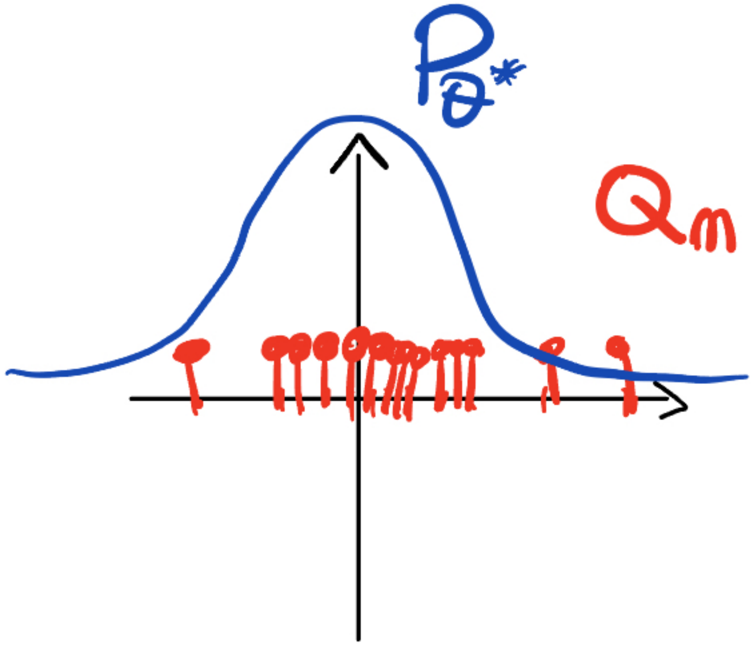
- **Examples:**

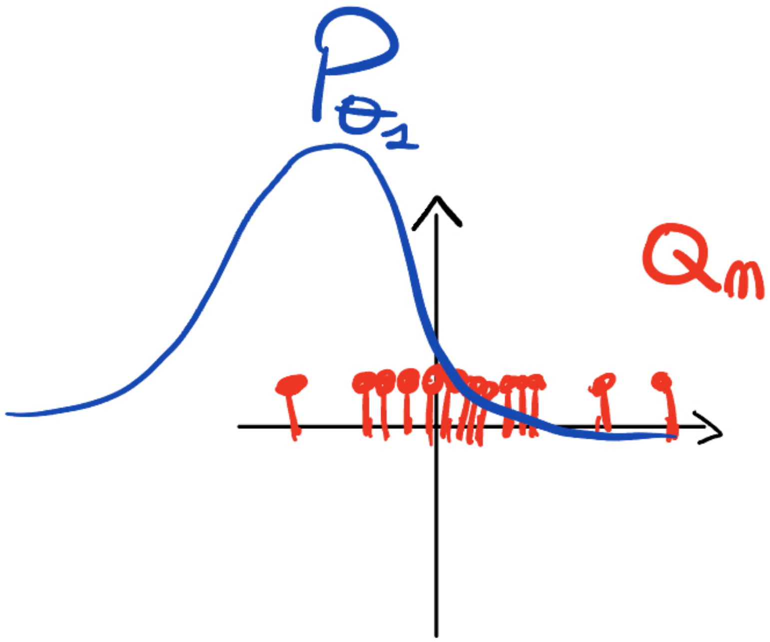
D compares moments $\longrightarrow$ Method of moments

D is KL divergence $\xrightarrow{\text{"morally"}}$ Maximum likelihood

A sketch of minimum distance estimation

$$\arg\min_{\theta \in \Theta} D(P_\theta \,||\, Q_n)$$





✓ $D(P_{\theta^*} \,||\, Q_n)$ is small

✗ $D(P_{\theta_1} \,||\, Q_n)$ is large

More on existing methods

- Many discrepancies have been used in the literature, including the Wasserstein distance, total variation distance, Beta divergences, Gamma divergences, etc…

More on existing methods

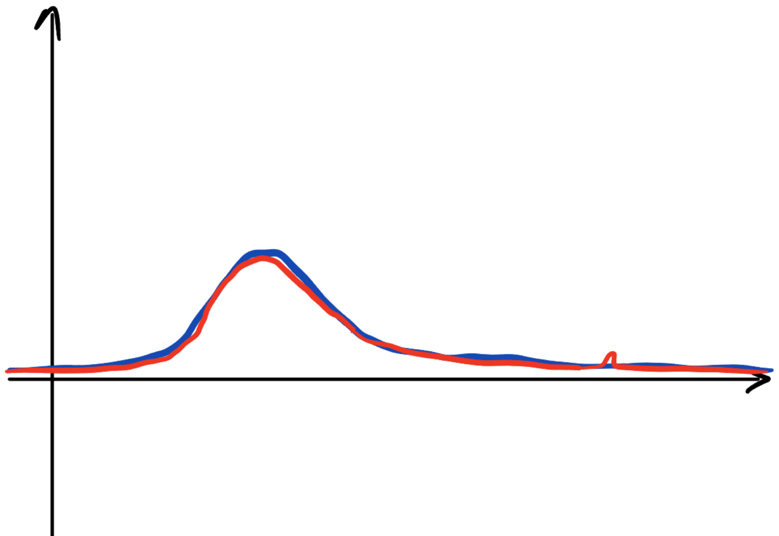
- Many discrepancies have been used in the literature, including the Wasserstein distance, total variation distance, Beta divergences, Gamma divergences, etc…
- There are typically two main questions to worry about: “*Is this discrepancy computationally tractable?*” and *“What properties does this discrepancy have?”*

More on existing methods

- Many discrepancies have been used in the literature, including the Wasserstein distance, total variation distance, Beta divergences, Gamma divergences, etc…

- There are typically two main questions to worry about: “*Is this discrepancy computationally tractable?*” and *“What properties does this discrepancy have?”*

- **Example:** are the distributions corresponding to the blue and red densities similar?

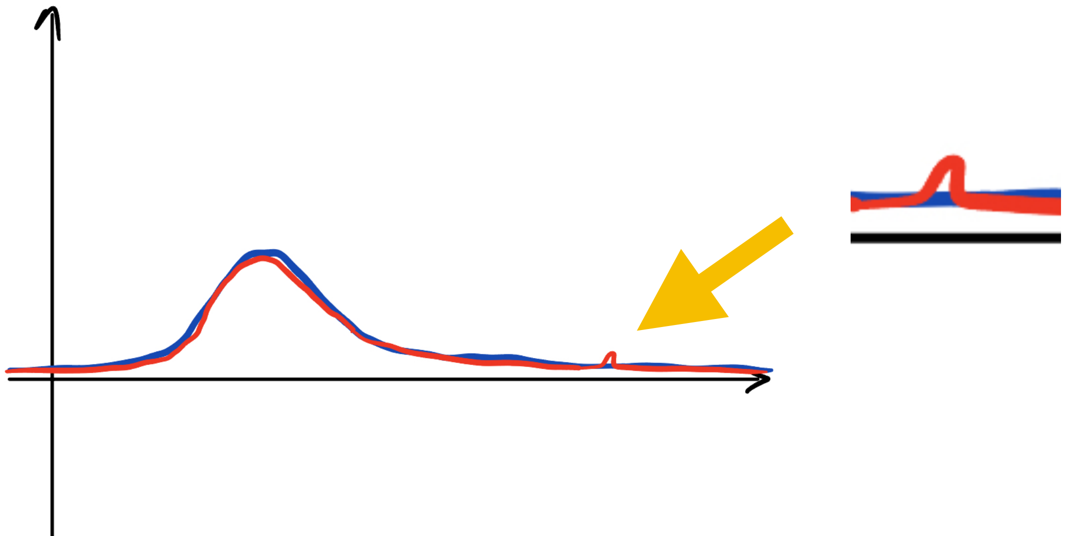


More on existing methods

- Many discrepancies have been used in the literature, including the Wasserstein distance, total variation distance, Beta divergences, Gamma divergences, etc…

- There are typically two main questions to worry about: "*Is this discrepancy computationally tractable?*" and *"What properties does this discrepancy have?"*

- **Example:** are the distributions corresponding to the blue and red densities similar?

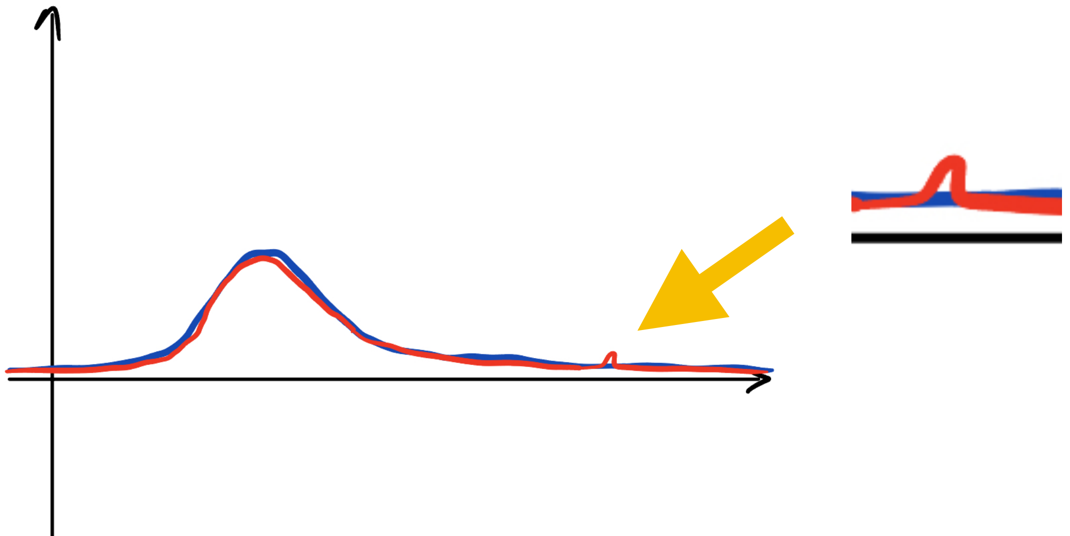


More on existing methods

- Many discrepancies have been used in the literature, including the Wasserstein distance, total variation distance, Beta divergences, Gamma divergences, etc…

- There are typically two main questions to worry about: “*Is this discrepancy computationally tractable?”* and *“What properties does this discrepancy have?”*

- **Example:** are the distributions corresponding to the blue and red densities similar?

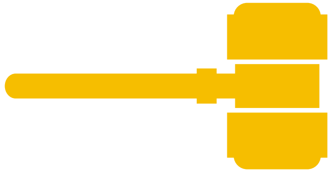


Answer: it depends on the discrepancy…

Minimum Stein discrepancy estimators

- We will come back to these properties later on. In the meantime…

- We can use our favourite hammer on this nail:



$$\arg\min_{\theta\in\Theta} \mathrm{SD}(P_\theta \,||\, Q_n)$$

Barp, A., Briol, F.-X., Duncan, A. B., Girolami, M., & Mackey, L. (2019). Minimum Stein discrepancy estimators. *NeurIPS*, 12964–12976.

Minimum Stein discrepancy estimators

- We will come back to these properties later on. In the meantime…

- We can use our favourite hammer on this nail:

$$\arg\min_{\theta\in\Theta} \mathrm{SD}(P_\theta \,||\, Q_n)$$

- **Examples:**
  - We recover score-matching with the Hyvarinen divergence.
  - For those that are old enough to know what this is, we can also recover minimum probability flow or contrastive divergence…

Barp, A., Briol, F.-X., Duncan, A. B., Girolami, M., & Mackey, L. (2019). Minimum Stein discrepancy estimators. *NeurIPS*, 12964–12976.

Generalised Bayesian Inference

- In Bayesian Inference, we typically do inference for parameters using a posterior distribution:

$$\pi(\theta \mid x_1, \ldots, x_n) \propto \prod_{i=1}^{n} p_\theta(x_i)\pi(\theta)$$

Generalised Bayesian Inference

- In Bayesian Inference, we typically do inference for parameters using a posterior distribution:

$$\underbrace{\pi(\theta \mid x_1, \ldots, x_n)}_{\text{Posterior}} \propto \underbrace{\prod_{i=1}^{n} p_\theta(x_i)}_{\text{Likelihood}} \underbrace{\pi(\theta)}_{\text{Prior}}$$

Generalised Bayesian Inference

- In Bayesian Inference, we typically do inference for parameters using a posterior distribution:

$$\underbrace{\pi(\theta \mid x_1, \ldots, x_n)}_{\text{Posterior}} \propto \prod_{i=1}^{n} \underbrace{p_\theta(x_i)}_{\text{Likelihood}} \underbrace{\pi(\theta)}_{\text{Prior}}$$

- Generalised Bayesian Inference proposes to use instead:

$$\pi(\theta \mid x_1, \ldots, x_n) \propto \exp\left(-L(\theta; x_1, \ldots, x_n)\right)\pi(\theta)$$

where $L(\theta; x_1, \ldots, x_n)$ is an empirical loss.

Generalised Bayesian Inference

- In Bayesian Inference, we typically do inference for parameters using a posterior distribution:

Posterior

$$\pi(\theta \mid x_1, \ldots, x_n) \propto \prod_{i=1}^{n} p_\theta(x_i)\pi(\theta)$$

Prior

Likelihood

- Generalised Bayesian Inference proposes to use instead:

Generalised Posterior

$$\pi(\theta \mid x_1, \ldots, x_n) \propto \exp\left(-L(\theta; x_1, \ldots, x_n)\right)\pi(\theta)$$

Prior

Loss

where $L(\theta; x_1, \ldots, x_n)$ is an empirical loss.

Generalised Bayesian Inference with Stein Discrepancies

- A natural choice of loss function is to pick a discrepancy:

$$\pi(\theta \,|\, x_1, \ldots, x_n) \propto \exp\left(-n\mathrm{SD}(P_\theta \,||\, Q_n)\right)\pi(\theta)$$

i.e. $L(\theta; x_1, \ldots, x_n) = n\mathrm{SD}(P_\theta, Q_n)$

Matsubara, T., Knoblauch, J., Briol, F.-X., & Oates, C. J. (2022). Robust generalised Bayesian inference for intractable likelihoods. *Journal of the Royal Statistical Society: Series B: (Statistical Methodology)*, *84*(3), 997–1022.

Generalised Bayesian Inference with Stein Discrepancies

- A natural choice of loss function is to pick a discrepancy:

$$\pi(\theta \mid x_1, \ldots, x_n) \propto \exp\left(-n\mathrm{SD}(P_\theta \,||\, Q_n)\right)\pi(\theta)$$

i.e. $L(\theta; x_1, \ldots, x_n) = n\mathrm{SD}(P_\theta, Q_n)$

Intuition: Our generalised posterior will have more mass in regions where $\mathrm{SD}(P_\theta \,||\, Q_n)$ is small (or equivalently where $\exp(-n\mathrm{SD}(P_\theta \,||\, Q_n))$ is large). This will typically happen close to the minimum Stein discrepancy estimator

Matsubara, T., Knoblauch, J., Briol, F.-X., & Oates, C. J. (2022). Robust generalised Bayesian inference for intractable likelihoods. *Journal of the Royal Statistical Society: Series B: (Statistical Methodology)*, *84*(3), 997–1022.

Why Stein discrepancies?

- A very reasonable question at this point is:

“Why Stein discrepancies? Why not anything else?”

Why Stein discrepancies?

- A very reasonable question at this point is:

  "Why Stein discrepancies? Why not anything else?"

- In turns out that they have two key properties:

  1. Their computational tractability makes them straightforward to apply even when dealing with somewhat complex models.
  2. The generator approach gives us a lot of flexibility in terms of which operator to use, and hence how the discrepancies encode similarity…

Weighted discrepancies

- One property we might want is "**outlier robustness**"; i.e. a small number of outliers do not impact our estimator/inference procedure.

Weighted discrepancies

- One property we might want is "**outlier robustness**"; i.e. a small number of outliers do not impact our estimator/inference procedure.

- This can be achieved by weighting our favourite discrepancies:

$$\text{DSM}(P\,||\,Q) := \mathbb{E}_{X\sim Q}\left[\|w(X)(\nabla_x \log p(X) - \nabla_x \log q(X))\|_2^2\right]$$

$$\text{DKSD}^2(P\,||\,Q) := \mathbb{E}_{X,X'\sim Q}\left[k_p(X,X')\right] \longleftarrow \quad k(x,x') = w(x)\tilde{k}(x,x')w(x')$$

Weighted discrepancies

- One property we might want is “**outlier robustness**”; i.e. a small number of outliers do not impact our estimator/inference procedure.

- This can be achieved by weighting our favourite discrepancies:

$$\text{DSM}(P\,||\,Q) := \mathbb{E}_{X\sim Q}\left[\|w(X)(\nabla_x \log p(X) - \nabla_x \log q(X))\|_2^2\right]$$

$$\text{DKSD}^2(P\,||\,Q) := \mathbb{E}_{X,X'\sim Q}\left[k_p(X,X')\right] \longleftarrow \quad k(x,x') = w(x)\tilde{k}(x,x')w(x')$$

- In particular, we can choose weights which decrease the impact of data far away from the modes of the distribution.

Robustness for KSD Bayes

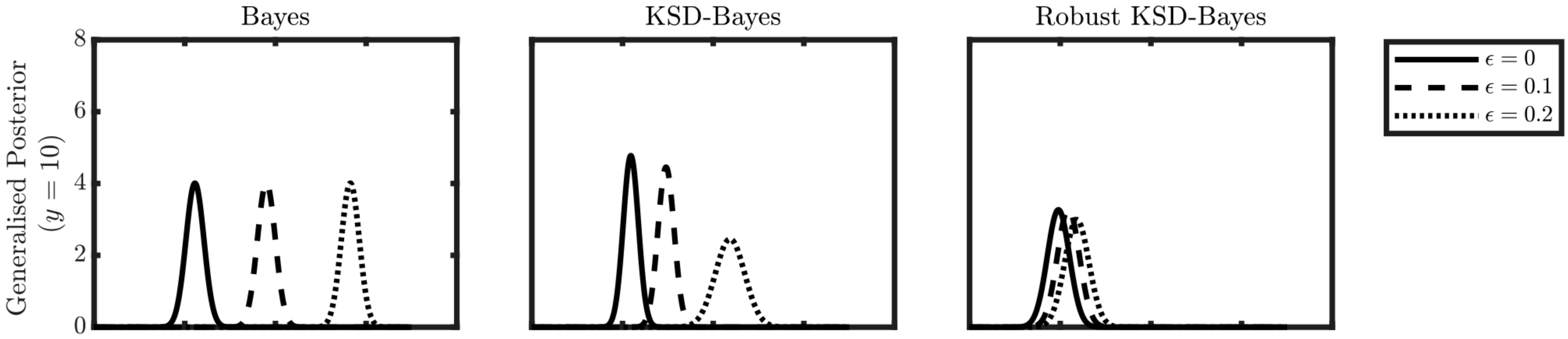
- Consider the following toy setup with a location model:

$$P_\theta = \mathcal{N}(\theta, 1) \qquad Q = (1-\epsilon)\mathcal{N}(\underbrace{\theta^*}_{=1}, 1) + \epsilon\mathcal{N}(10, 1)$$

Robustness for KSD Bayes

- Consider the following toy setup with a location model:

$$P_\theta = \mathcal{N}(\theta, 1) \qquad\qquad Q = (1-\epsilon)\mathcal{N}(\underbrace{\theta^*}_{=1}, 1) + \epsilon\mathcal{N}(10, 1)$$



$$w(x) = (1 + x^2)^{-\frac{1}{2}}$$

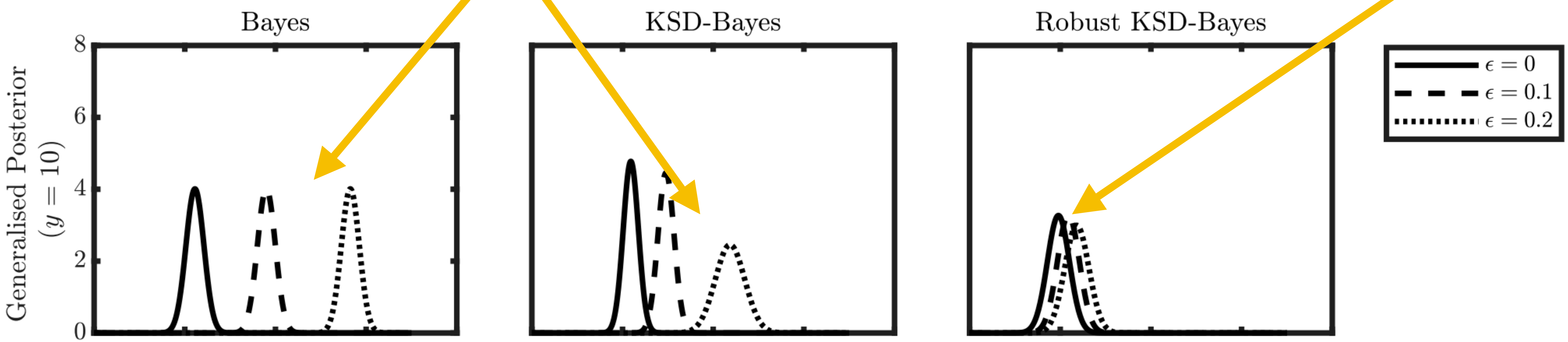
Robustness for KSD Bayes

- Consider the following toy setup with a location model:

$$P_\theta = \mathcal{N}(\theta, 1) \qquad\qquad Q = (1-\epsilon)\mathcal{N}(\underbrace{\theta^*}_{=1}, 1) + \epsilon\mathcal{N}(10, 1)$$

Not robust!

Robust!



$$w(x) = (1 + x^2)^{-\frac{1}{2}}$$

Intractable likelihoods

- The second property relates to computational tractability. We have already discussed the fact that some models have intractable likelihoods:

$$p_\theta(x) = \frac{\tilde{p}_\theta(x)}{C_\theta}$$

Intractable likelihoods

- The second property relates to computational tractability. We have already discussed the fact that some models have intractable likelihoods:

$$p_\theta(x) = \frac{\tilde{p}_\theta(x)}{C_\theta}$$

→ Stein discrepancies are ideal for estimating their parameters!

Intractable likelihoods

- The second property relates to computational tractability. We have already discussed the fact that some models have intractable likelihoods:

$$p_\theta(x) = \frac{\tilde{p}_\theta(x)}{C_\theta}$$

→ Stein discrepancies are ideal for estimating their parameters!

- The situation is much worse for Bayes, as we get **doubly intractable problems**:

$$\pi(\theta | x_1, \ldots, x_n) \propto \prod_{i=1}^n p_\theta(x_i)\pi(\theta) = \frac{1}{C}\prod_{i=1}^n \frac{\tilde{p}_\theta(x_i)}{C_\theta}\pi(\theta)$$

Intractable likelihoods

- The second property relates to computational tractability. We have already discussed the fact that some models have intractable likelihoods:

$$p_\theta(x) = \frac{\tilde{p}_\theta(x)}{C_\theta}$$

→ Stein discrepancies are ideal for estimating their parameters!

- The situation is much worse for Bayes, as we get **doubly intractable problems**:

$$\pi(\theta | x_1, \ldots, x_n) \propto \prod_{i=1}^{n} p_\theta(x_i)\pi(\theta) = \frac{1}{C}\prod_{i=1}^{n} \frac{\tilde{p}_\theta(x_i)}{C_\theta}\pi(\theta)$$

→ Stein discrepancies remove only the worst constant (i.e. C_θ but not C)!

Stein discrepancies as quadratic forms

- Assume that you have a (natural) exponential family model:

$$p_\theta(x) \propto \exp(-T(x)^\top \theta + b(\theta) + h(x))$$

for some $T : \mathbb{R}^d \to \mathbb{R}^p$, $b : \mathbb{R}^p \to \mathbb{R}$ and $h : \mathbb{R}^d \to \mathbb{R}$.

Stein discrepancies as quadratic forms

- Assume that you have a (natural) exponential family model:

$$p_\theta(x) \propto \exp(-T(x)^\top \theta + b(\theta) + h(x))$$

for some $T : \mathbb{R}^d \to \mathbb{R}^p$, $b : \mathbb{R}^p \to \mathbb{R}$ and $h : \mathbb{R}^d \to \mathbb{R}$.

- **Key result:** any squared Stein discrepancy based on a Langevin Stein operator is quadratic in θ:

$$\text{SD}^2(P_\theta || Q_n) = \theta^\top A_n \theta + b_n^\top \theta + c_n$$

Stein discrepancies as quadratic forms

- Assume that you have a (natural) exponential family model:

$$p_\theta(x) \propto \exp(-T(x)^\top \theta + b(\theta) + h(x))$$

for some $T : \mathbb{R}^d \to \mathbb{R}^p$, $b : \mathbb{R}^p \to \mathbb{R}$ and $h : \mathbb{R}^d \to \mathbb{R}$.

- **Key result:** any squared Stein discrepancy based on a Langevin Stein operator is quadratic in θ:

$$\mathrm{SD}^2(P_\theta || Q_n) = \theta^\top A_n \theta + b_n^\top \theta + c_n$$

→ This works even when we do not know the normalisation constant!

Conjugacy for generalised Bayes

- Which model has a likelihood which looks like exponential of a quadratic form?

Conjugacy for generalised Bayes

- Which model has a likelihood which looks like exponential of a quadratic form?

→ The Gaussian location model!

Conjugacy for generalised Bayes

- Which model has a likelihood which looks like exponential of a quadratic form?

→ The Gaussian location model!

- We therefore have a generalised posterior of the following form:

$$\pi(\theta \mid x_1, \ldots, x_n) \propto \exp\left(-\theta^\top A_n \theta + b_n^\top \theta + c_n\right)\pi(\theta)$$

Conjugacy for generalised Bayes

- Which model has a likelihood which looks like exponential of a quadratic form?

→ The Gaussian location model!

- We therefore have a generalised posterior of the following form:

$$\pi(\theta \mid x_1, \ldots, x_n) \propto \underbrace{\exp\left(-\theta^\top A_n \theta + b_n^\top \theta + c_n\right)}_{\text{Gaussian}} \pi(\theta)$$

Conjugacy for generalised Bayes

- Which model has a likelihood which looks like exponential of a quadratic form?

→ The Gaussian location model!

- We therefore have a generalised posterior of the following form:

$$\pi(\theta \mid x_1, \ldots, x_n) \propto \exp\left(-\theta^\top A_n \theta + b_n^\top \theta + c_n\right) \pi(\theta)$$

Gaussian

Gaussian???

Conjugacy for generalised Bayes

- Which model has a likelihood which looks like exponential of a quadratic form?

→ The Gaussian location model!

- We therefore have a generalised posterior of the following form:

$$\pi(\theta \mid x_1, \ldots, x_n) \propto \exp\left(-\theta^\top A_n \theta + b_n^\top \theta + c_n\right) \pi(\theta)$$

Gaussian!! **Gaussian** **Gaussian???**

Conjugacy for generalised Bayes

- Which model has a likelihood which looks like exponential of a quadratic form?

→ The Gaussian location model!

- We therefore have a generalised posterior of the following form:

$$\pi(\theta \,|\, x_1, \ldots, x_n) \propto \exp\left(-\theta^\top A_n \theta + b_n^\top \theta + c_n\right)\pi(\theta)$$

Gaussian!! **Gaussian** **Gaussian???**

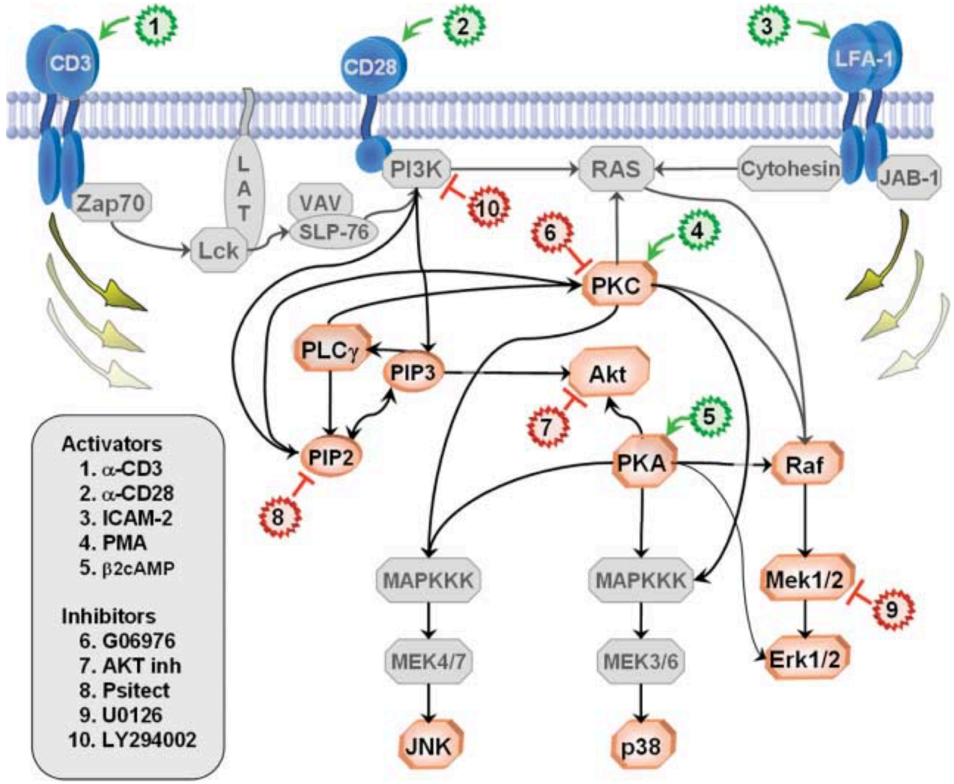
→ We get conjugacy for all natural exponential family models even when we do not know their normalisation constant!

Protein signalling networks

Parameters: $\theta_i \geq 0,\ \theta_{i,j} \geq 0$

$$p_\theta(x) \propto \exp\left(-\sum_{i=1}^{d} \theta_i \exp(x_i) - \sum_{i<j} \theta_{i,j} \exp(x_i)\exp(x_j)\right)$$

Strength of interactions
between proteins i and j

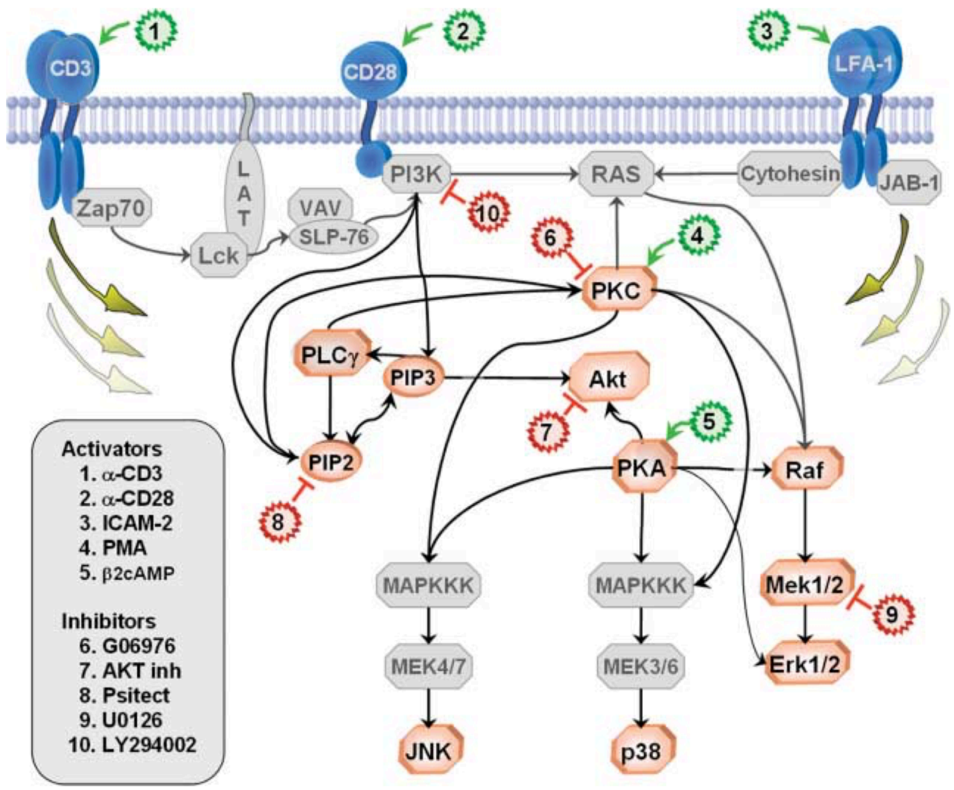


Protein signalling networks

Parameters: $\theta_i \geq 0,\ \theta_{i,j} \geq 0$

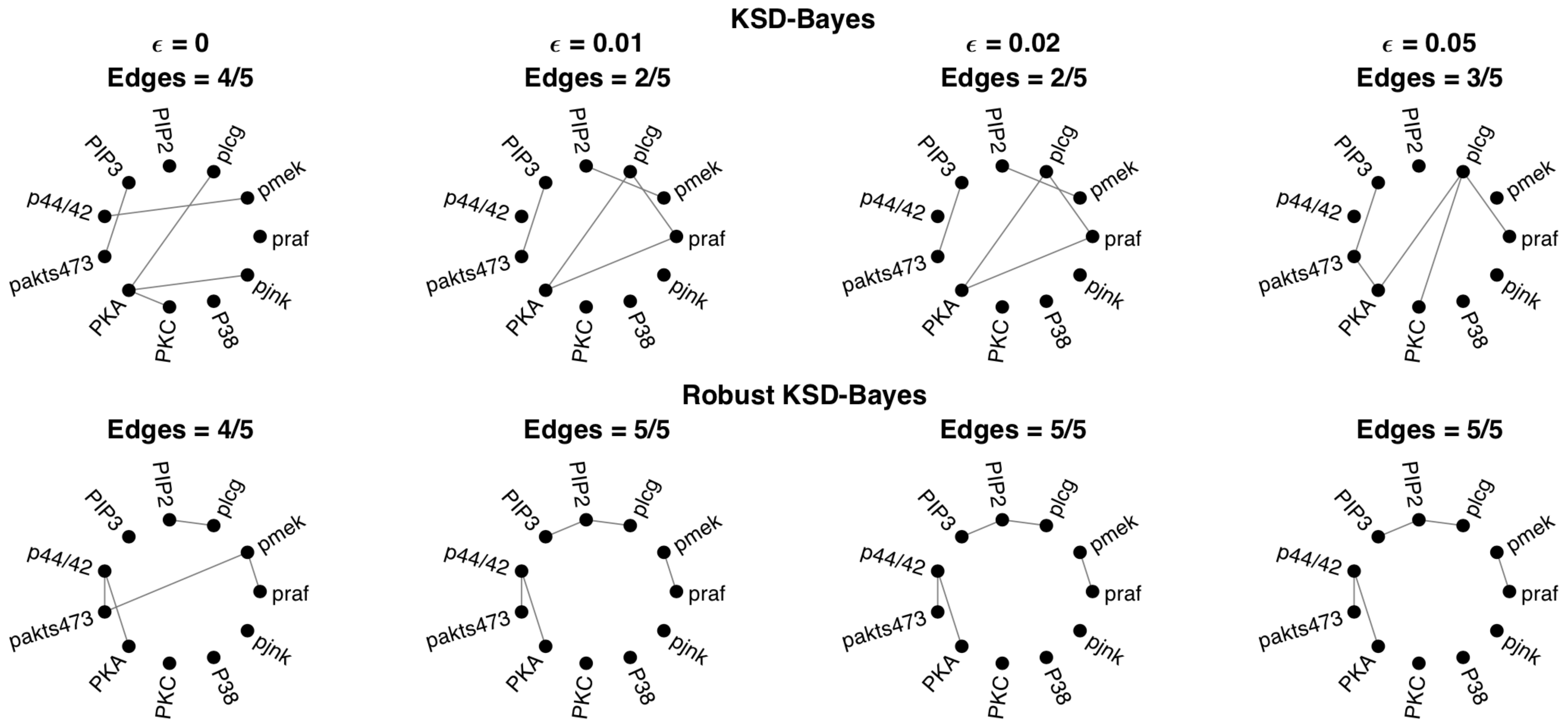
$$p_\theta(x) \propto \exp\left(-\sum_{i=1}^{d} \theta_i \exp(x_i) - \sum_{i<j} \theta_{i,j} \exp(x_i)\exp(x_j) \right)$$

Strength of interactions
between proteins i and j

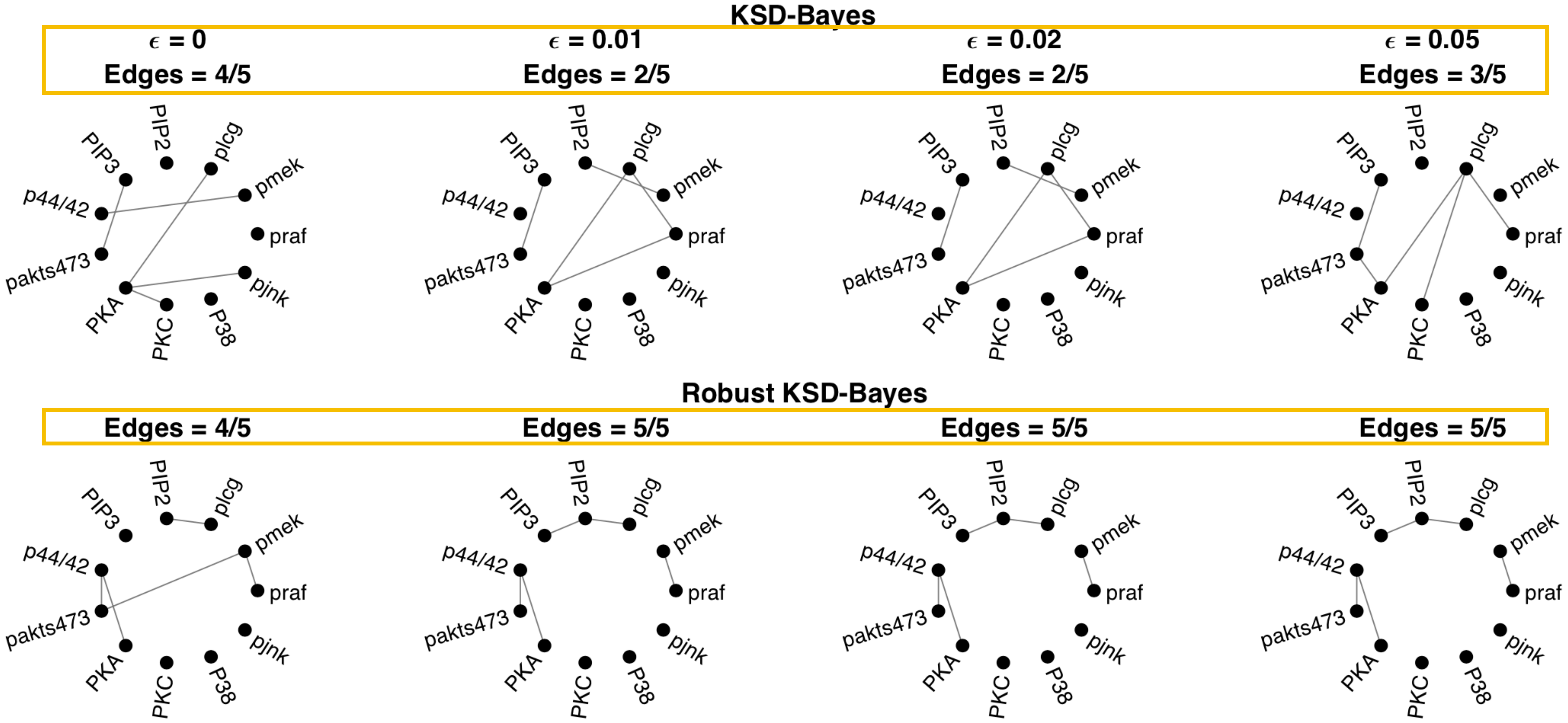


This is an exponential family, so we can have conjugate (and robust) gen-Bayes!

Protein signalling networks



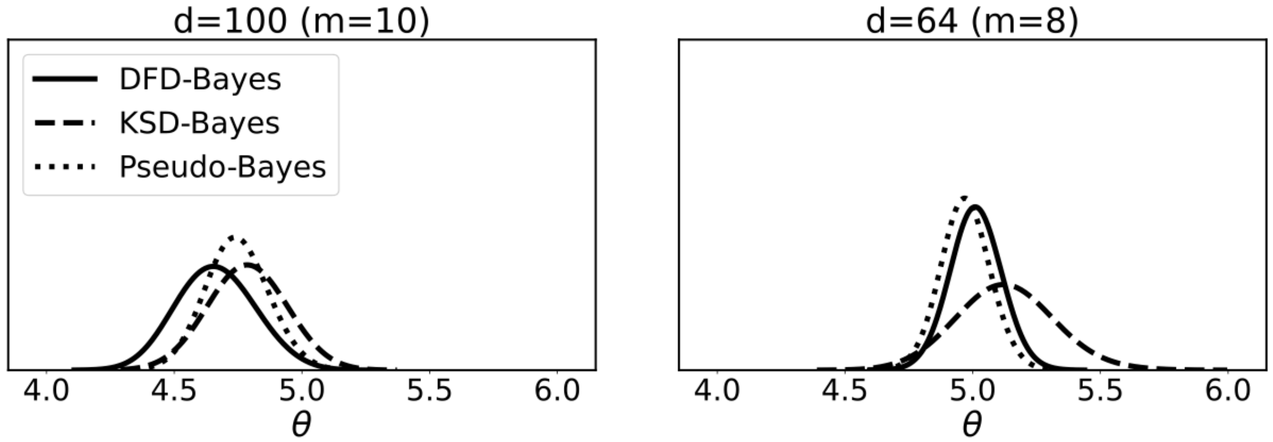
Protein signalling networks



Ising model

Data space: $\mathcal{X} = \{0,1\}^d$, $n = 1000$

$$p_\theta(x) \propto \exp\left(\frac{1}{\theta}\sum_{i=1}^{d}\sum_{j\in\mathcal{N}_i} x_i x_j\right)$$

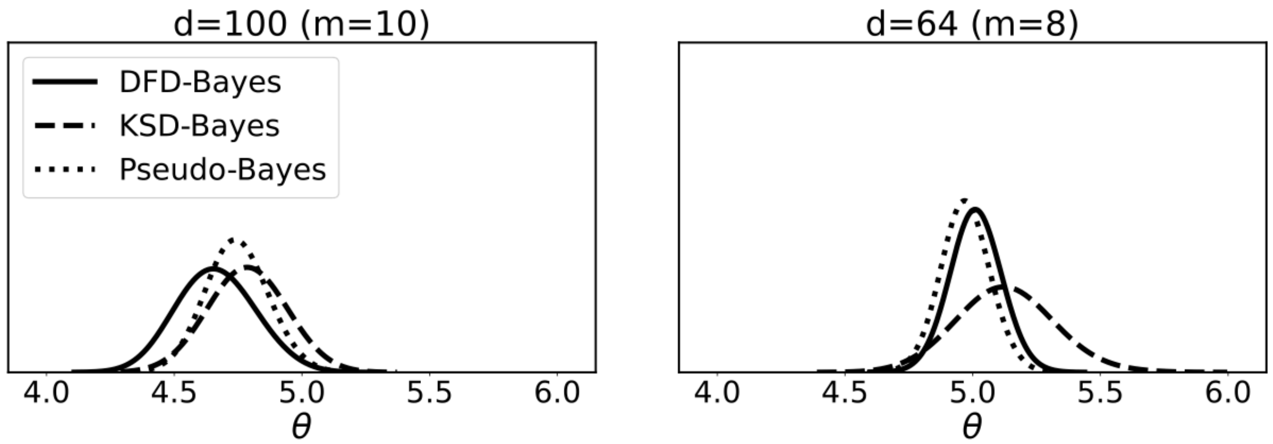


Matsubara, T., Knoblauch, J., Briol, F.-X., & Oates, C. J. (2023). Generalised Bayesian inference for discrete intractable likelihood. *Journal of the American Statistical Association, to Appear*.

Ising model

Data space: $\mathcal{X} = \{0,1\}^d$, $n = 1000$

$$p_\theta(x) \propto \exp\left(\frac{1}{\theta}\sum_{i=1}^{d}\sum_{j\in\mathcal{N}_i} x_i x_j\right)$$



<u>Computational cost:</u>

Bayes: ???

DFD-Bayes: 540s

KSD-Bayes: 2353s

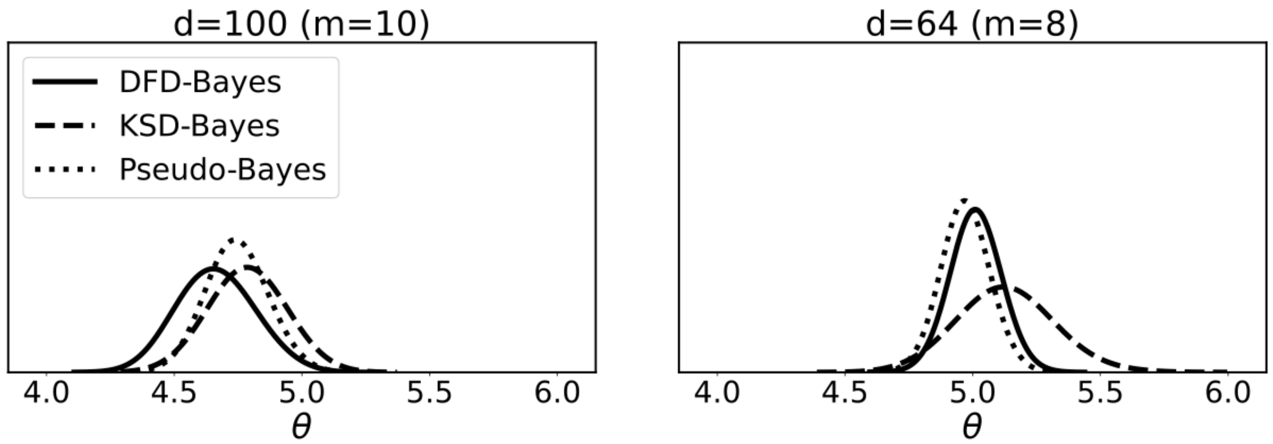
Pseudo-Bayes: 1053s

Matsubara, T., Knoblauch, J., Briol, F.-X., & Oates, C. J. (2023). Generalised Bayesian inference for discrete intractable likelihood. *Journal of the American Statistical Association, to Appear*.

Ising model

Data space: $\mathcal{X} = \{0,1\}^d$, $n = 1000$

$$p_\theta(x) \propto \exp\left(\frac{1}{\theta}\sum_{i=1}^{d}\sum_{j\in\mathcal{N}_i} x_i x_j\right)$$



Computational cost:

Bayes is not feasible here due to double intractability!

Bayes: ???

DFD-Bayes: 540s

KSD-Bayes: 2353s

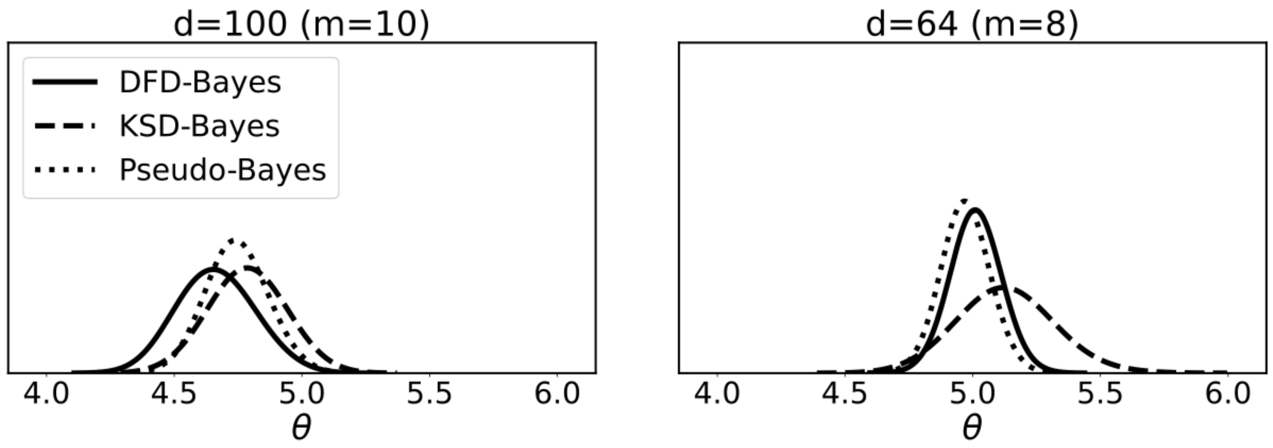
Pseudo-Bayes: 1053s

Matsubara, T., Knoblauch, J., Briol, F.-X., & Oates, C. J. (2023). Generalised Bayesian inference for discrete intractable likelihood. *Journal of the American Statistical Association, to Appear*.

Ising model

Data space: $\mathcal{X} = \{0,1\}^d, n = 1000$

$$p_\theta(x) \propto \exp\left(\frac{1}{\theta}\sum_{i=1}^{d}\sum_{j\in\mathcal{N}_i} x_i x_j\right)$$



<u>Computational cost:</u>

Bayes: ???

DFD-Bayes: 540s

KSD-Bayes: 2353s

Pseudo-Bayes: 1053s

Bayes is not feasible here due to double intractability!

Pseudo-Bayes uses the wrong model and so does not converge when we get more data…!

Matsubara, T., Knoblauch, J., Briol, F.-X., & Oates, C. J. (2023). Generalised Bayesian inference for discrete intractable likelihood. *Journal of the American Statistical Association, to Appear*.

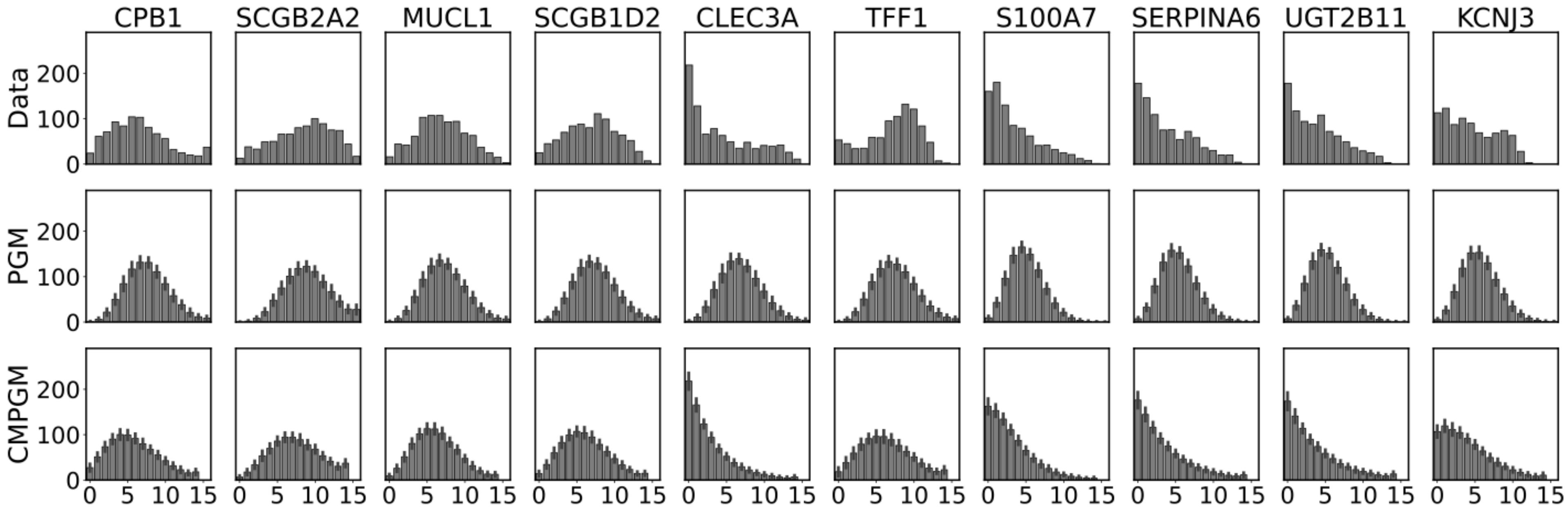
Conway-Maxwell Poisson graphical model

$$p_\theta(x) \propto \exp\left(\sum_{i=1}^{d}\theta_i x_i - \sum_{i=1}^{d}\sum_{j\in\mathcal{N}_i}\theta_{i,j}x_i x_j - \sum_{i=1}^{d}\log(x_i!)\right)$$

“Truth”

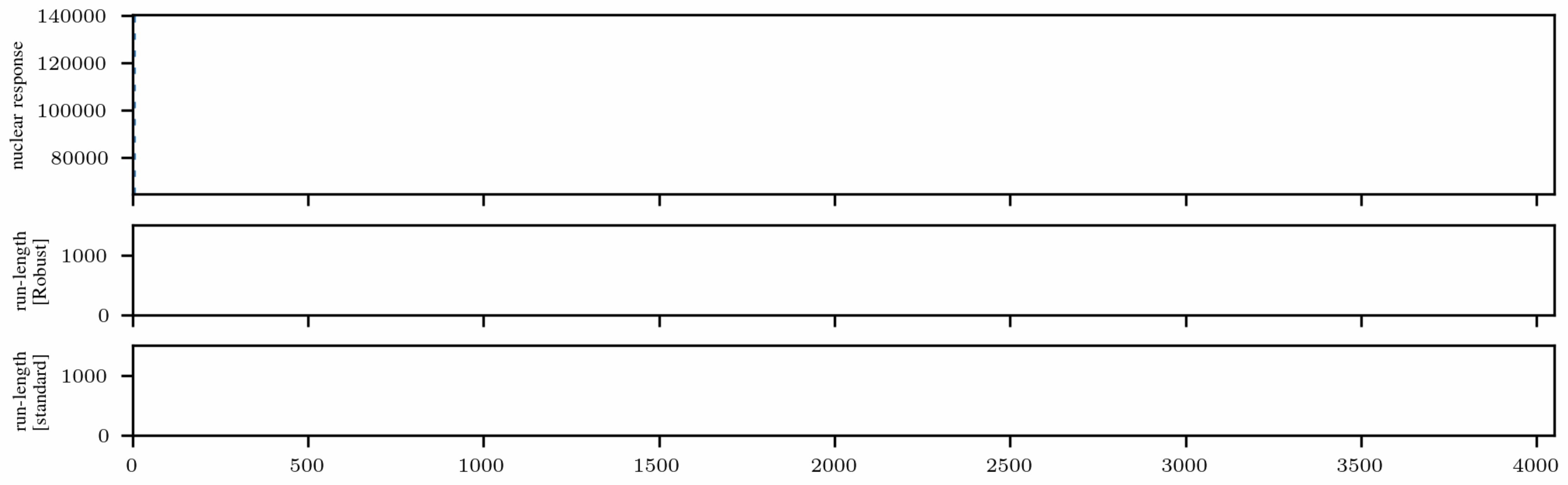
“Easy-ish”

“Very hard”



Bayesian online change-point detection

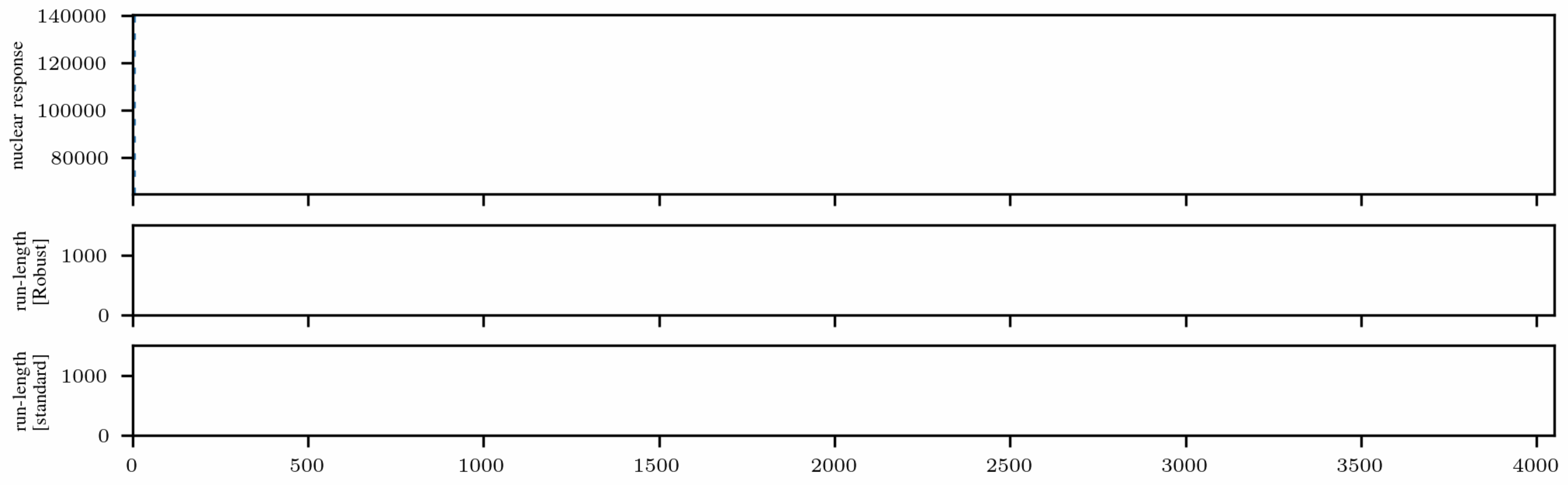
Conjugacy and robustness can be helpful for much simpler likelihoods…



Altamirano, M., Briol, F.-X., & Knoblauch, J. (2023). Robust and scalable Bayesian online changepoint detection. *ICML*, 642–663.

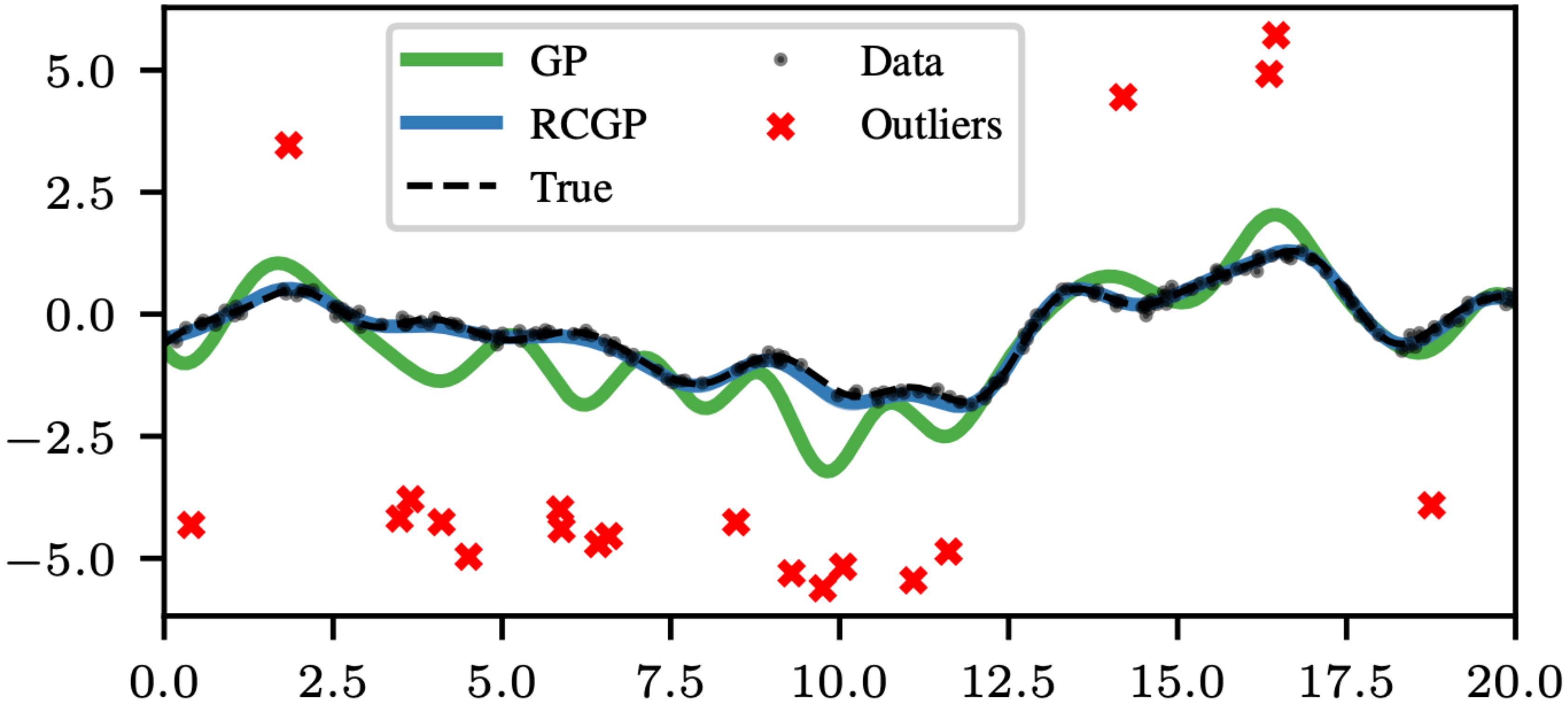
Bayesian online change-point detection

Conjugacy and robustness can be helpful for much simpler likelihoods…



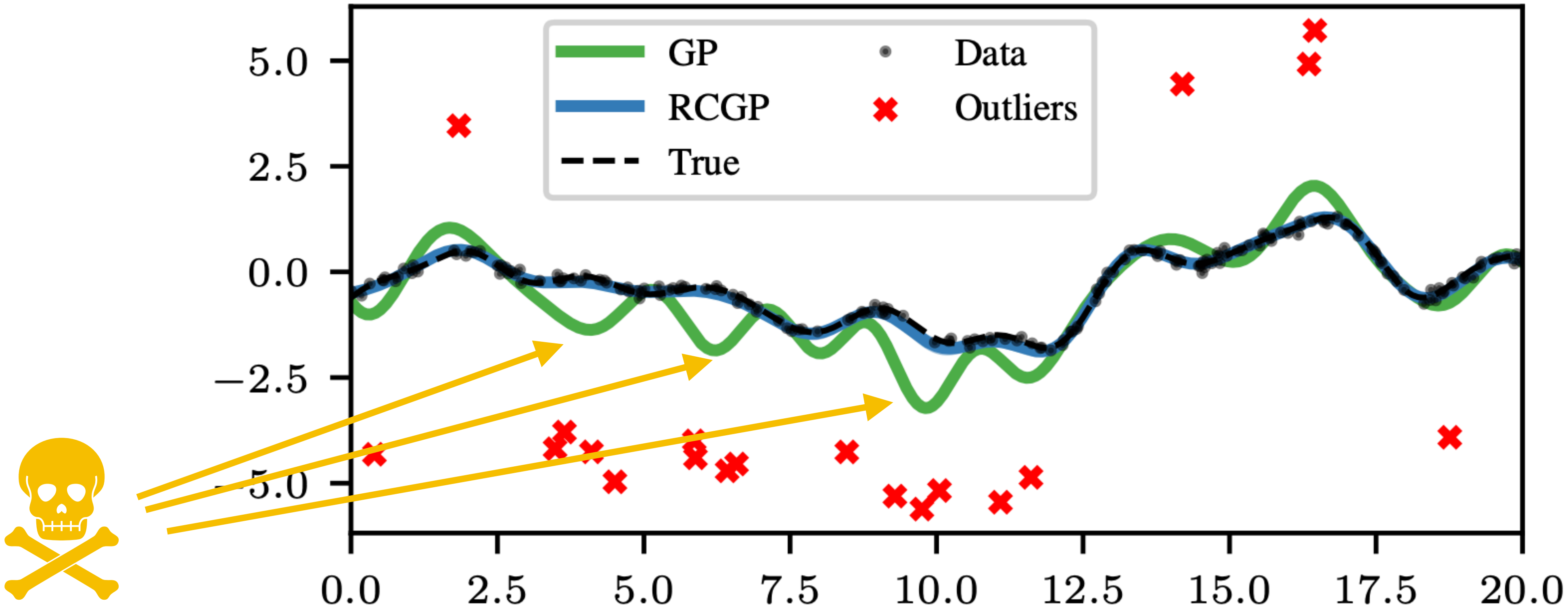
Altamirano, M., Briol, F.-X., & Knoblauch, J. (2023). Robust and scalable Bayesian online changepoint detection. *ICML*, 642–663.

Robust Gaussian process regression



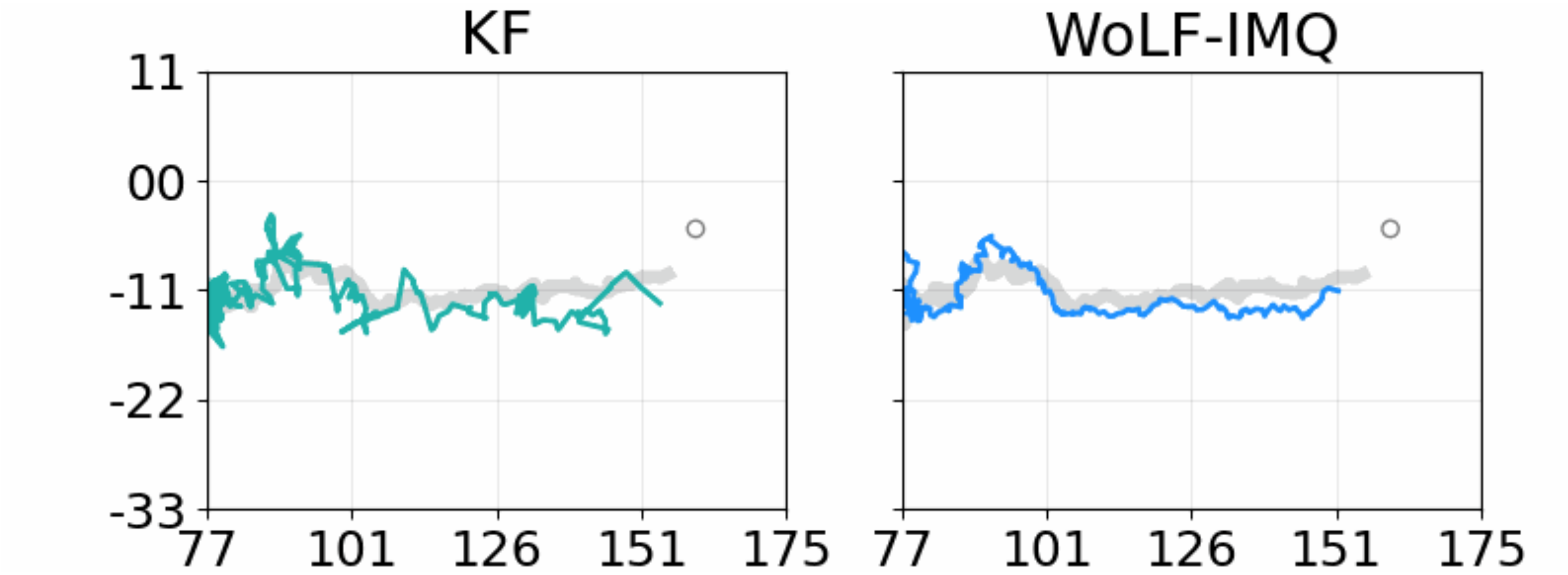
Altamirano, M., Briol, F.-X., & Knoblauch, J. (2023). Robust and conjugate Gaussian process regression. *arXiv:2311.00463*.

Robust Gaussian process regression



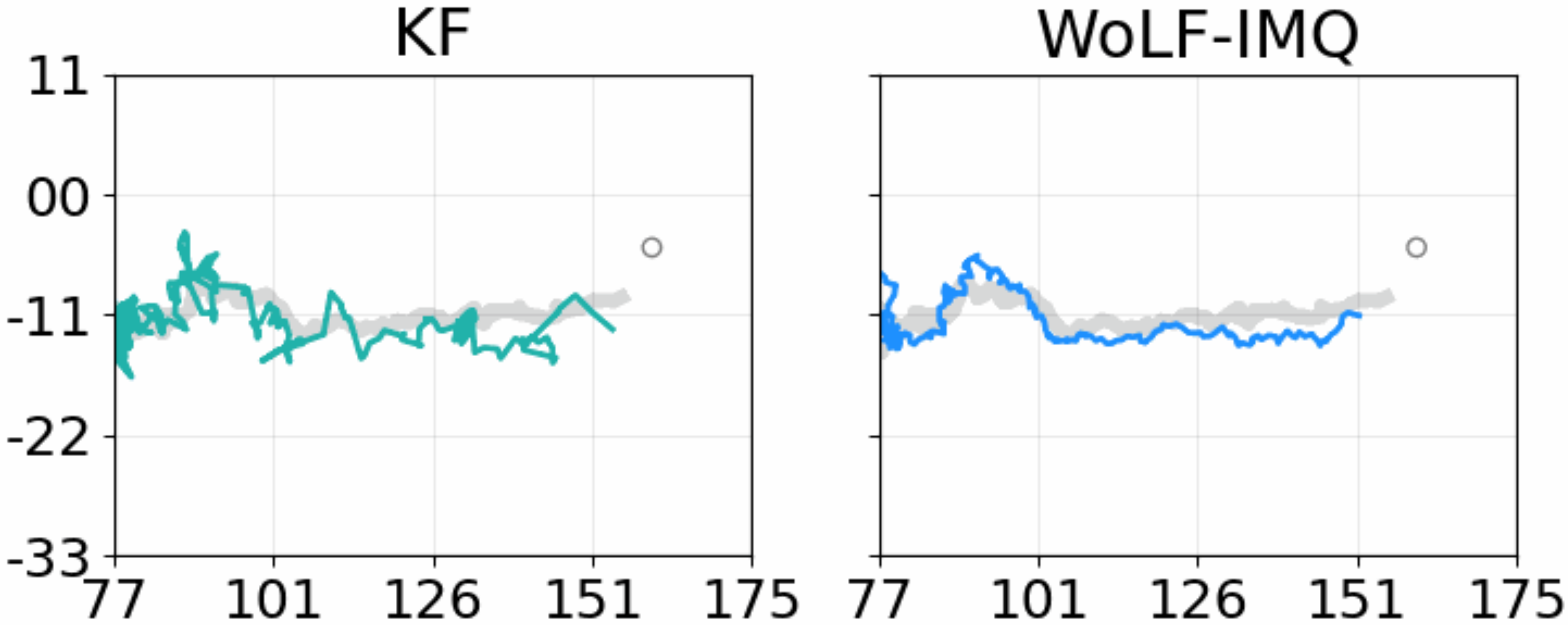
Altamirano, M., Briol, F.-X., & Knoblauch, J. (2023). Robust and conjugate Gaussian process regression. *arXiv:2311.00463*.

Not Stein but still related….



Duran-Martin, G., Altamirano, M., Shestopaloff, A. Y., Knoblauch, J., Jones, M., Briol, F.-X., & Murphy, K. (2024). *Outlier-robust Kalman filtering through generalised Bayes*. (Under review)

Not Stein but still related….



Duran-Martin, G., Altamirano, M., Shestopaloff, A. Y., Knoblauch, J., Jones, M., Briol, F.-X., & Murphy, K. (2024). *Outlier-robust Kalman filtering through generalised Bayes*. (Under review)

Overview: parameter estimation and Bayes with Stein's method

- Parameter estimation is challenging in the following two setting: (i) model misspecification, (i) complex models leading to challenging computation.

- Stein discrepancies can tackle these issues due to their computational tractability and their flexibility!

Stein’s method as a computational tool

Measuring sample quality

Computational statistics with MCMC

- Suppose we are performing Bayesian inference and end up with some posterior distribution denoted P.

- The posterior is often intractable, and needs to be approximated through sampling. One such approach consists of running a Markov chain with invariant distribution P.

Computational statistics with MCMC

- Suppose we are performing Bayesian inference and end up with some posterior distribution denoted P.

- The posterior is often intractable, and needs to be approximated through sampling. One such approach consists of running a Markov chain with invariant distribution P.

→ This is called **Markov chain Monte Carlo (MCMC)**!

Computational statistics with MCMC

- Suppose we are performing Bayesian inference and end up with some posterior distribution denoted P.

- The posterior is often intractable, and needs to be approximated through sampling. One such approach consists of running a Markov chain with invariant distribution P.

→ This is called **Markov chain Monte Carlo (MCMC)**!

- Ergodic theorems and central limit theorems can be used to **justify this approach asymptotically (i.e. as $n \to \infty$)**, but there are still many practical problems with this in practice…

Issues with MCMC

“Good MCMC”

Issues with MCMC

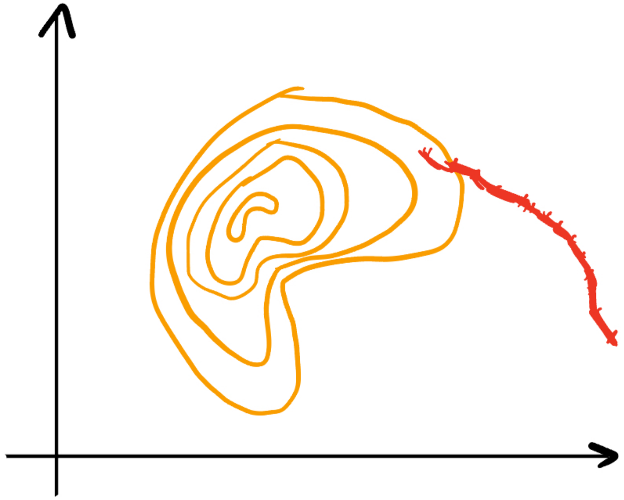
“Good MCMC”

“Slow mixing”

Issues with MCMC

“Good MCMC”

“Slow mixing”

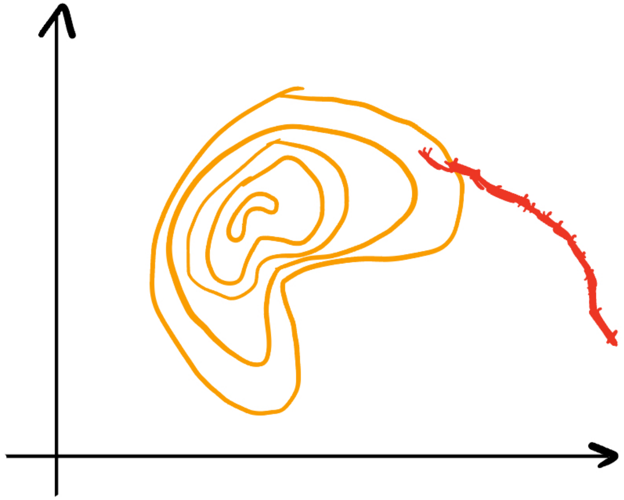


“Poor initialisation”

Issues with MCMC

"Good MCMC"

"Slow mixing"



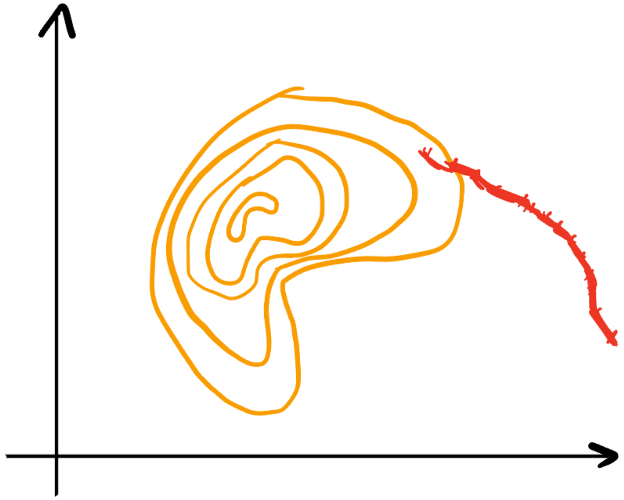
"Poor initialisation"

The main problem is that we typically only see the red trajectory and not the orange contour lines…

Issues with MCMC

"Good MCMC"

"Slow mixing"



"Poor initialisation"

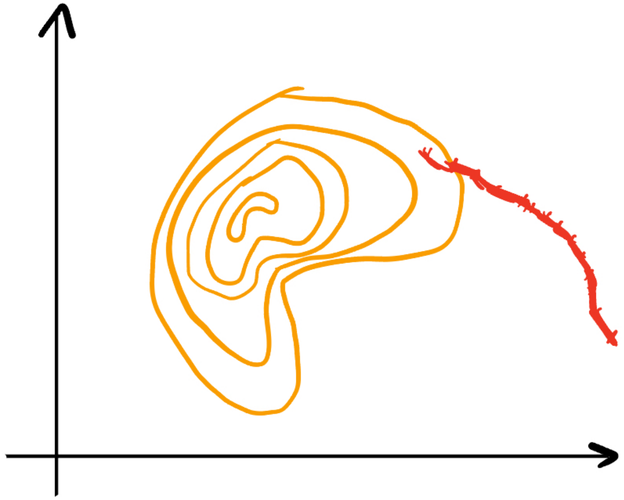
The main problem is that we typically only see the red trajectory and not the orange contour lines…

<u>Question 1:</u> Do we have a good MCMC sampler?

Issues with MCMC

"Good MCMC"

"Slow mixing"



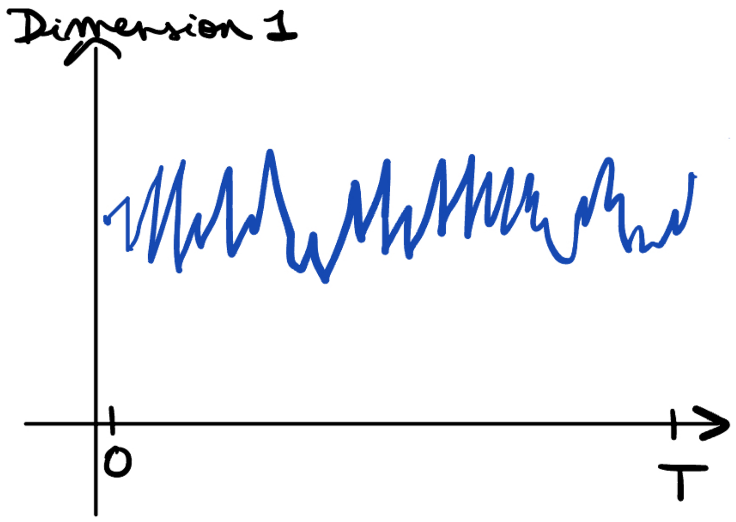
"Poor initialisation"

The main problem is that we typically only see the red trajectory and not the orange contour lines…

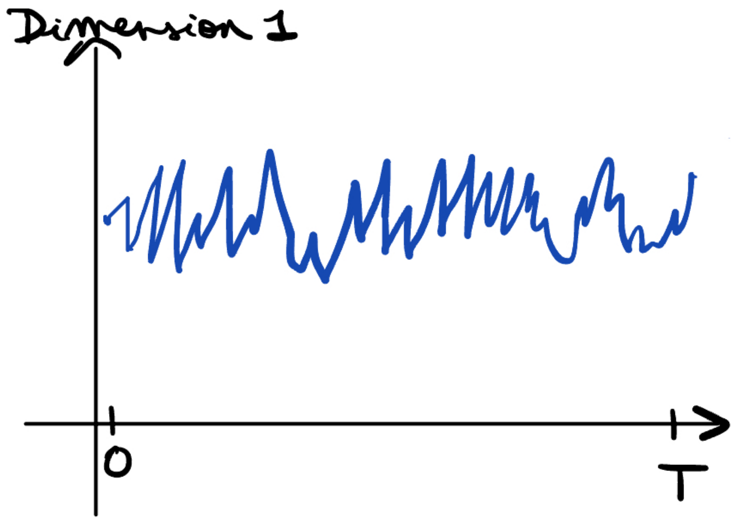
Question 1: *Do we have a good MCMC sampler?*

Question 2: *Have we run the MCMC sampler for long enough?*

Trace-plots for MCMC

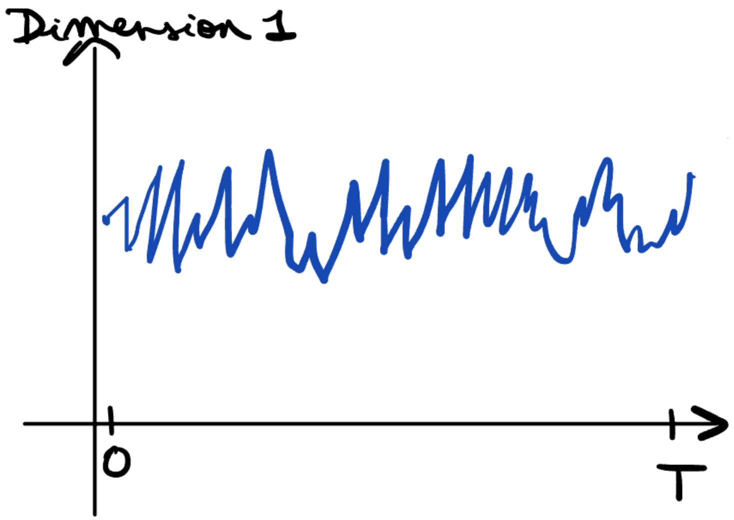


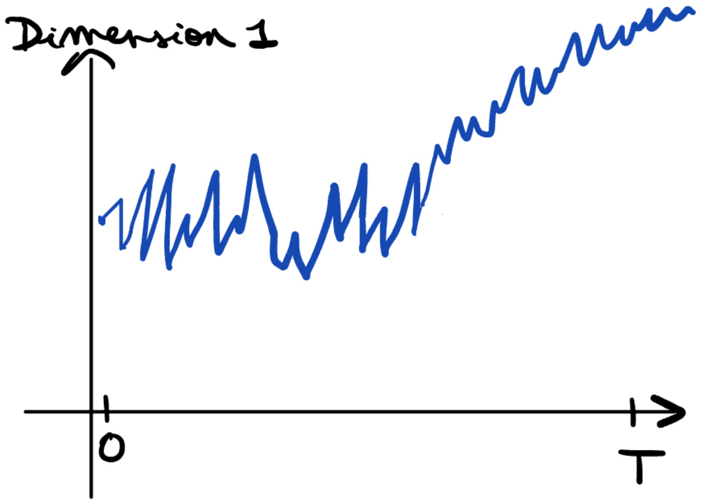
Trace-plots for MCMC



Visually seems to be mixing…
Now let me look at the other dimensions…

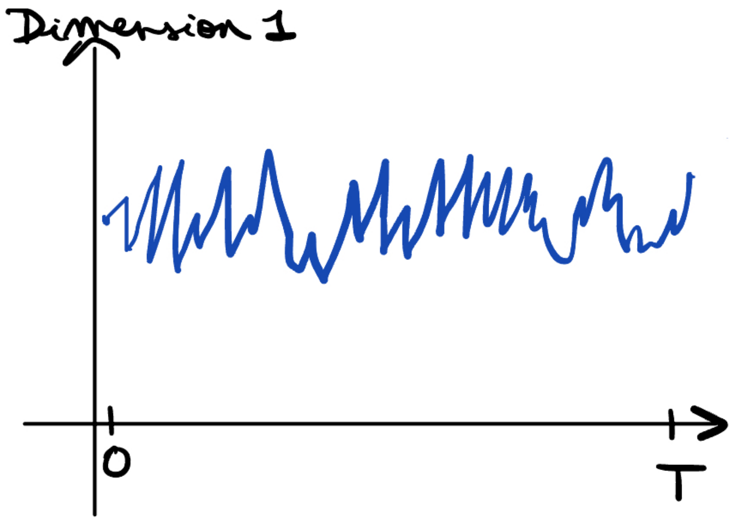
Trace-plots for MCMC

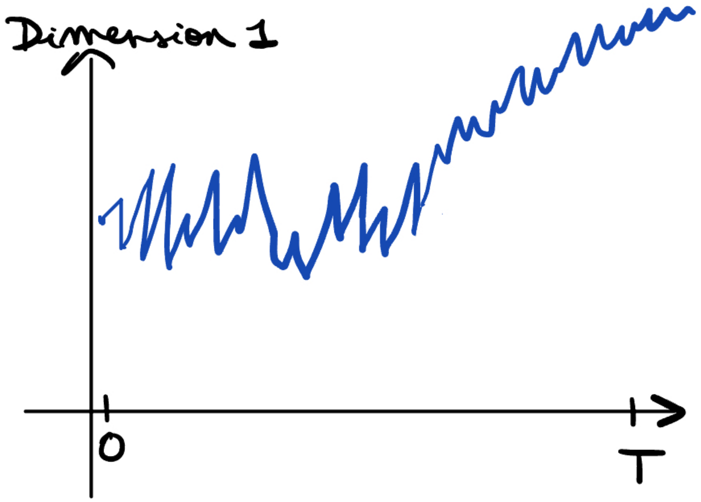




Visually seems to be mixing…
Now let me look at the other dimensions…

Trace-plots for MCMC

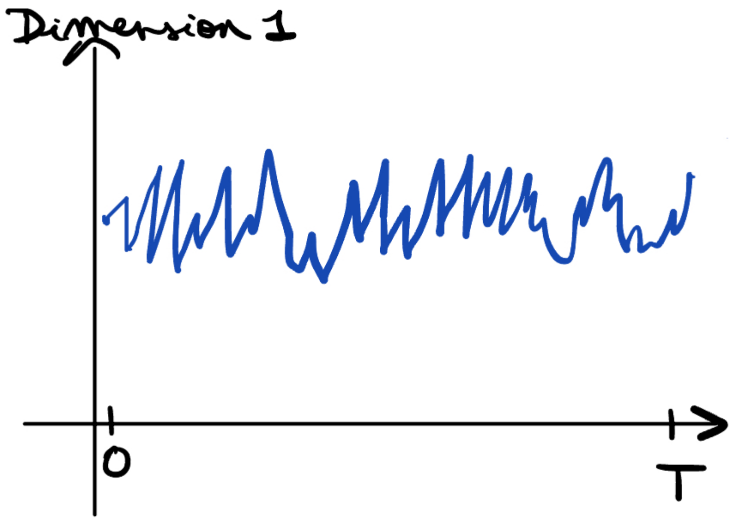


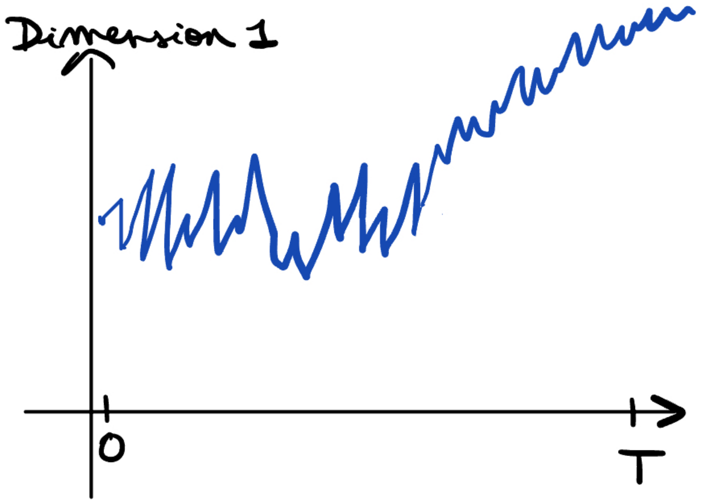


Visually seems to be mixing…
Now let me look at the other dimensions…

Uh oh… hasn't mixed so well…

Trace-plots for MCMC





Visually seems to be mixing…
Now let me look at the other dimensions…

Uh oh… hasn't mixed so well…

This is really **not** a **scalable/rigorous** approach….

Other diagnostics for MCMC

- Another approach is to track the **effective sample size**:

$$\text{ESS} = \frac{n}{1 + \sum_{k=1}^{\infty} \rho_k}$$

Other diagnostics for MCMC

- Another approach is to track the **effective sample size**:

$$\text{ESS} = \frac{n}{1 + \sum_{k=1}^{\infty} \rho_k}$$

Number of MCMC samples

Autocorrelation at lag k

Other diagnostics for MCMC

- Another approach is to track the **effective sample size**:

$$\text{ESS} = \frac{n}{1 + \sum_{k=1}^{\infty} \rho_k}$$

Number of MCMC samples

Autocorrelation at lag k

- **Potential issues:**
  - This is not always very reliable as a way of estimating how good our samples are as we need to estimate the autocorrelation.

Other diagnostics for MCMC

- Another approach is to track the **effective sample size**:

$$\text{ESS} = \frac{n}{1 + \sum_{k=1}^{\infty} \rho_k}$$

Number of MCMC samples

Autocorrelation at lag k

- **Potential issues:**
  - This is not always very reliable as a way of estimating how good our samples are as we need to estimate the autocorrelation.
  - It is also limited to MCMC, but as we will see shortly there are many other approaches for approximating a target with a point set!

Other diagnostics for MCMC

- Another approach is to track the **effective sample size**:

$$\text{ESS} = \frac{n}{1 + \sum_{k=1}^{\infty} \rho_k}$$

Number of MCMC samples

Autocorrelation at lag k

- **Potential issues:**
  - This is not always very reliable as a way of estimating how good our samples are as we need to estimate the autocorrelation.
  - It is also limited to MCMC, but as we will see shortly there are many other approaches for approximating a target with a point set!
  - Is not valid for stochastic gradient MCMC or any other approximate MCMC methods where we do not necessarily target the right P

Measuring sample quality

- A natural approach would be to look at some discrepancy:

$$\mathrm{D}(P \,||\, Q_n)$$

Target distribution

Particle approximation: $Q_n = \frac{1}{n}\sum_{i=1}^{n} \delta_{x_i}$

Measuring sample quality

- A natural approach would be to look at some discrepancy:

$$\mathrm{D}(P\,||\,Q_n)$$

Target distribution

Particle approximation: $Q_n = \frac{1}{n}\sum_{i=1}^{n} \delta_{x_i}$

- This is indeed what is done to study convergence of MCMC at a theoretical level, in which case the discrepancy is the total variation distance. (You may have heard of concepts such as geometric ergodicity?)

Measuring sample quality

- A natural approach would be to look at some discrepancy:

$$\mathrm{D}(P \,||\, Q_n)$$

Target distribution

Particle approximation: $Q_n = \frac{1}{n}\sum_{i=1}^{n} \delta_{x_i}$

- This is indeed what is done to study convergence of MCMC at a theoretical level, in which case the discrepancy is the total variation distance. (You may have heard of concepts such as geometric ergodicity?)

✗ This is completely useless as a practical tool since we cannot compute it!

Measuring sample quality with SDs

$$\mathrm{SD}(P \,||\, Q_n) \to 0 \text{ ??}$$

Gorham, J., & Mackey, L. (2015). Measuring sample quality with Stein's method. *NeurIPS*, 226–234.

Gorham, J., & Mackey, L. (2017). Measuring sample quality with kernels. *ICML*, 1292–1301.

Gorham, J., Duncan, A., Mackey, L., & Vollmer, S. (2019). Measuring sample quality with diffusions. *Annals of Applied Probability*, *29*(5), 2884–2928.

Measuring sample quality with SDs

$$\text{SD}(P\,||\,Q_n) \to 0 \text{ ??}$$

- The graph Stein discrepancy and the KSD have been proposed for this task since they are both **computable!**

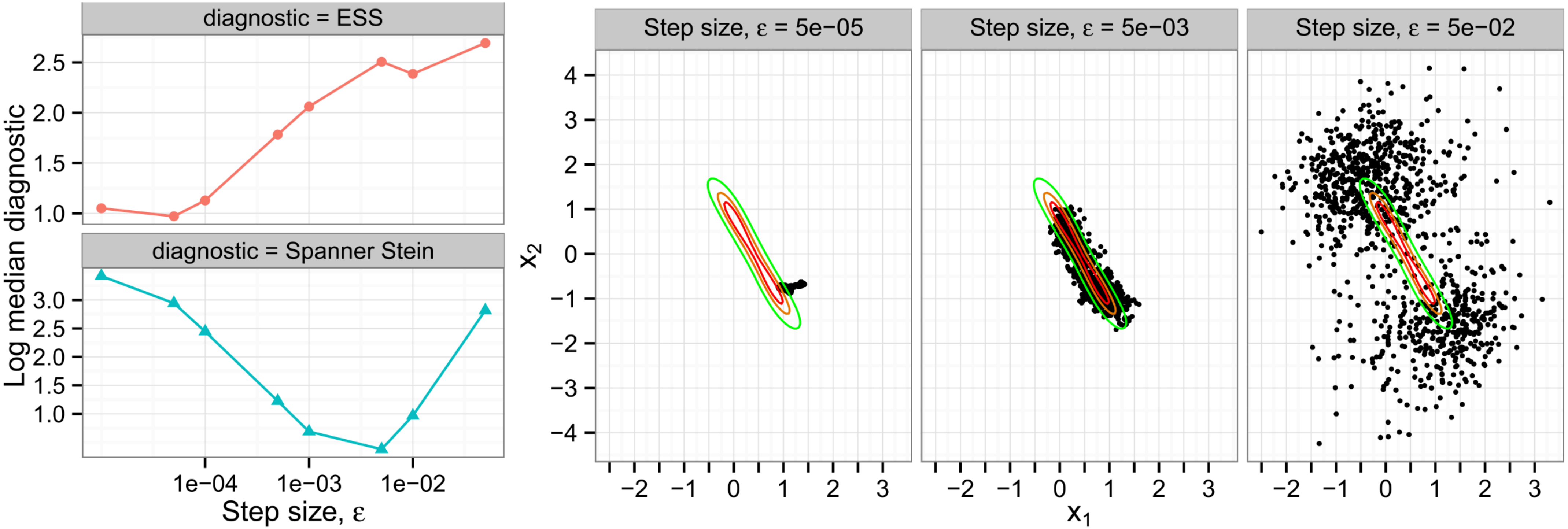
- The former essentially always controls weak convergence, whilst the latter does so under certain conditions of the kernel.

Gorham, J., & Mackey, L. (2015). Measuring sample quality with Stein's method. *NeurIPS*, 226–234.

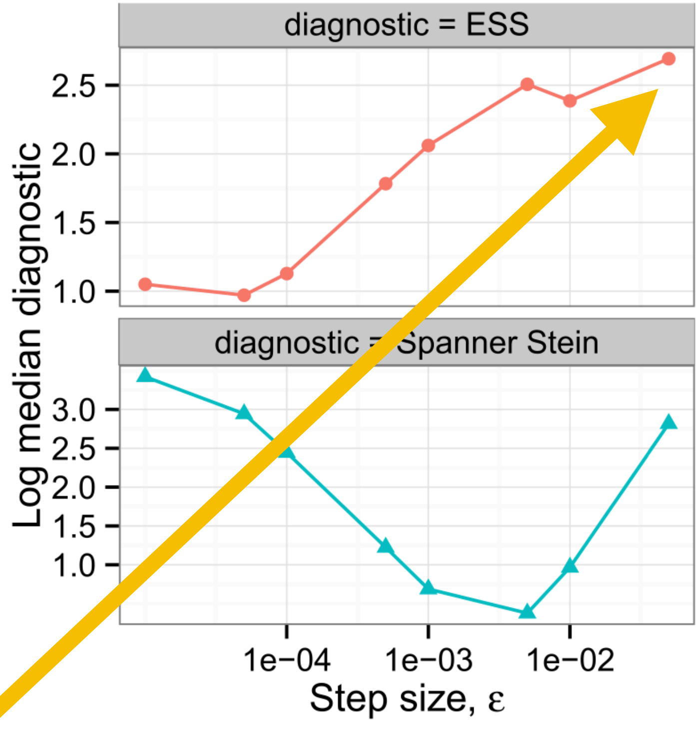
Gorham, J., & Mackey, L. (2017). Measuring sample quality with kernels. *ICML*, 1292–1301.

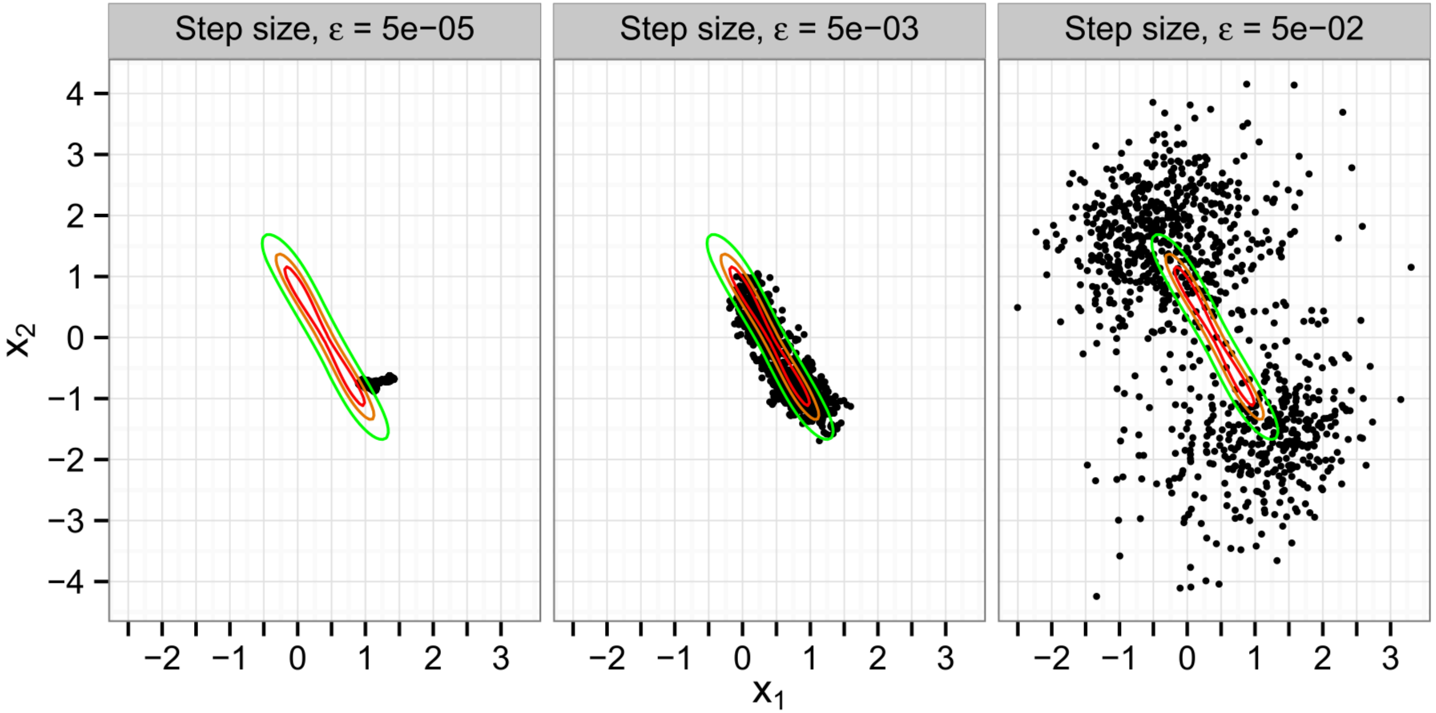
Gorham, J., Duncan, A., Mackey, L., & Vollmer, S. (2019). Measuring sample quality with diffusions. *Annals of Applied Probability*, *29*(5), 2884–2928.

Example: Stochastic gradient langevin dynamics



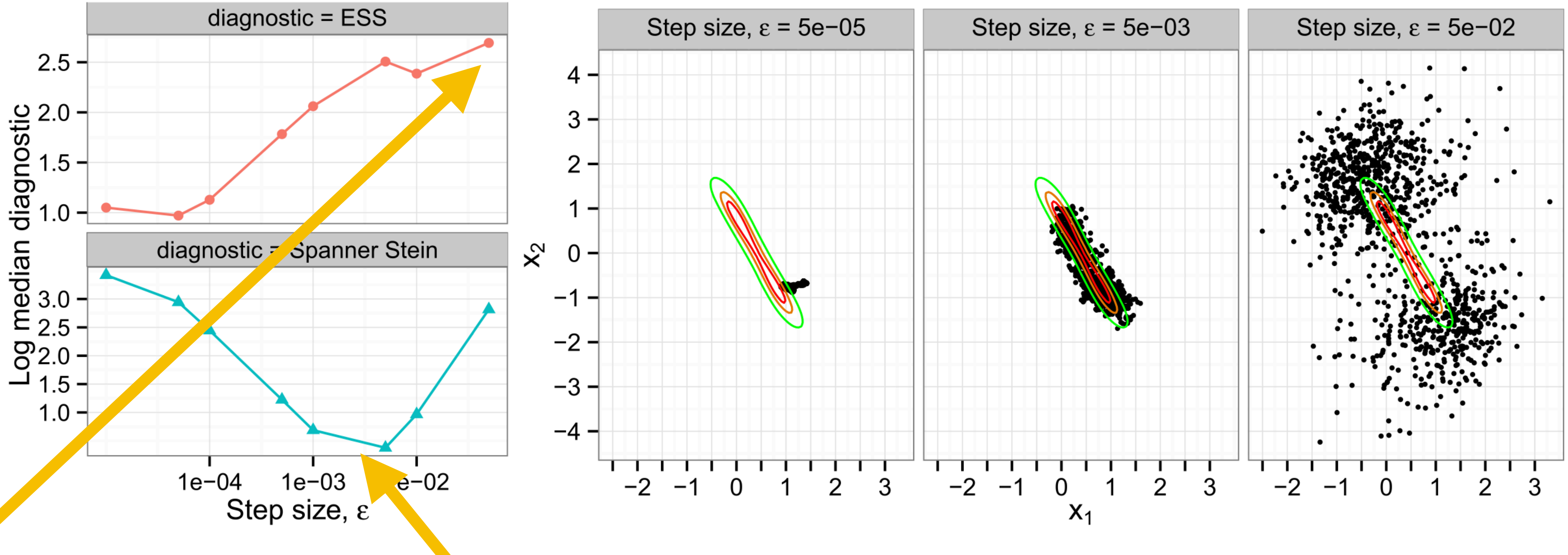
Example: Stochastic gradient langevin dynamics





Identifies sampler which jumps around too much…

Example: Stochastic gradient langevin dynamics



Identifies sampler which jumps around too much…

Correctly identifies good sampler!

Overview: measuring sample quality with Stein's method

- Measuring the quality of a point set approximation of a target P distribution is really hard!

Overview: measuring sample quality with Stein's method

- Measuring the quality of a point set approximation of a target P distribution is really hard!

- A natural approach is to use a Stein discrepancy between that point set and the target:

$$\mathrm{SD}(P\,||\,Q_n)$$

Overview: measuring sample quality with Stein's method

- Measuring the quality of a point set approximation of a target P distribution is really hard!

- A natural approach is to use a Stein discrepancy between that point set and the target:

$$\mathrm{SD}(P\,||\,Q_n)$$

- This allows us to answer concretely many questions that were previously completely intractable from a computational viewpoint…!

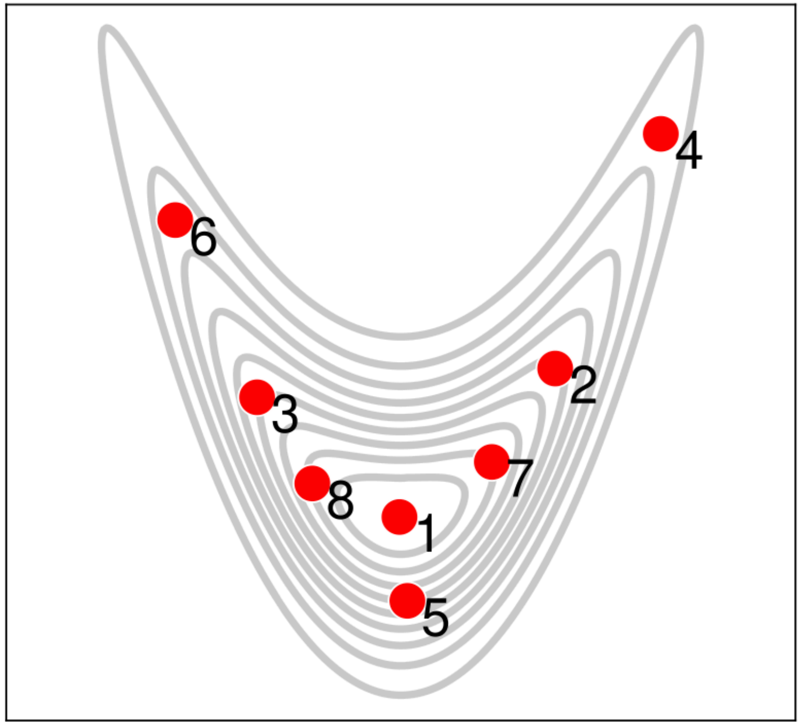
Stein’s method as a computational tool

Deterministic approximations of probability distributions

Deterministic approximations

- Suppose we have a target distribution P.
- We would like a very good approximation of the form:

$$P \approx \frac{1}{n}\sum_{i=1}^{n} \delta_{x_i}$$



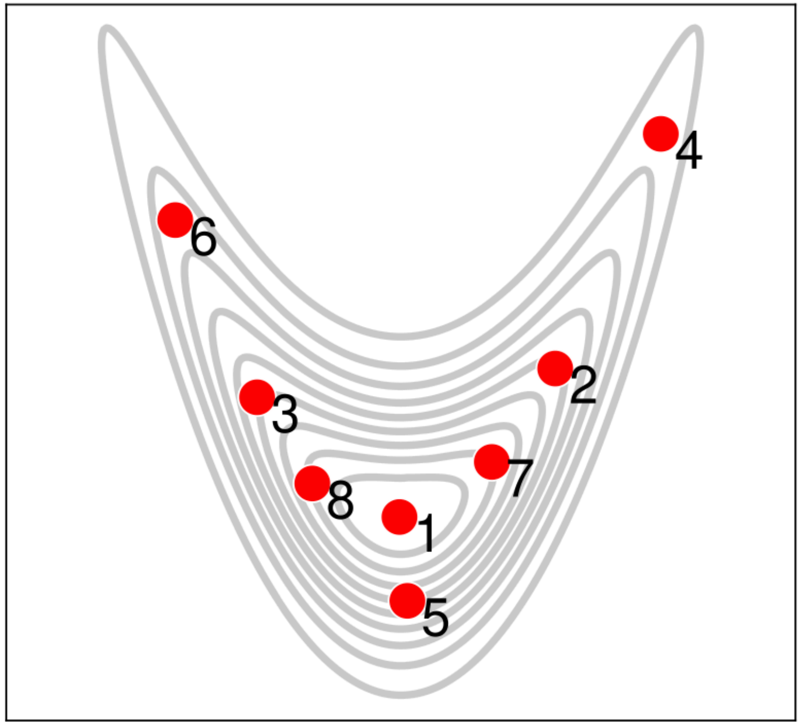
Deterministic approximations

- Suppose we have a target distribution P.
- We would like a very good approximation of the form:

$$P \approx \frac{1}{n} \sum_{i=1}^{n} \delta_{x_i}$$

- The main question is:

"How should we pick the points $x_1, \ldots, x_n$?"



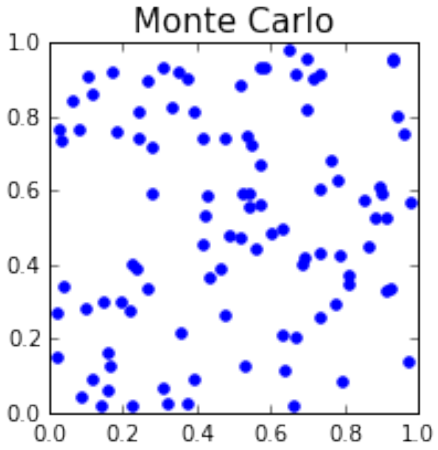
Monte Carlo vs quasi-Monte Carlo

- There is lots of research on this question when $P = \text{Unif}([0,1]^d)$….

Monte Carlo vs quasi-Monte Carlo

- There is lots of research on this question when $P = \text{Unif}([0,1]^d)$....

- The simplest option would be Monte Carlo; i.e. to sample iid observation from P.

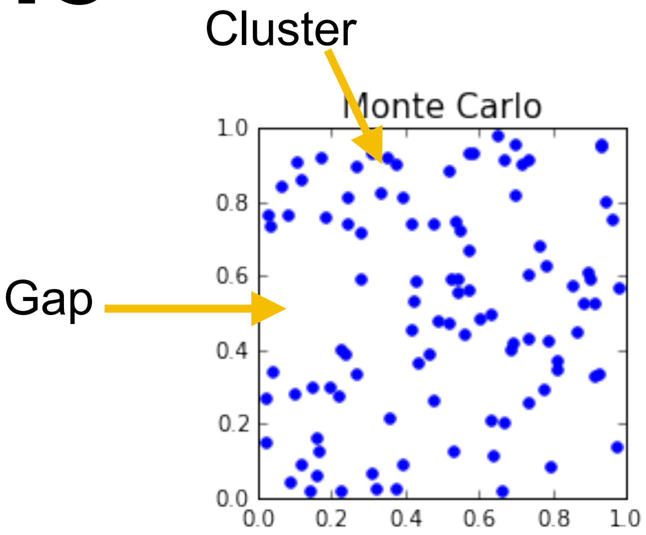


Monte Carlo vs quasi-Monte Carlo

- There is lots of research on this question when $P = \text{Unif}([0,1]^d)$....

- The simplest option would be Monte Carlo; i.e. to sample iid observation from P.

- This is wasteful because it leaves lots of gaps or clustered points..



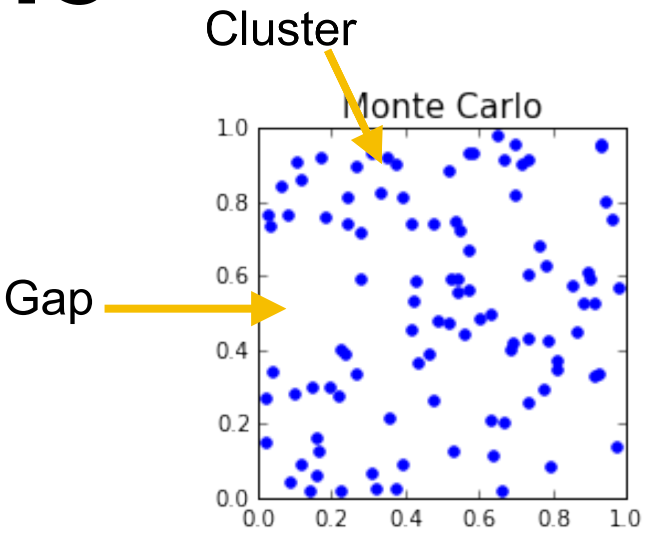
Monte Carlo vs quasi-Monte Carlo

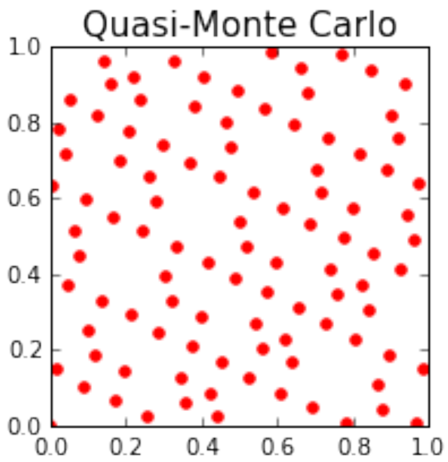
- There is lots of research on this question when $P = \text{Unif}([0,1]^d)$….

- The simplest option would be Monte Carlo; i.e. to sample iid observation from P.

- This is wasteful because it leaves lots of gaps or clustered points..

- Instead, a zoo of deterministic point sets or sequences have been proposed under the name **Quasi-Monte Carlo.**





High-level idea behind QMC

- QMC points aim to do the following:

$$D\left(P, \frac{1}{n}\sum_{i=1}^{n} \delta_{x_i}\right) \to 0 \qquad \text{at a “fast” rate as} \qquad n \to \infty$$

High-level idea behind QMC

- QMC points aim to do the following:

$$D\left(P, \frac{1}{n}\sum_{i=1}^{n} \delta_{x_i}\right) \to 0 \qquad \text{at a "fast" rate as} \qquad n \to \infty$$

- "Fast" typically means at least $O\left(\frac{\log(n)^\alpha}{n}\right)$

High-level idea behind QMC

- QMC points aim to do the following:

$$D\left(P, \frac{1}{n}\sum_{i=1}^{n}\delta_{x_i}\right) \to 0 \qquad \text{at a "fast" rate as} \qquad n \to \infty$$

- "Fast" typically means at least $O\left(\frac{\log(n)^{\alpha}}{n}\right)$

- D is typically the **star discrepancy.**

The star discrepancy

- The **star discrepancy** is a function of a dataset which tells us how spread out these point are over the domain.

$$D_{\text{star}}\left(U([0,1]^d), \frac{1}{n}\sum_{i=1}^{n}\delta_{x_i}\right)$$

$$= \sup_{B=[0,B_1)\times\ldots\times[0,B_d)} \left| \frac{\text{\#points in } B}{n} - \text{Vol}(B) \right|$$

The star discrepancy

- The **star discrepancy** is a function of a dataset which tells us how spread out these point are over the domain.

$$D_{\text{star}}\left(U([0,1]^d), \frac{1}{n}\sum_{i=1}^{n}\delta_{x_i}\right)$$

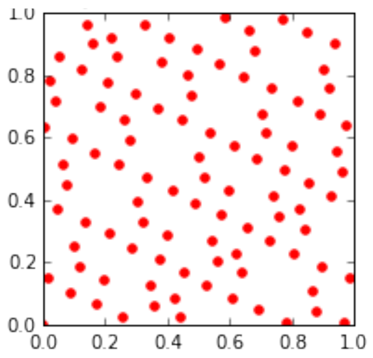
$$= \sup_{B=[0,B_1)\times\ldots\times[0,B_d)} \left| \frac{\#\text{points in } B}{n} - \text{Vol}(B) \right|$$

Sup over boxes anchored at origin!

The star discrepancy

- The **star discrepancy** is a function of a dataset which tells us how spread out these point are over the domain.

Dataset



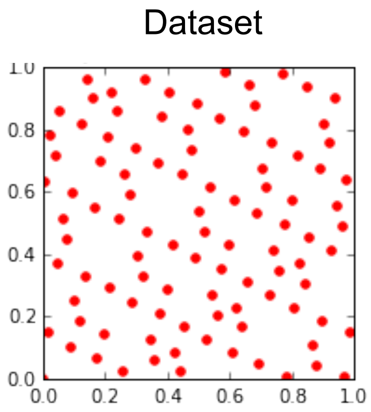
$$D_{\text{star}}\left(U([0,1]^d), \frac{1}{n}\sum_{i=1}^{n} \delta_{x_i} \right)$$

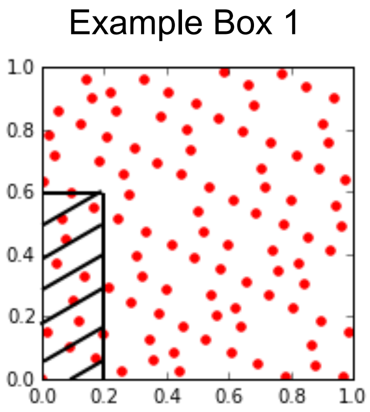
$$= \sup_{B=[0,B_1)\times\ldots\times[0,B_d)} \left| \frac{\#\text{points in } B}{n} - \text{Vol}(B) \right|$$

Sup over boxes anchored at origin!

The star discrepancy

- The **star discrepancy** is a function of a dataset which tells us how spread out these point are over the domain.





$$D_{\text{star}}\left(U([0,1]^d), \frac{1}{n}\sum_{i=1}^{n}\delta_{x_i}\right)$$

$$= \sup_{B=[0,B_1)\times\ldots\times[0,B_d)} \left| \frac{\#\text{points in } B}{n} - \text{Vol}(B) \right|$$

Sup over boxes anchored at origin!

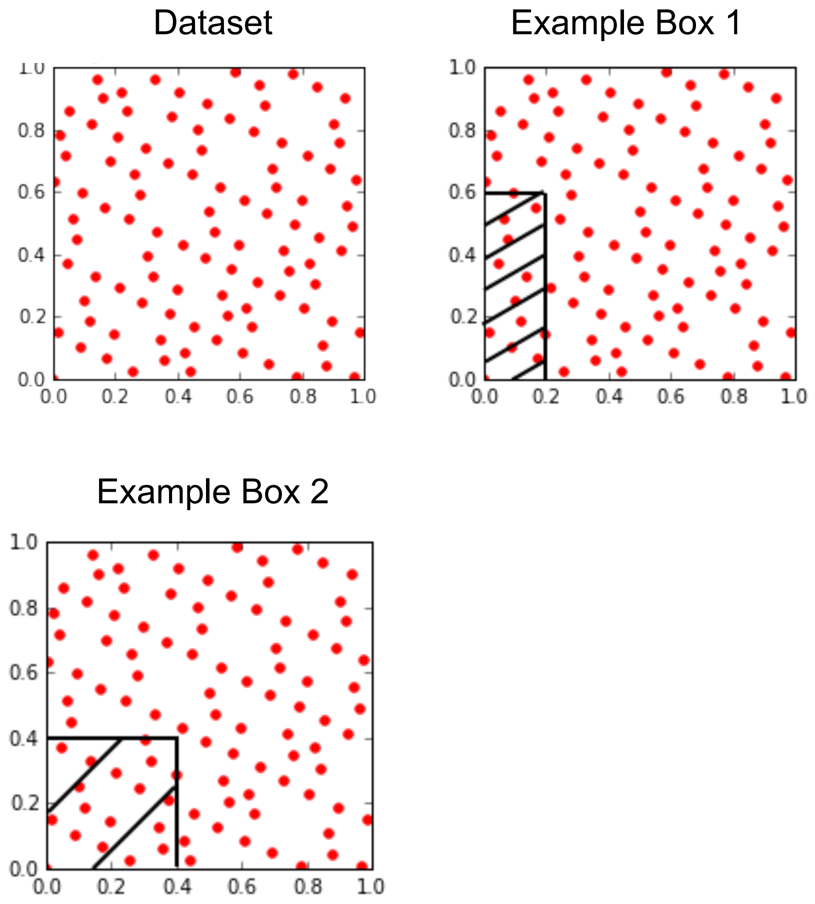
The star discrepancy

- The **star discrepancy** is a function of a dataset which tells us how spread out these point are over the domain.

$$D_{\text{star}}\left(U([0,1]^d), \frac{1}{n}\sum_{i=1}^{n}\delta_{x_i}\right)$$

$$= \sup_{B=[0,B_1)\times\ldots\times[0,B_d)} \left| \frac{\#\text{points in } B}{n} - \text{Vol}(B) \right|$$

Sup over boxes anchored at origin!



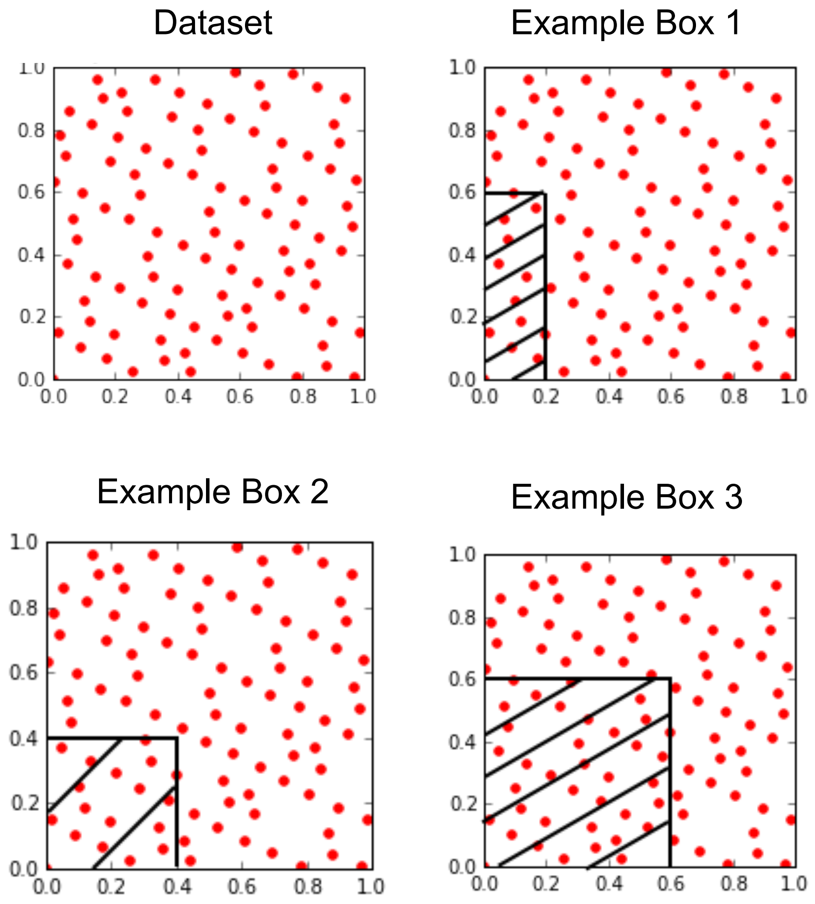
The star discrepancy

- The **star discrepancy** is a function of a dataset which tells us how spread out these point are over the domain.

$$D_{\text{star}}\left(U([0,1]^d), \frac{1}{n}\sum_{i=1}^{n}\delta_{x_i}\right)$$

$$= \sup_{B=[0,B_1)\times\ldots\times[0,B_d)} \left| \frac{\text{\#points in } B}{n} - \text{Vol}(B) \right|$$

Sup over boxes anchored at origin!

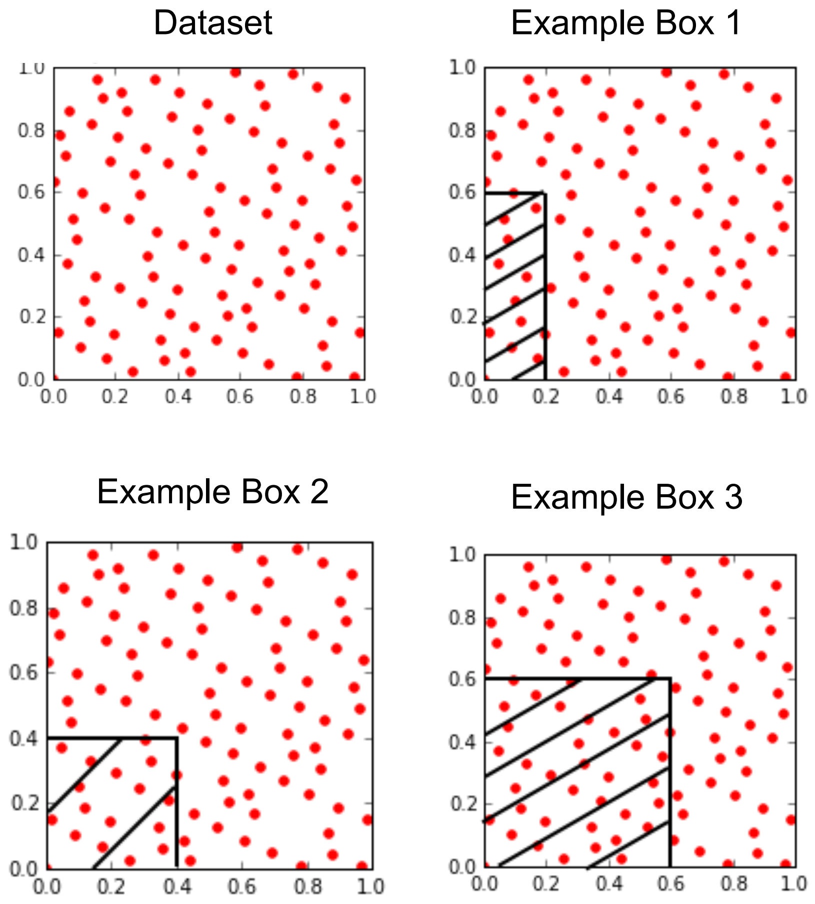


The star discrepancy

- The **star discrepancy** is a function of a dataset which tells us how spread out these point are over the domain.

$$D_{\text{star}}\left(U([0,1]^d), \frac{1}{n}\sum_{i=1}^{n} \delta_{x_i}\right) = \sup_{B=[0,B_1)\times\ldots\times[0,B_d)} \left| \frac{\#\text{points in } B}{n} - \text{Vol}(B) \right|$$

Sup over boxes anchored at origin!



- It can also be thought of as a measure of dissimilarity between our dataset and a $U([0,1]^d)$!

Low-discrepancy sequences

- The star discrepancy is a convenient choice since we have that:

$$\left| \mathbb{E}_{X\sim P}[f(X)] - \frac{1}{n}\sum_{i=1}^{n} f(x_i) \right| \leq V(f) \times D_{\text{star}}\left(U([0,1]^d), \frac{1}{n}\sum_{i=1}^{n} \delta_{x_i} \right)$$

Low-discrepancy sequences

- The star discrepancy is a convenient choice since we have that:

$$\left| \mathbb{E}_{X\sim P}[f(X)] - \frac{1}{n}\sum_{i=1}^{n} f(x_i) \right| \leq V(f) \times D_{\text{star}}\left(U([0,1]^d), \frac{1}{n}\sum_{i=1}^{n} \delta_{x_i} \right)$$

Integration error

Complexity of the function

Star discrepancy

Low-discrepancy sequences

- The star discrepancy is a convenient choice since we have that:

$$\underbrace{\left| \mathbb{E}_{X\sim P}[f(X)] - \frac{1}{n}\sum_{i=1}^{n} f(x_i) \right|}_{\text{Integration error}} \leq \underbrace{V(f)}_{\text{Complexity of the function}} \times \underbrace{D_{\text{star}}\left(U([0,1]^d), \frac{1}{n}\sum_{i=1}^{n} \delta_{x_i} \right)}_{\text{Star discrepancy}}$$

- Low discrepancy sequences include well known constructions such as Sobol and Halton sequences, for which we therefore have guarantees of fast convergence of the integration error to zero!

Low-discrepancy sequences

- The star discrepancy is a convenient choice since we have that:

$$\underbrace{\left| \mathbb{E}_{X\sim P}[f(X)] - \frac{1}{n}\sum_{i=1}^{n} f(x_i) \right|}_{\text{Integration error}} \leq \underbrace{V(f)}_{\text{Complexity of the function}} \times \underbrace{D_{\text{star}}\left(U([0,1]^d), \frac{1}{n}\sum_{i=1}^{n}\delta_{x_i} \right)}_{\text{Star discrepancy}}$$

- Low discrepancy sequences include well known constructions such as Sobol and Halton sequences, for which we therefore have guarantees of fast convergence of the integration error to zero!

→ A major limitation of this approach is that you can only approximate $P = \text{Unif}([0,1]^d)$!

Stein Points

- Choosing another discrepancy (i.e. our favourite hammer) can lead to more practical algorithms:

$$\arg \min_{x_1,\ldots,x_n \in \mathbb{R}^d} \mathrm{KSD}\left(P \,\middle|\middle|\, \frac{1}{n} \sum_{i=1}^{n} \delta_{x_i} \right)$$

- This is still a very high-dimensional and non-convex optimisation problem, so we need to introduce some approximation….

Chen, W. Y., Mackey, L., Gorham, J., **Briol, F-X.**, & Oates, C. J. (2018). Stein points. *ICML*, 1320–1350.

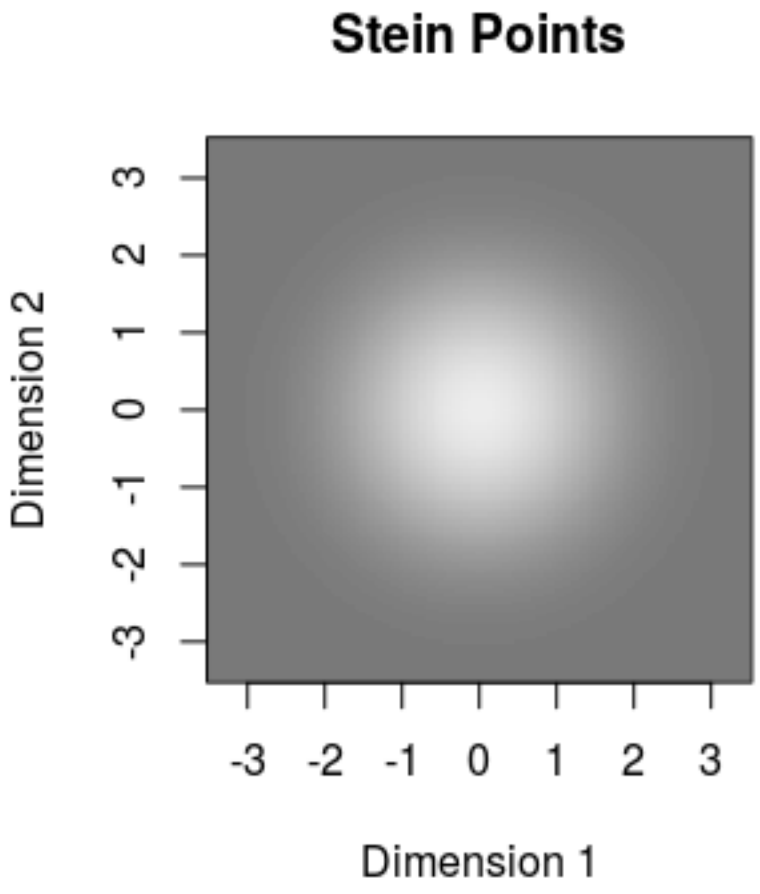
Chen, W. Y., Barp, A., **Briol, F-X**., Gorham, J., Girolami, M., Mackey, L., & Oates, C. J. (2019). Stein point Markov chain Monte Carlo. *ICML*, 1737–1767.

Greedy Stein Points

- We choose points one at a time to decrease the KSD the most.
- Thanks to the nice expression for the KSD, this simply becomes:

$$x_n \in \arg\min_{x \in \mathbb{R}^d} \text{KSD}\left(P \middle|\middle| \frac{1}{n}\sum_{i=1}^{n-1} \delta_{x_i} + \frac{1}{n}\delta_x\right)$$

$$= \arg\min_{x \in \mathbb{R}^d} \frac{k_P(x,x)}{2} + \sum_{i=1}^{n-1} k_P(x_i, x)$$



Example: 2d-Gaussian.

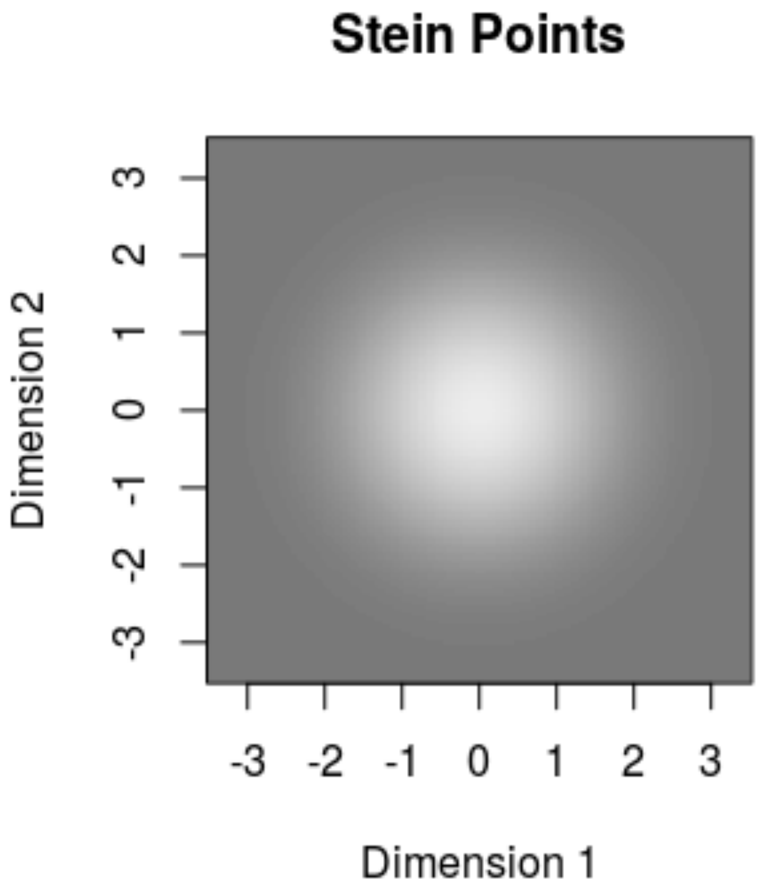
Greedy Stein Points

- We choose points one at a time to decrease the KSD the most.

- Thanks to the nice expression for the KSD, this simply becomes:

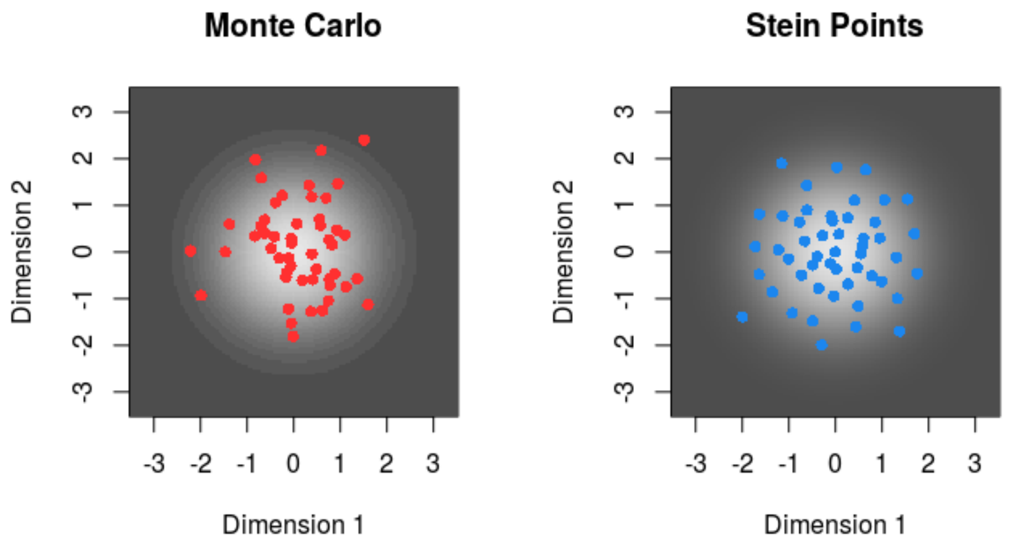
$$x_n \in \arg\min_{x \in \mathbb{R}^d} \mathrm{KSD}\left(P \middle|\middle| \frac{1}{n}\sum_{i=1}^{n-1} \delta_{x_i} + \frac{1}{n}\delta_x\right)$$

$$= \arg\min_{x \in \mathbb{R}^d} \frac{k_P(x,x)}{2} + \sum_{i=1}^{n-1} k_P(x_i, x)$$



Example: 2d-Gaussian.

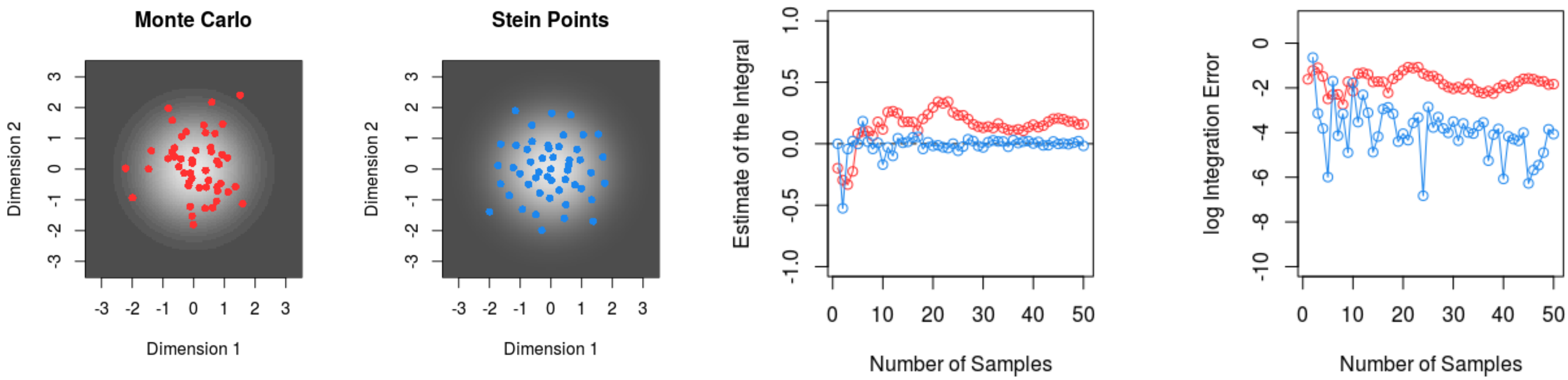
Greedy Stein Points on Gaussian



$$\mathbb{E}_{X\sim P}[f(X)] = ?$$

$$f(x) = \sin(x_1) + \sin(x_2) \qquad P = N(0, I_{2\times 2})$$

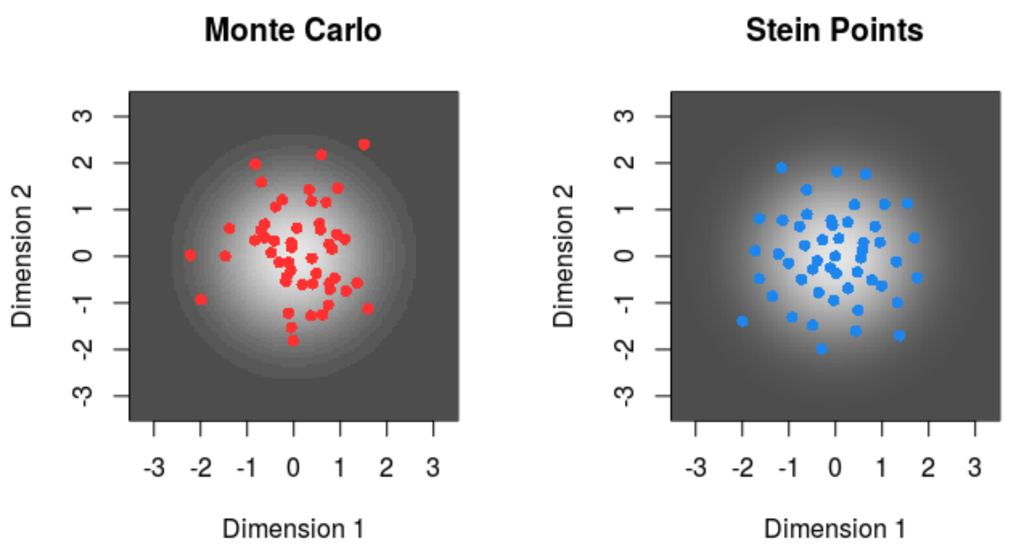
Greedy Stein Points on Gaussian

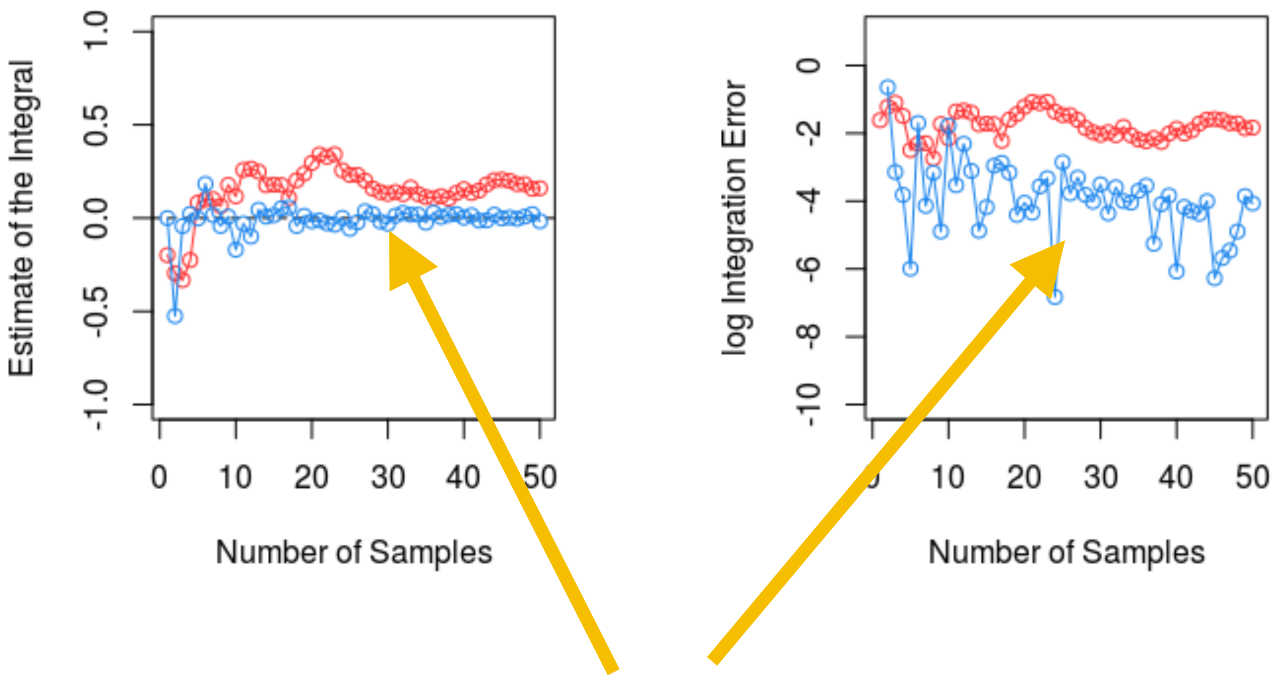


$$\mathbb{E}_{X \sim P}[f(X)] = ?$$

$$f(x) = \sin(x_1) + \sin(x_2) \qquad P = N(0, I_{2\times 2})$$

Greedy Stein Points on Gaussian





$$\mathbb{E}_{X \sim P}[f(X)] = ?$$

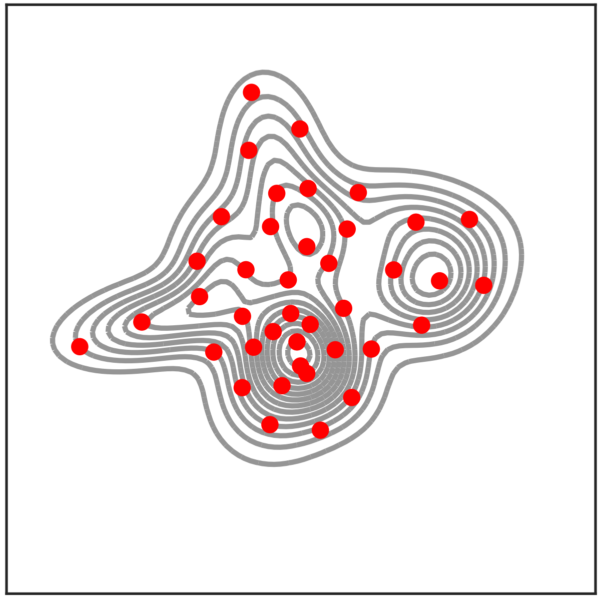
$$f(x) = \sin(x_1) + \sin(x_2) \qquad P = N(0, I_{2\times 2})$$

Convergence is much faster for Stein Points than Monte Carlo!

Stein Points for complex targets

- One of the main advantages of Stein points is that we can approximate any distribution P for which we have a suitable Stein characterisation!

- This includes complex probabilistic models, or Bayesian posterior distributions!



https://github.com/wilson-ye-chen/stein_points

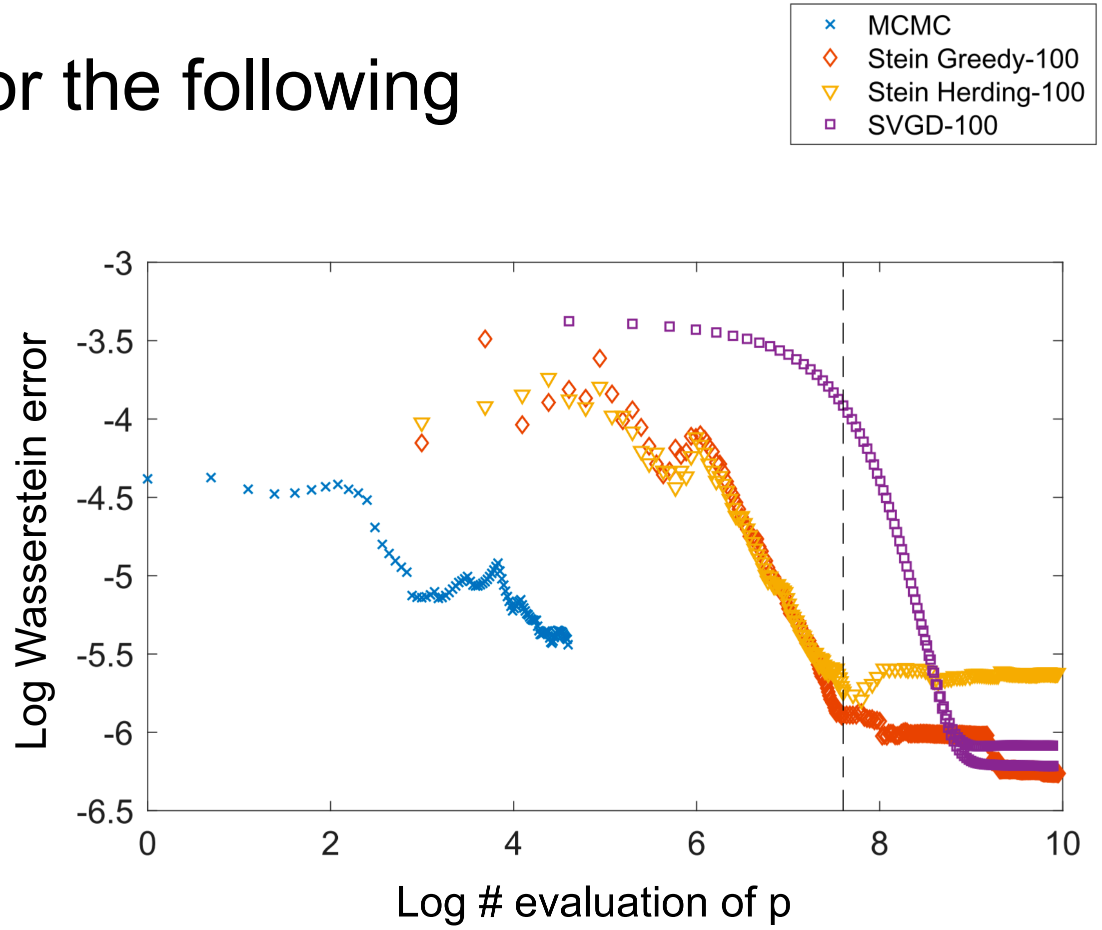
https://github.com/wilson-ye-chen/sp-mcmc

Example: IGARCH posterior

- Consider some Bayesian posterior for the following time-series model:

$$y_t = \sigma_t \epsilon_t, \quad \epsilon_t \sim N(0,1)$$

$$\sigma_t^2 = \boxed{\theta_1} + \boxed{\theta_2} y_{t-1}^2 + (1 - \boxed{\theta_2})\sigma_{t-1}^2$$



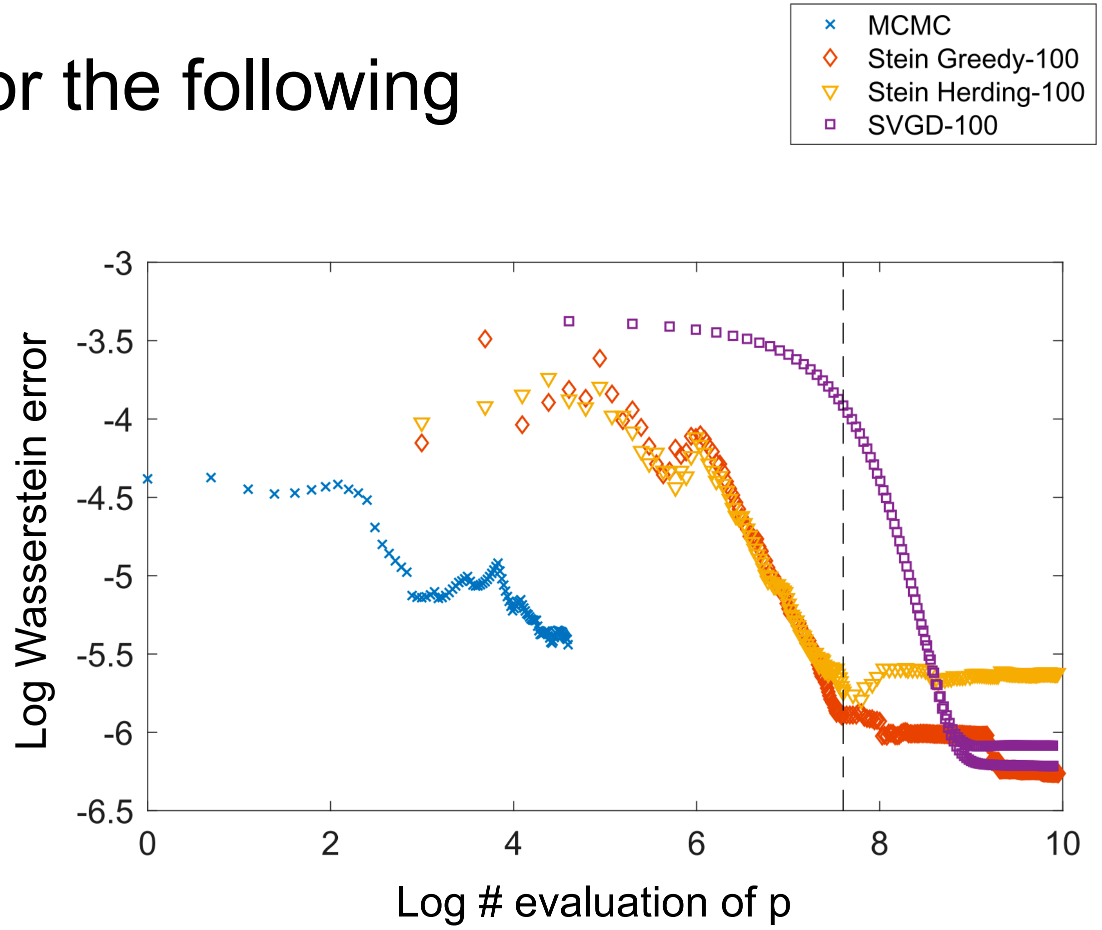
Example: IGARCH posterior

- Consider some Bayesian posterior for the following time-series model:

$$y_t = \sigma_t \epsilon_t, \quad \epsilon_t \sim N(0,1)$$

$$\sigma_t^2 = \boxed{\theta_1} + \boxed{\theta_2} y_{t-1}^2 + (1 - \boxed{\theta_2})\sigma_{t-1}^2$$

- Stein points give much smaller Wasserstein distance approximation than MCMC!

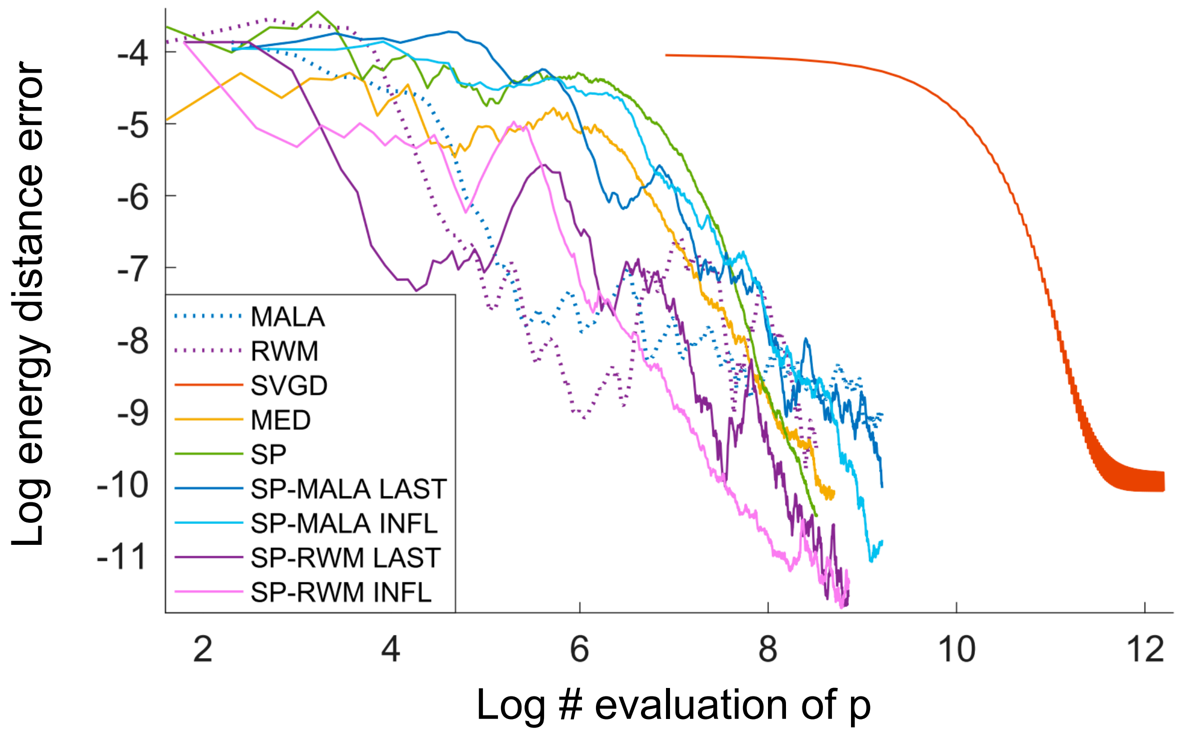


Example: IGARCH posterior

- Consider some Bayesian posterior for the following time-series model:

$$y_t = \sigma_t \epsilon_t, \quad \epsilon_t \sim N(0,1)$$

$$\sigma_t^2 = \boxed{\theta_1} + \boxed{\theta_2} y_{t-1}^2 + (1 - \boxed{\theta_2}) \sigma_{t-1}^2$$



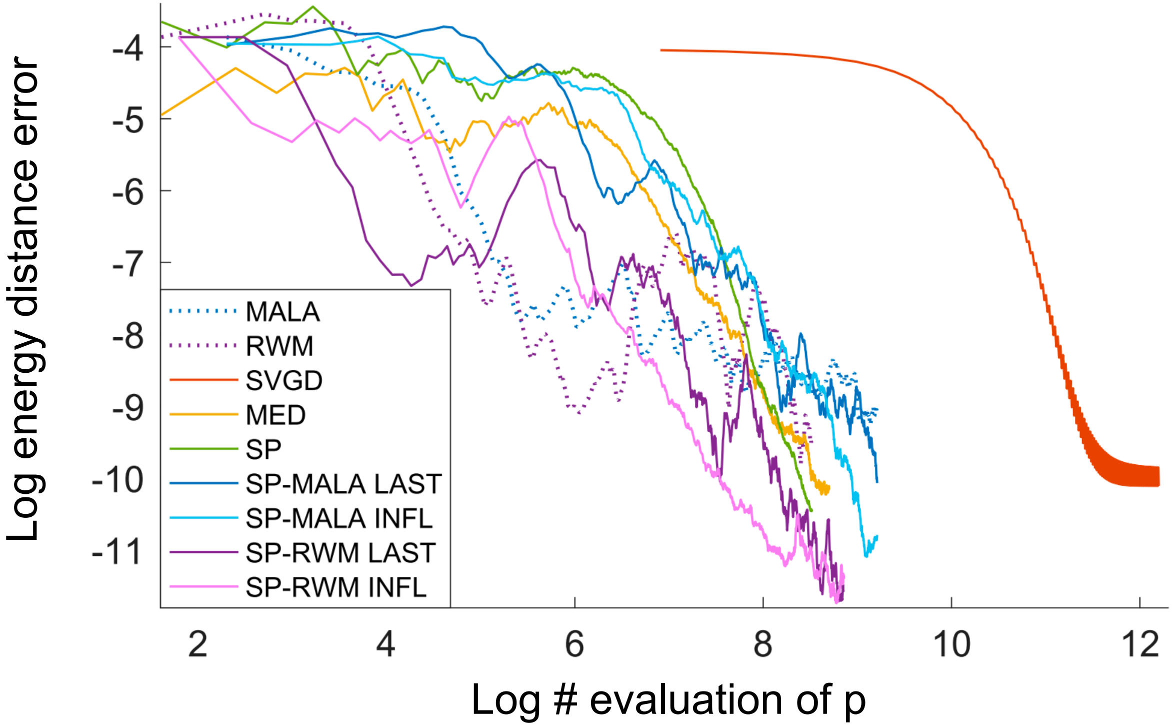
Example: IGARCH posterior

- Consider some Bayesian posterior for the following time-series model:

$$y_t = \sigma_t \epsilon_t, \quad \epsilon_t \sim N(0,1)$$

$$\sigma_t^2 = \boxed{\theta_1} + \boxed{\theta_2} y_{t-1}^2 + (1 - \boxed{\theta_2}) \sigma_{t-1}^2$$

- Advanced versions of Stein Points can do much better…



Stein's method as a computational tool

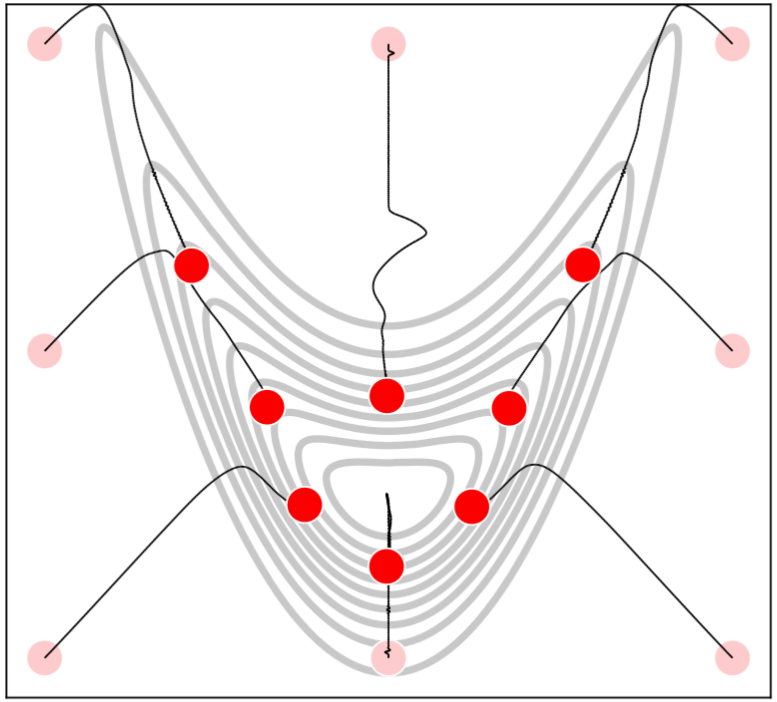
Stein Variational Gradient Descent

Particle-based approximations

Stein Variational Gradient Descent: A General Purpose Bayesian Inference Algorithm

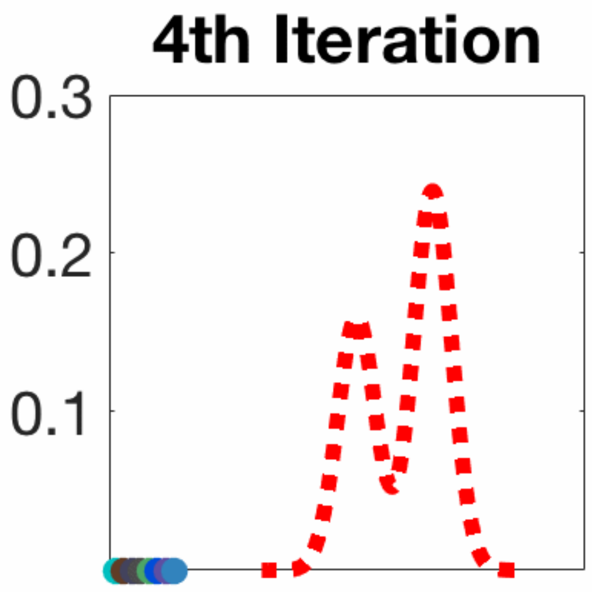
Qiang Liu **Dilin Wang**
Department of Computer Science
Dartmouth College

Stein Variational Gradient Descent



Particle-based approximations

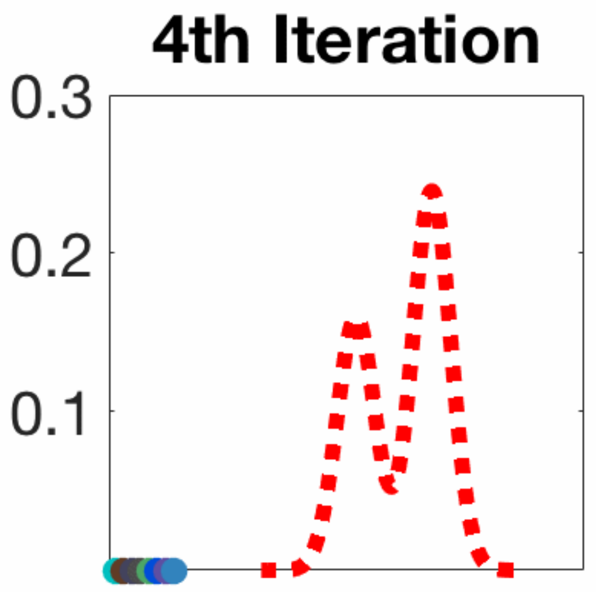
- We are still interested in approximating some distribution P.
- This time, we start with some particles which we then move towards P.



[**Credit:** Qiang Liu (UT Austin) https://www.cs.utexas.edu/~qlearning/project.html?p=svgd]

Particle-based approximations

- We are still interested in approximating some distribution P.
- This time, we start with some particles which we then move towards P.

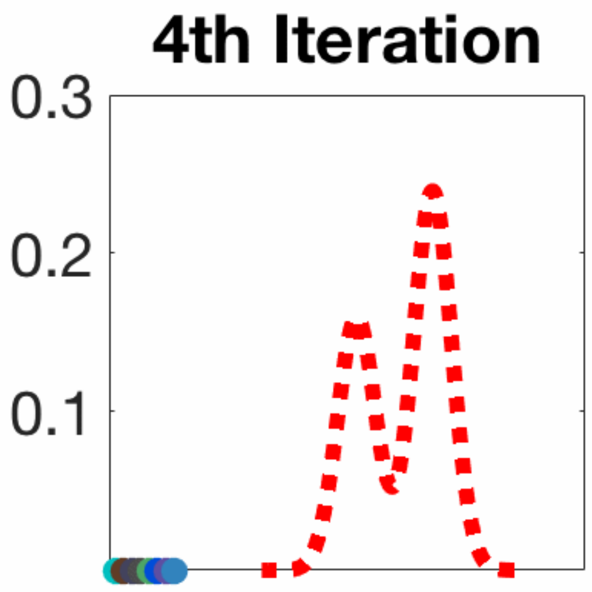


[**Credit:** Qiang Liu (UT Austin) https://www.cs.utexas.edu/~qlearning/project.html?p=svgd]

Particle-based approximations

- We are still interested in approximating some distribution P.
- This time, we start with some particles which we then move towards P.
- The idea is to define a map, and to recursively transport particles through this map towards P:

$$\Phi^g(x) = x + \epsilon g(x)$$



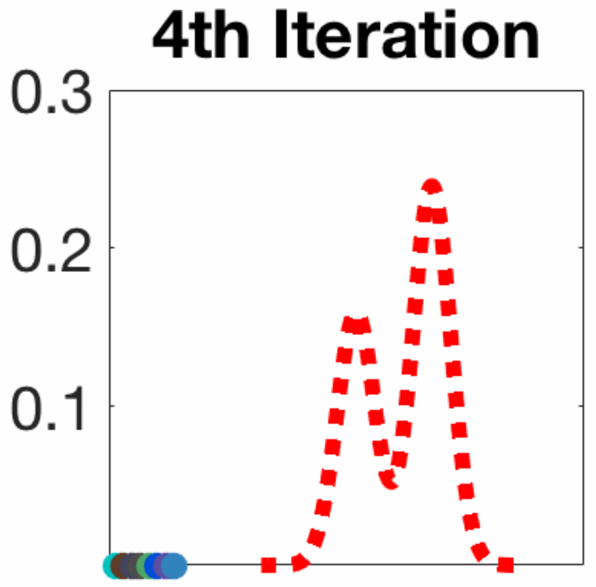
[**Credit:** Qiang Liu (UT Austin) https://www.cs.utexas.edu/~qlearning/project.html?p=svgd]

Particle-based approximations

- We are still interested in approximating some distribution P.
- This time, we start with some particles which we then move towards P.
- The idea is to define a map, and to recursively transport particles through this map towards P:

$$\Phi^g(x) = x + \epsilon g(x)$$

Current Position — Step-size — Direction of move



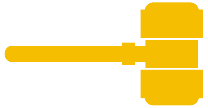
[**Credit:** Qiang Liu (UT Austin) https://www.cs.utexas.edu/~qlearning/project.html?p=svgd]

SVGD as transport of measure

- The algorithm mostly relies on this identity:

$$\max_{g \in \mathcal{G}} \left\{ -\frac{d}{d\epsilon} \mathrm{KL}\left(\Phi^g_{\#} Q \,||\, P \right) \Big|_{\epsilon=0} \right\} = \mathrm{KSD}(P \,||\, Q)$$

where $\Phi^g(x) = x + \epsilon g(x)$



Once again we are using our hammer….

SVGD as transport of measure

- The algorithm mostly relies on this identity:

$$\max_{g \in \mathcal{G}} \left\{ \boxed{ -\frac{d}{d\epsilon} \mathrm{KL}\left(\Phi^g_{\#} Q \,||\, P \right) \Big|_{\epsilon=0} } \right\} = \mathrm{KSD}(P \,||\, Q)$$

The rate of decrease of the KL divergence under the transport map Φ^g

where $\Phi^g(x) = x + \epsilon g(x)$

Once again we are using our hammer….

SVGD as transport of measure

- The algorithm mostly relies on this identity:

$$\max_{g \in \mathcal{G}} \left\{ \boxed{ -\frac{d}{d\epsilon} \mathrm{KL}\left(\Phi^g_{\#} Q \,||\, P \right) \Big|_{\epsilon=0} } \right\} = \mathrm{KSD}(P\,||\,Q)$$

where $\Phi^g(x) = x + \epsilon g(x)$

The rate of decrease of the KL divergence under the transport map Φ^g

- The best transport map is therefore the function:

$$g^*_{Q,P}(\cdot) \propto \mathbb{E}_{X \sim Q}[\nabla \log p(X) k(X, \cdot) + \nabla_x k(X, \cdot)]$$

Once again we are using our hammer….

Stein variational gradient descent (SVGD)

- We should therefore move as follows:

$$\Phi^*(x) = x + \epsilon g^*_{Q,P}(x) = x + \epsilon \mathbb{E}_{X \sim Q}[\nabla \log p(X) k(X, x) + \nabla_x k(X, x)]$$

Stein variational gradient descent (SVGD)

- We should therefore move as follows:

$$\Phi^*(x) = x + \epsilon g^*_{Q,P}(x) = x + \epsilon \mathbb{E}_{X\sim Q}[\nabla \log p(X) k(X, x) + \nabla_x k(X, x)]$$

- In practice, we do not have Q but a particle approximation:

$$x_i^{t+1} \leftarrow x_i^t + \epsilon \frac{1}{n} \sum_{j=1}^{n} \nabla \log p(x_j^t)(\mathbf{1} \times k(x_j^t, x_i^t)) + \nabla_{x_j} k(x_j^t, x_i^t)$$

for every iteration $t = 1,2,\ldots,T$.

Stein variational gradient descent (SVGD)

- We should therefore move as follows:

$$\Phi^*(x) = x + \epsilon g^*_{Q,P}(x) = x + \epsilon \mathbb{E}_{X \sim Q}[\nabla \log p(X) k(X, x) + \nabla_x k(X, x)]$$

- In practice, we do not have Q but a particle approximation:

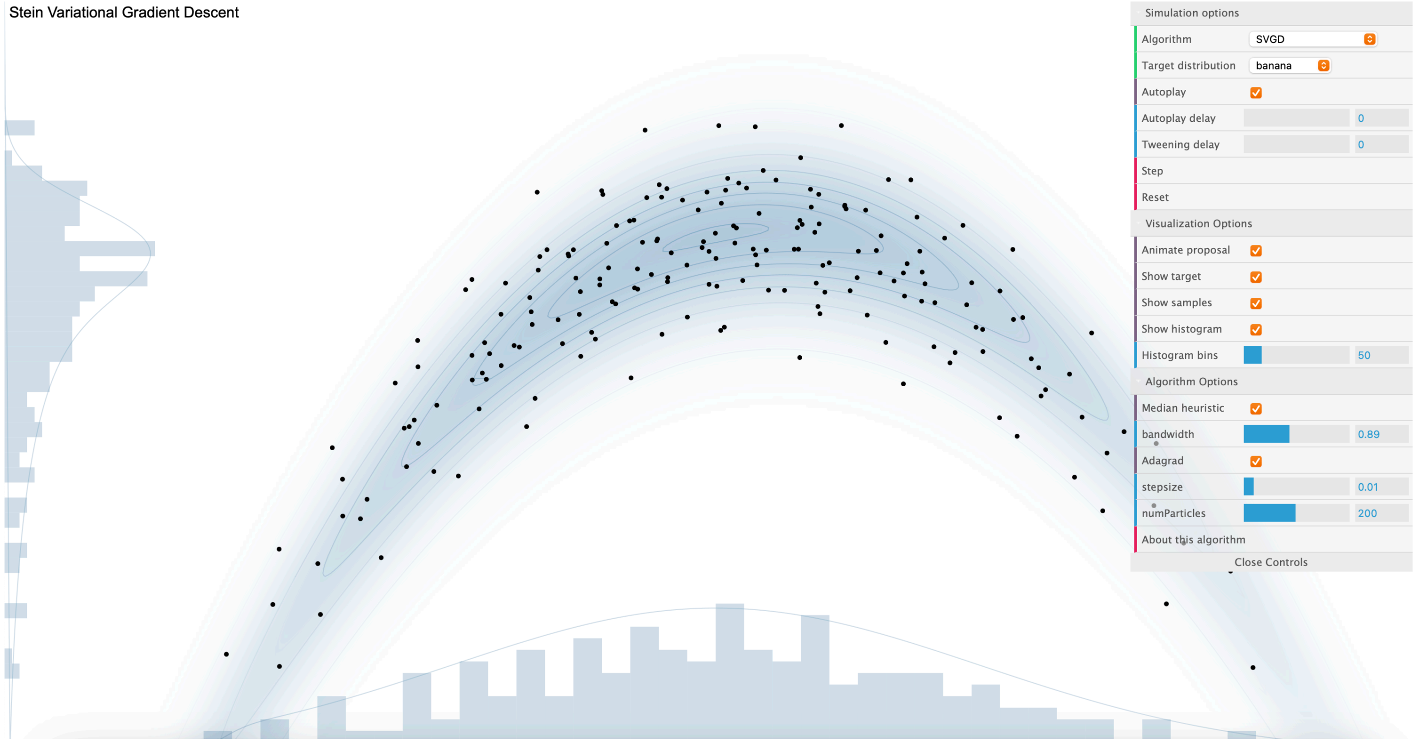
$$x_i^{t+1} \leftarrow x_i^t + \epsilon \frac{1}{n} \sum_{j=1}^{n} \nabla \log p(x_j^t)(\mathbf{1} \times k(x_j^t, x_i^t)) + \nabla_{x_j} k(x_j^t, x_i^t)$$

for every iteration $t = 1, 2, \ldots, T$.

Pushes particles towards regions of high prob under P

Pushes particles away from one another ("repulsive force")

SVGD in practice



https://chi-feng.github.io/mcmc-demo/app.html?algorithm=SVGD

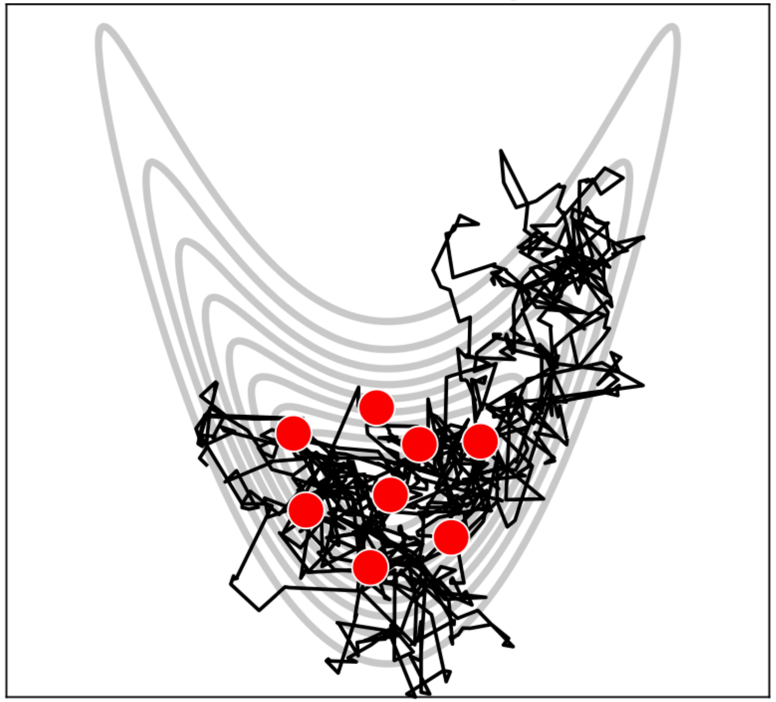
Stein’s method as a computational tool

Thinning MCMC

Thinning MCMC

- Suppose we would like to compute some predictive distribution:

$$p(y^* \mid x^*, x_1, \ldots, x_n, y_1, \ldots, y_n) = \int_\Theta p(y^* \mid x^*, \theta) p(\theta \mid x_1, \ldots, x_n, y_1, \ldots, y_n) d\theta$$



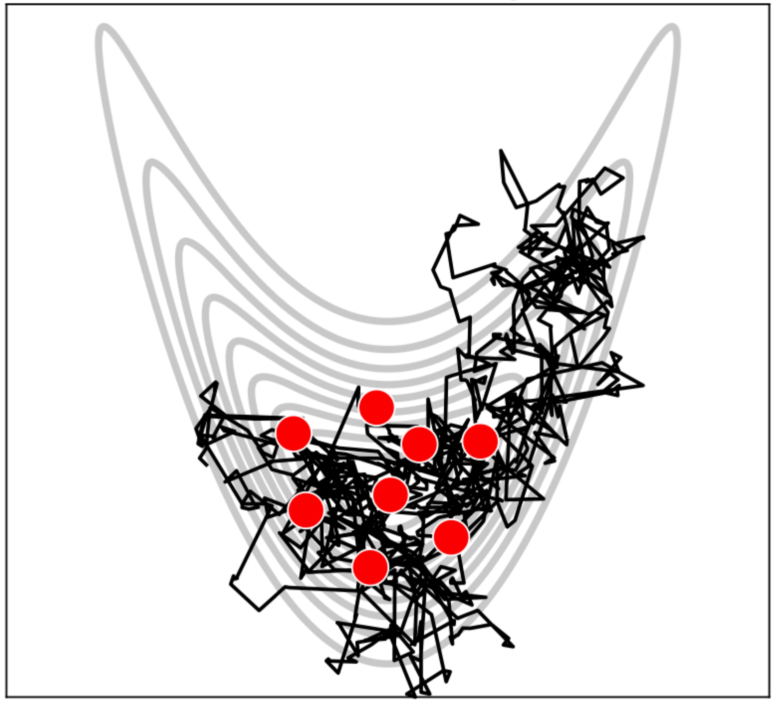
Thinning MCMC

- Suppose we would like to compute some predictive distribution:

$$p(y^* \mid x^*, x_1, \ldots, x_n, y_1, \ldots, y_n)$$
$$= \int_\Theta p(y^* \mid x^*, \theta)p(\theta \mid x_1, \ldots, x_n, y_1, \ldots, y_n)d\theta$$

Expensive likelihood

Posterior distribution
(approximated with samples)



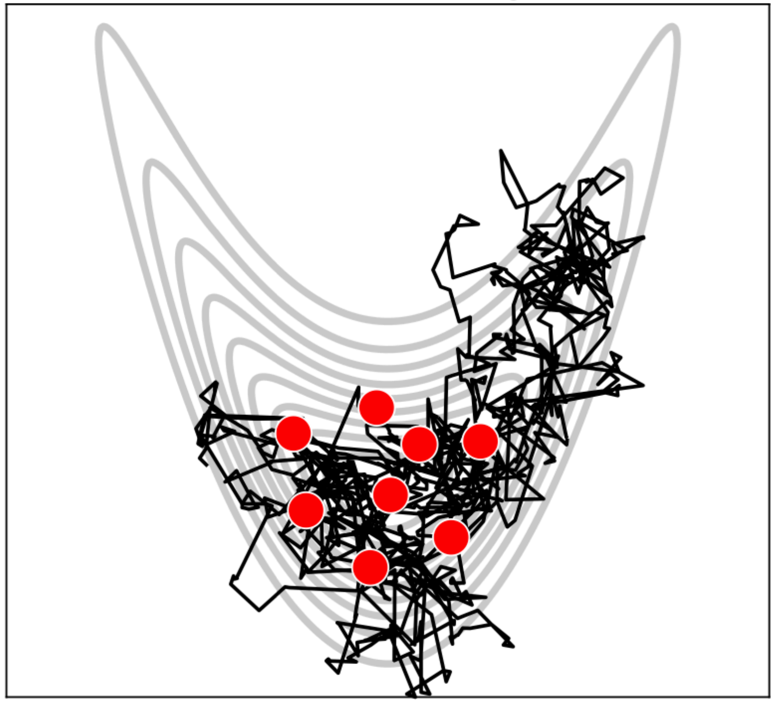
Thinning MCMC

- Suppose we would like to compute some predictive distribution:

$$p(y^* \mid x^*, x_1, \ldots, x_n, y_1, \ldots, y_n) = \int_\Theta p(y^* \mid x^*, \theta) p(\theta \mid x_1, \ldots, x_n, y_1, \ldots, y_n) d\theta$$

Expensive likelihood

Posterior distribution (approximated with samples)



- Clearly we do not want to have a very long chain as this will otherwise be very expensive!

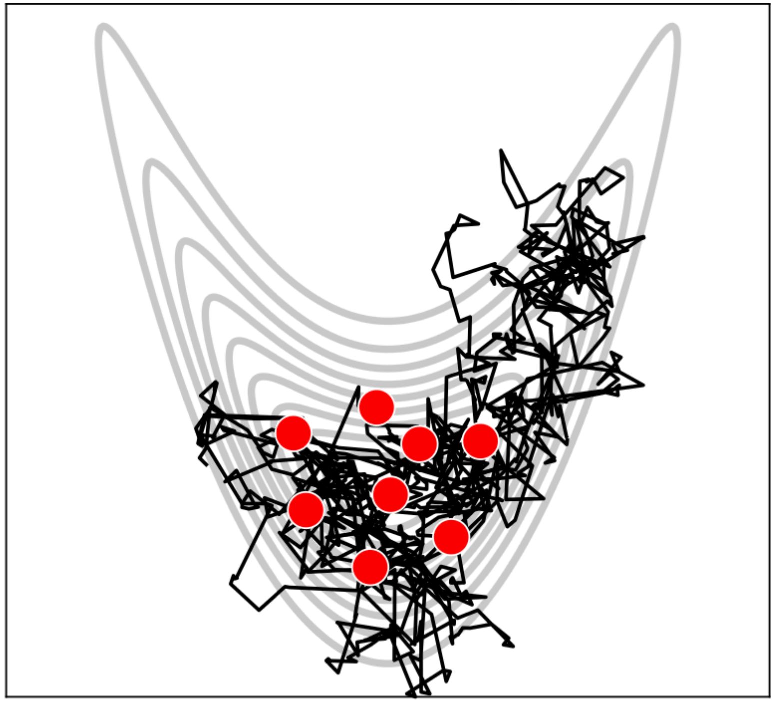
Thinning MCMC

- Suppose we would like to compute some predictive distribution:

$$p(y^* \mid x^*, x_1, \ldots, x_n, y_1, \ldots, y_n) = \int_\Theta p(y^* \mid x^*, \theta) p(\theta \mid x_1, \ldots, x_n, y_1, \ldots, y_n) d\theta$$

Expensive likelihood

Posterior distribution
(approximated with samples)



- Clearly we do not want to have a very long chain as this will otherwise be very expensive!

- **Solution:** Thinning our MCMC sampler!

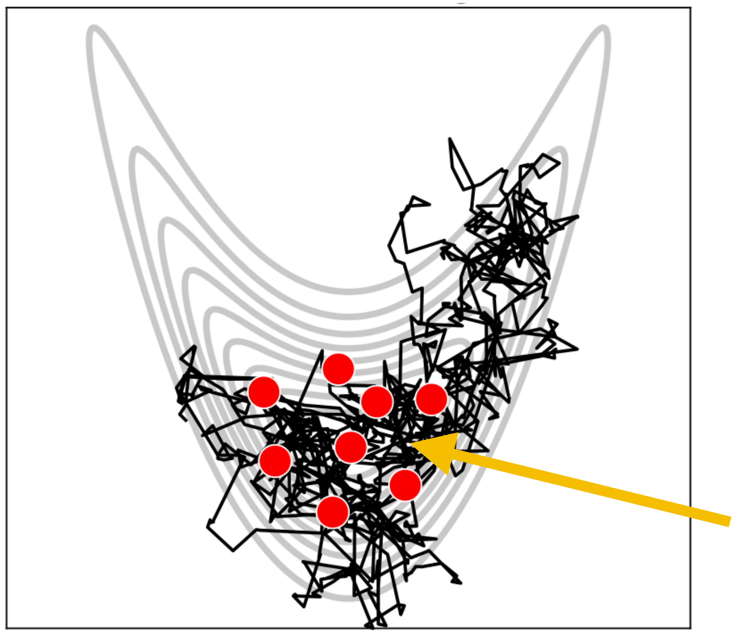
Thinning MCMC

- The simplest method is independent sub-sampling.
- …but the independence can be quite **wasteful** as we might end up with some very similar samples!

Thinning MCMC

- The simplest method is independent sub-sampling.

- …but the independence can be quite **wasteful** as we might end up with some very similar samples!



We ideally want an approximation where points are far from one another but concentrated in region of high probability mass…

Stein thinning

- Let's use our favourite hammer on this nail:

$$\arg \min_{\{x_i\}_{i=1}^n \subset \{y_i\}_{i=1}^N} \text{KSD}\left(P \middle|\middle| \frac{1}{n}\sum_{i=1}^{n} \delta_{x_i}\right)$$

Riabiz, M., Chen, W., Cockayne, J., Swietach, P., Niederer, S. A., Mackey, L., & Oates, C. J. (2022). Optimal thinning of MCMC output. *JRSSB*, *84*(4), 1059–1081.

Stein thinning

- Let’s use our favourite hammer on this nail:

$$\arg \min_{\{x_i\}_{i=1}^n \subset \{y_i\}_{i=1}^N} \text{KSD}\left(P \,\Big|\Big|\, \frac{1}{n} \sum_{i=1}^n \delta_{x_i} \right)$$

Original approximation of P

Riabiz, M., Chen, W., Cockayne, J., Swietach, P., Niederer, S. A., Mackey, L., & Oates, C. J. (2022). Optimal thinning of MCMC output. *JRSSB*, *84*(4), 1059–1081.

Stein thinning

- Let's use our favourite hammer on this nail:

$$\underset{\{x_i\}_{i=1}^{n} \subset \{y_i\}_{i=1}^{N}}{\arg\min} \; \mathrm{KSD}\left(P \,\middle|\middle|\, \frac{1}{n} \sum_{i=1}^{n} \delta_{x_i} \right)$$

Sub-sample

Original approximation of P

Riabiz, M., Chen, W., Cockayne, J., Swietach, P., Niederer, S. A., Mackey, L., & Oates, C. J. (2022). Optimal thinning of MCMC output. *JRSSB*, *84*(4), 1059–1081.

UCL

Stein thinning

- Let's use our favourite hammer on this nail:

$$\underset{\{x_i\}_{i=1}^{n} \subset \{y_i\}_{i=1}^{N}}{\arg\min} \ \mathrm{KSD}\left(P \,\middle\|\, \frac{1}{n}\sum_{i=1}^{n} \delta_{x_i}\right)$$

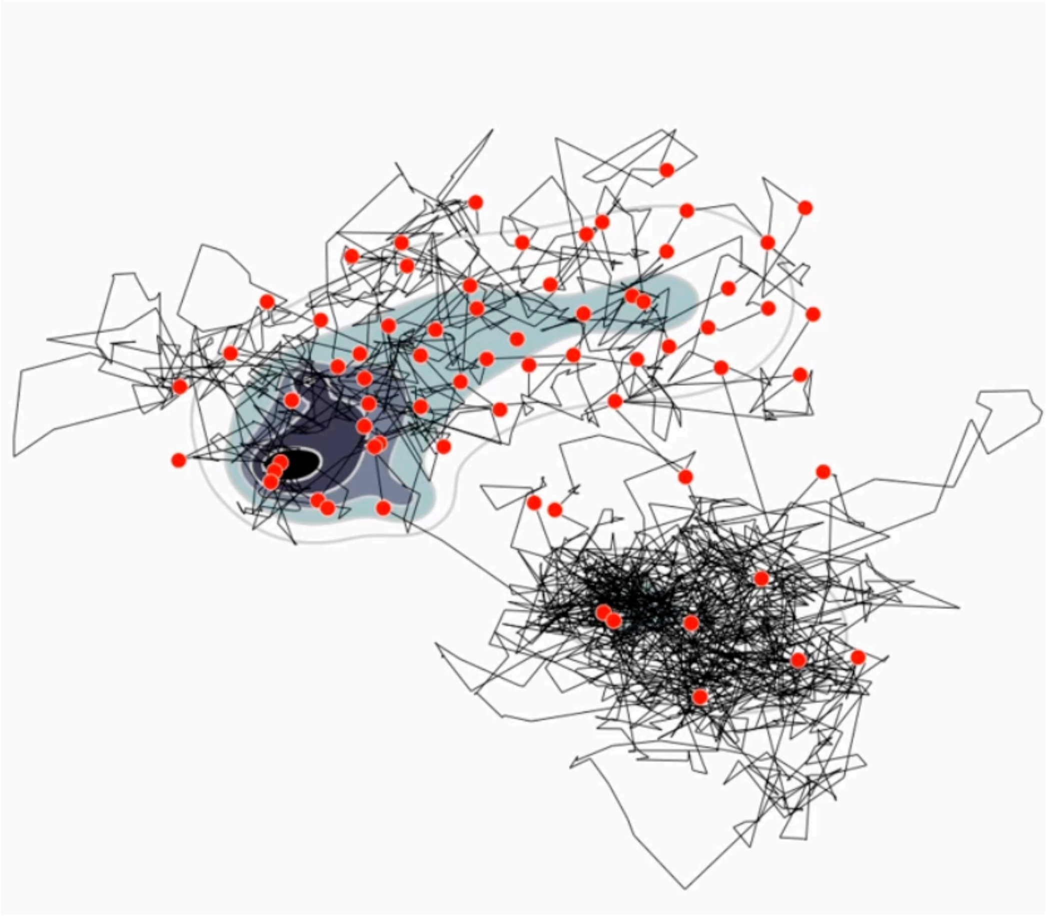
Sub-sample

Original approximation of P

- Similarly to Stein points, this is usually intractable so we select one point at a time.

Riabiz, M., Chen, W., Cockayne, J., Swietach, P., Niederer, S. A., Mackey, L., & Oates, C. J. (2022). Optimal thinning of MCMC output. *JRSSB*, *84*(4), 1059–1081.

Stein thinning in practice



[https://en.wikipedia.org/wiki/File:Stein_Thinning_of_MCMC_output.webm]

Stein's method as a computational tool

Importance sampling

Importance sampling

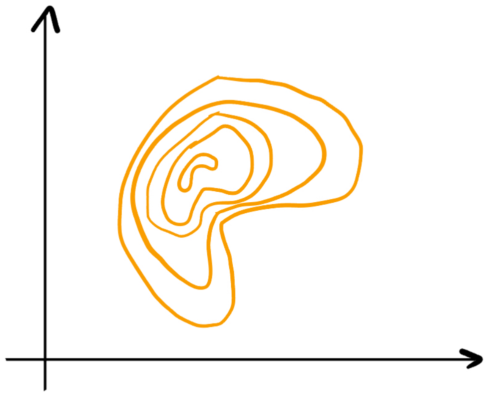
- Sometimes we want to sample from P but cannot do so…

Importance sampling

- Sometimes we want to sample from P but cannot do so…
- **Importance sampling** proposes to sample from P', then weight the samples to correct from the fact that we are sampling from the wrong distribution

Importance sampling

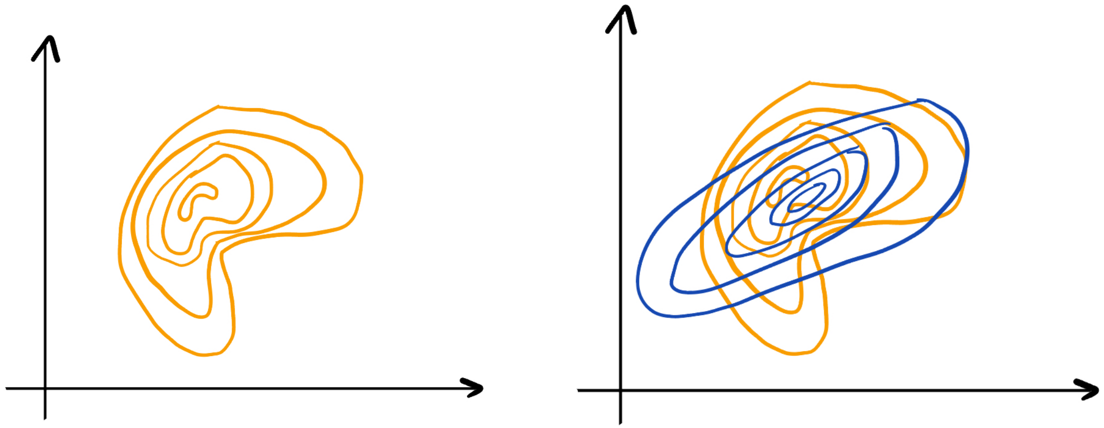
- Sometimes we want to sample from P but cannot do so…
- **Importance sampling** proposes to sample from P', then weight the samples to correct from the fact that we are sampling from the wrong distribution



Target distribution P

Importance sampling

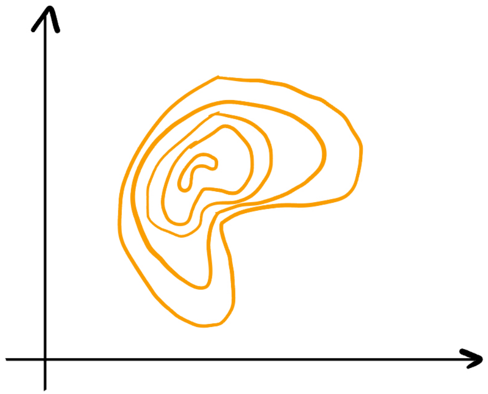
- Sometimes we want to sample from P but cannot do so…
- **Importance sampling** proposes to sample from P', then weight the samples to correct from the fact that we are sampling from the wrong distribution



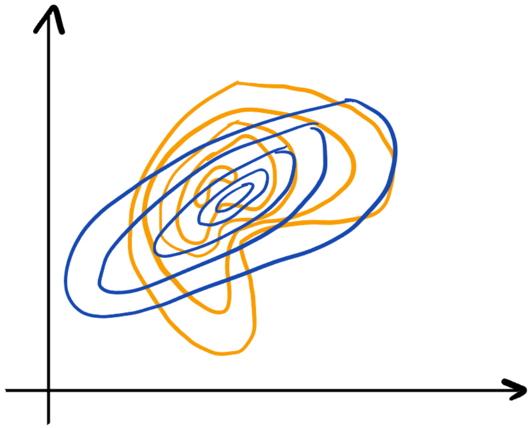
Target distribution P Importance distribution

Importance sampling

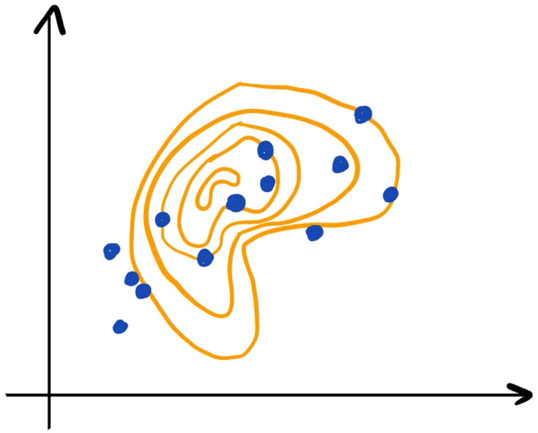
- Sometimes we want to sample from P but cannot do so…
- **Importance sampling** proposes to sample from P', then weight the samples to correct from the fact that we are sampling from the wrong distribution



Target distribution P



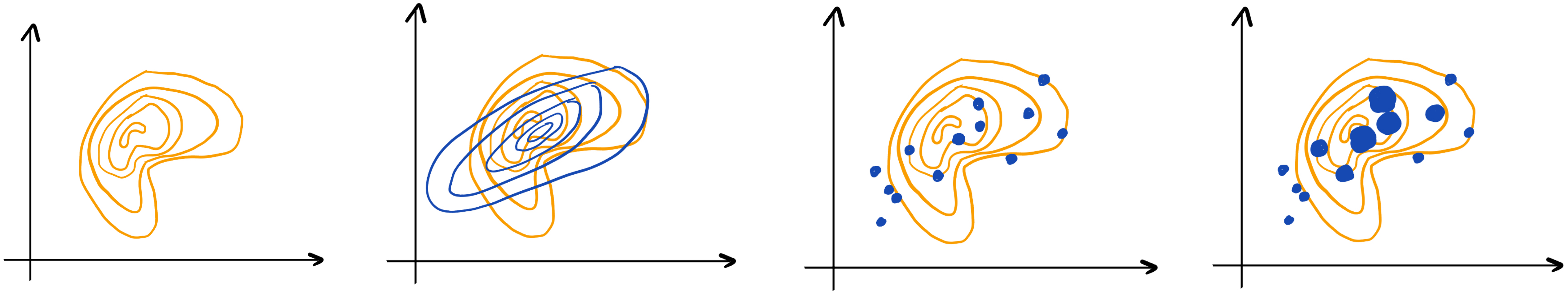
Importance distribution



IIID realisations from importance distribution

Importance sampling

- Sometimes we want to sample from P but cannot do so…
- **Importance sampling** proposes to sample from P', then weight the samples to correct from the fact that we are sampling from the wrong distribution



Target distribution P

Importance distribution

IIID realisations from importance distribution

Weighted samples from target distribution

Importance sampling

- To find the appropriate weights, we can use the following derivation:

$$\mathbb{E}_{X \sim P}[f(X)] = \int f(x)p(x)dx$$

Importance sampling

- To find the appropriate weights, we can use the following derivation:

$$\mathbb{E}_{X\sim P}[f(X)] = \int f(x)p(x)dx = \int f(x)\frac{p(x)}{p'(x)}p'(x)dx$$

Importance sampling

- To find the appropriate weights, we can use the following derivation:

$$\mathbb{E}_{X\sim P}[f(X)] = \int f(x)p(x)dx = \int f(x)\frac{p(x)}{p'(x)}p'(x)dx = \mathbb{E}_{X\sim P'}[w(X)f(X)]$$

$$w(x) = \frac{p(x)}{p'(x)}$$

Importance sampling

- To find the appropriate weights, we can use the following derivation:

$$\mathbb{E}_{X \sim P}[f(X)] = \int f(x)p(x)dx = \int f(x)\frac{p(x)}{p'(x)}p'(x)dx = \mathbb{E}_{X \sim P'}[w(X)f(X)] \approx \frac{1}{n}\sum_{i=1}^{n} w(x_i)f(x_i)$$

$$w(x) = \frac{p(x)}{p'(x)}$$

Importance sampling

- To find the appropriate weights, we can use the following derivation:

$$\mathbb{E}_{X\sim P}[f(X)] = \int f(x)p(x)dx = \int f(x)\frac{p(x)}{p'(x)}p'(x)dx = \mathbb{E}_{X\sim P'}[w(X)f(X)] \approx \frac{1}{n}\sum_{i=1}^{n} w(x_i)f(x_i)$$

$$w(x) = \frac{p(x)}{p'(x)}$$

- This is at the core of many algorithms in computational statistics such as sequential Monte Carlo, variational inference, simulation-based inference, etc..

Importance sampling

- To find the appropriate weights, we can use the following derivation:

$$\mathbb{E}_{X\sim P}[f(X)] = \int f(x)p(x)dx = \int f(x)\frac{p(x)}{p'(x)}p'(x)dx = \mathbb{E}_{X\sim P'}[w(X)f(X)] \approx \frac{1}{n}\sum_{i=1}^{n} w(x_i)f(x_i)$$

$$w(x) = \frac{p(x)}{p'(x)}$$

- This is at the core of many algorithms in computational statistics such as sequential Monte Carlo, variational inference, simulation-based inference, etc..

- **Question:** *“This choice of weights gives us good Monte Carlo estimators, but is it the best possible way to weight our samples?”*

Stein importance sampling

Stage 1: Sample $x_1, \ldots, x_n$ from some proposal P'

Stage 2: $$\arg \min_{w_1,\ldots,w_n \geq 0, \sum_{i=1}^n w_i = 1} \text{KSD}\left(P \,\Big|\Big|\, \sum_{i=1}^n w_i \delta_{x_i}\right)$$

Liu, Q., & Lee, J. D. (2017). Black-box importance sampling. *AISTATS*, 952–961.

Wang, C., Chen, W., Kanagawa, H., & Oates, C. J. (2023). Stein Π-Importance Sampling. *NeurIPS*.

Stein importance sampling

Stage 1: Sample $x_1, \ldots, x_n$ from some proposal P'

Stage 2: $$\arg \min_{w_1,\ldots,w_n \geq 0, \sum_{i=1}^n w_i = 1} \text{KSD}\left(P \middle\| \sum_{i=1}^n w_i \delta_{x_i}\right)$$

Gives a stable set of weights, and exact estimation for constant functions

Liu, Q., & Lee, J. D. (2017). Black-box importance sampling. *AISTATS*, 952–961.

Wang, C., Chen, W., Kanagawa, H., & Oates, C. J. (2023). Stein Π-Importance Sampling. *NeurIPS*.

Stein importance sampling

Stage 1: Sample $x_1, \ldots, x_n$ from some proposal P'

Stage 2: $$\arg\min_{w_1,\ldots,w_n \geq 0,\, \sum_{i=1}^n w_i = 1} \mathrm{KSD}\left(P \,\middle\|\, \sum_{i=1}^n w_i \delta_{x_i}\right)$$

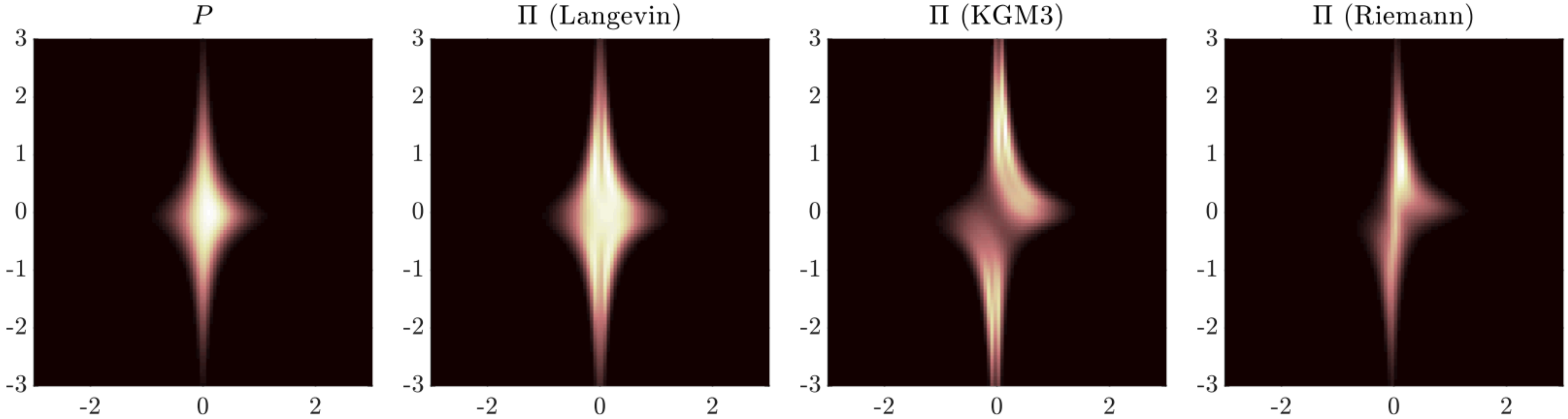
Gives a stable set of weights, and exact estimation for constant functions

- A standard approach for the proposal P' is to use a Markov chain which approximates the target P (or close enough).

Liu, Q., & Lee, J. D. (2017). Black-box importance sampling. *AISTATS*, 952–961.

Wang, C., Chen, W., Kanagawa, H., & Oates, C. J. (2023). Stein Π-Importance Sampling. *NeurIPS*.

Stein importance sampling with different kernels



[Wang, 2023]

Overview: point set approximation with Stein's method

- We have seen many approaches (importance sampling, thinning, deterministic, gradient flows) to getting a good point set approximation of a target:

$$P \approx \sum_{i=1}^{n} w_i \delta_{x_i}$$

Overview: point set approximation with Stein's method

- We have seen many approaches (importance sampling, thinning, deterministic, gradient flows) to getting a good point set approximation of a target:

$$P \approx \sum_{i=1}^{n} w_i \delta_{x_i}$$

→ Having a computable Stein discrepancy which can be used for most P's with unnormalised densities is a real asset here!

Stein’s method as a computational tool

Control variates for Monte Carlo

Monte Carlo methods

- We have already discussed extensively the need for good estimators of:

$$\mathbb{E}_{X \sim P}[f(X)] \approx \frac{1}{n} \sum_{i=1}^{n} f(x_i)$$

Monte Carlo methods

- We have already discussed extensively the need for good estimators of:

$$\mathbb{E}_{X\sim P}[f(X)] \approx \frac{1}{n}\sum_{i=1}^{n} f(x_i)$$

- To know how well this will perform, we can look at the central limit theorem:

$$\sqrt{n}\left(\mathbb{E}_{X\sim P}[f(X)] - \frac{1}{n}\sum_{i=1}^{n} f(x_i)\right) \to N(0,\mathrm{Var}[f])$$

Monte Carlo methods

- We have already discussed extensively the need for good estimators of:

$$\mathbb{E}_{X\sim P}[f(X)] \approx \frac{1}{n}\sum_{i=1}^{n} f(x_i)$$

- To know how well this will perform, we can look at the central limit theorem:

$$\sqrt{n}\left(\mathbb{E}_{X\sim P}[f(X)] - \frac{1}{n}\sum_{i=1}^{n} f(x_i)\right) \to N(0,\mathrm{Var}[f])$$

$$\mathrm{Var}[f] = \mathbb{E}_{X\sim P}\left[(f(X) - \mathbb{E}_{X\sim P}[f(X)])^2\right]$$

Monte Carlo methods

- We have already discussed extensively the need for good estimators of:

$$\mathbb{E}_{X\sim P}[f(X)] \approx \frac{1}{n}\sum_{i=1}^{n} f(x_i)$$

- To know how well this will perform, we can look at the central limit theorem:

$$\sqrt{n}\left(\mathbb{E}_{X\sim P}[f(X)] - \frac{1}{n}\sum_{i=1}^{n} f(x_i)\right) \to N(0,\mathrm{Var}[f])$$

If f is "complicated" where P assigns a lot of mass, this will be large!

$$\mathrm{Var}[f] = \mathbb{E}_{X\sim P}\left[(f(X) - \mathbb{E}_{X\sim P}[f(X)])^2\right]$$

Monte Carlo methods

- We have already discussed extensively the need for good estimators of:

$$\mathbb{E}_{X\sim P}[f(X)] \approx \frac{1}{n}\sum_{i=1}^{n} f(x_i)$$

- To know how well this will perform, we can look at the central limit theorem:

$$\sqrt{n}\left(\mathbb{E}_{X\sim P}[f(X)] - \frac{1}{n}\sum_{i=1}^{n} f(x_i)\right) \to N(0,\text{Var}[f])$$

If f is "complicated" where P assigns a lot of mass, this will be large!

$$\text{Var}[f] = \mathbb{E}_{X\sim P}\left[(f(X) - \mathbb{E}_{X\sim P}[f(X)])^2\right]$$

- The above is for standard Monte Carlo, but similar results hold for MCMC, QMC, etc…

The control variate trick

- Suppose we have a function h for which $\mathbb{E}_{X\sim P}[h(X)] = c$ and c **is known.**

The control variate trick

- Suppose we have a function h for which $\mathbb{E}_{X\sim P}[h(X)] = c$ and c **is known.**
- Then we could rewrite our integral as follows:

$$\mathbb{E}_{X\sim P}[f(X)]$$

The control variate trick

- Suppose we have a function h for which $\mathbb{E}_{X\sim P}[h(X)] = c$ and c **is known.**
- Then we could rewrite our integral as follows:

$$\mathbb{E}_{X\sim P}[f(X)] = \mathbb{E}_{X\sim P}[f(X)] + \mathbb{E}_{X\sim P}[h(X)] - \mathbb{E}_{X\sim P}[h(X)]$$

The control variate trick

- Suppose we have a function h for which $\mathbb{E}_{X\sim P}[h(X)] = c$ and c **is known.**
- Then we could rewrite our integral as follows:

$$\mathbb{E}_{X\sim P}[f(X)] = \mathbb{E}_{X\sim P}[f(X)] + \mathbb{E}_{X\sim P}[h(X)] - \mathbb{E}_{X\sim P}[h(X)] = \mathbb{E}_{X\sim P}[f(X) - h(X)] + c$$

The control variate trick

- Suppose we have a function h for which $\mathbb{E}_{X\sim P}[h(X)] = c$ and c **is known.**
- Then we could rewrite our integral as follows:

$$\mathbb{E}_{X\sim P}[f(X)] = \mathbb{E}_{X\sim P}[f(X)] + \mathbb{E}_{X\sim P}[h(X)] - \mathbb{E}_{X\sim P}[h(X)] = \mathbb{E}_{X\sim P}[f(X) - h(X)] + c$$

- We therefore have a choice of estimator:

<u>Estimator 1</u>: Monte Carlo:

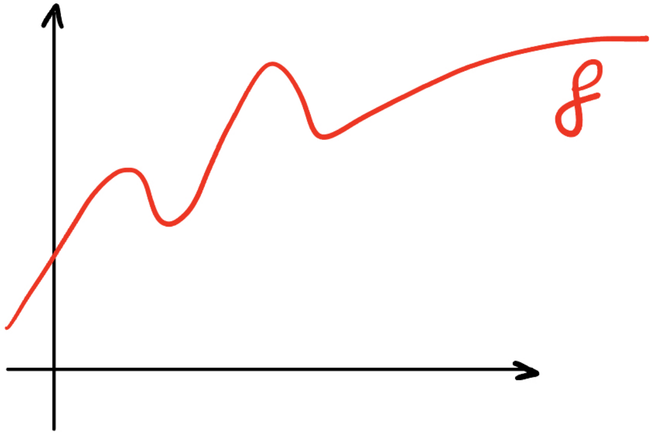
$$\mathbb{E}_{X\sim P}[f(X)] \approx \frac{1}{n}\sum_{i=1}^{n} f(x_i)$$

<u>Estimator 2</u>: Control Variate

$$\mathbb{E}_{X\sim P}[f(X)] \approx \frac{1}{n}\sum_{i=1}^{n} (f(x_i) - h(x_i)) + c$$

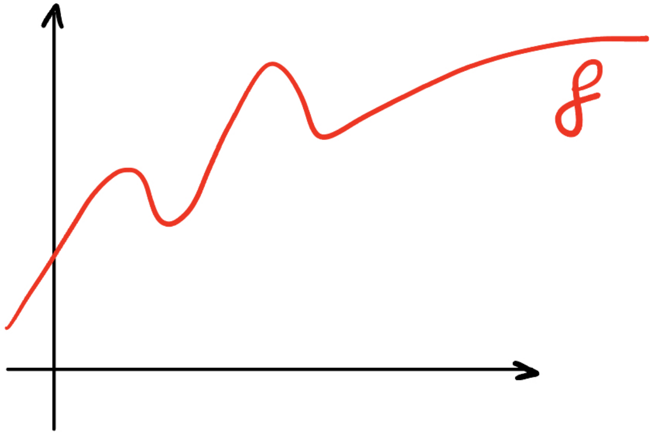
The control variate (CV)!

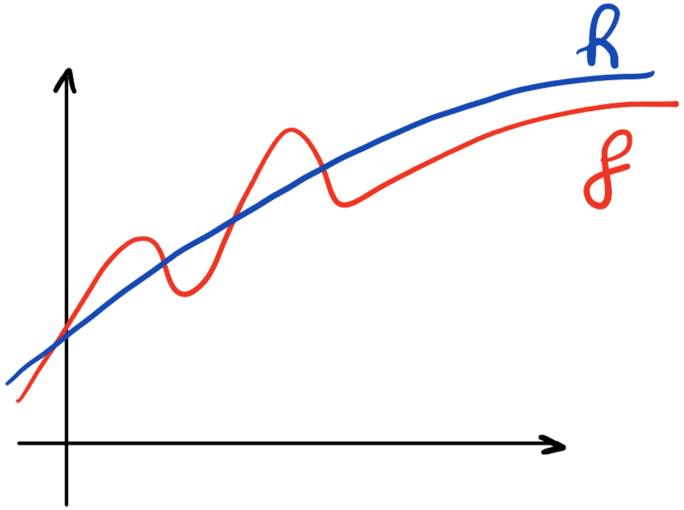
A sketch of control variate estimators



Integrand f

A sketch of control variate estimators

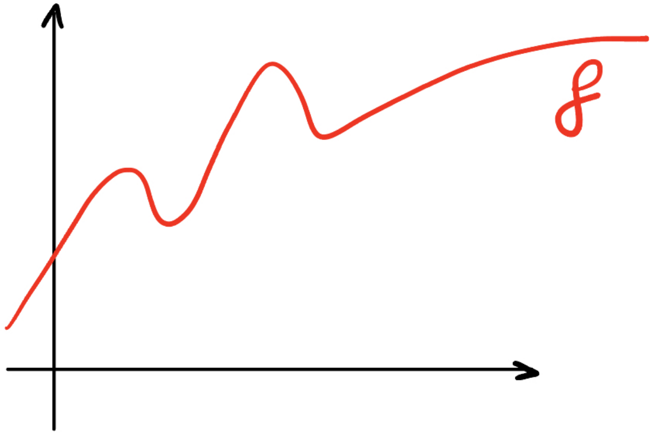




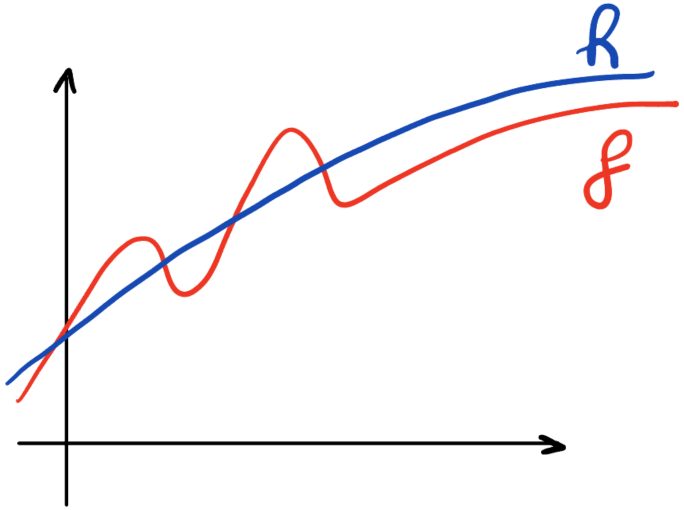
Integrand f

Step 1: Find control variate h with known $\mathbb{E}_{X\sim P}[h(X)]$

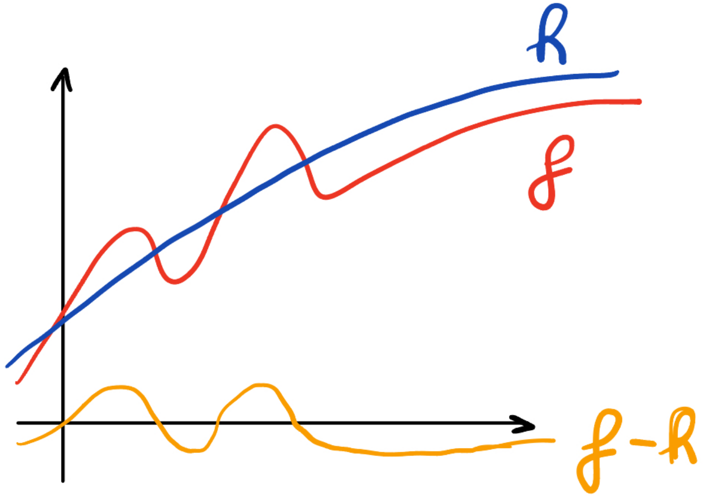
A sketch of control variate estimators



Integrand f

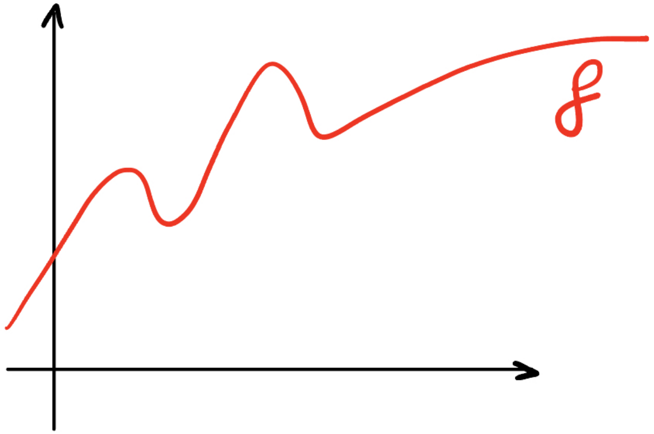


Step 1: Find control variate h with known $\mathbb{E}_{X\sim P}[h(X)]$

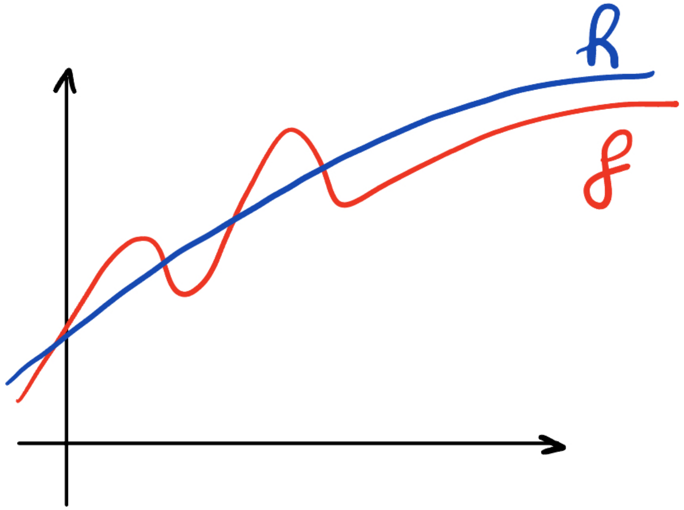


Step 2: Estimate $\mathbb{E}_{X\sim P}[f(X) - h(X)]$ with a Monte Carlo estimator

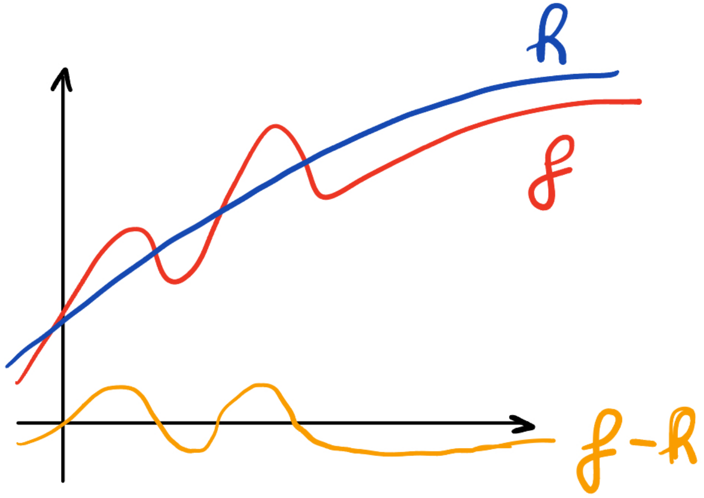
A sketch of control variate estimators



Integrand f



Step 1: Find control variate h with known $\mathbb{E}_{X\sim P}[h(X)]$



Step 2: Estimate $\mathbb{E}_{X\sim P}[f(X) - h(X)]$ with a Monte Carlo estimator

Turns out that if we choose h carefully, then the Monte Carlo estimator of $f - h$ will be much more accurate than the Monte Carlo estimator of f

Existing control variates

- Using the CLT, we see that the accuracy of control variate estimators depend on

$$\text{Var}[f - h]$$

- This leads to a few key questions:

"How do we guarantee that Var$[f - h] \ll$ *Var*$[f]$*?"*

"Can we choose h *to minimise Var*$[f - h]$*?"*

"How do we guarantee that we know the integral of h*?"*

Existing methods

- **Problem:** In general it is really hard to find a function $h : \mathbb{R}^d \to \mathbb{R}$ with known $E_{X \sim P}[h(X)]$.

Existing methods

- **Problem:** In general it is really hard to find a function $h : \mathbb{R}^d \to \mathbb{R}$ with known $E_{X \sim P}[h(X)]$.
- Existing methods focus on simple distributions P such as a Gaussian or a uniform and simple h, such as a polynomial.
- This is severely limiting for computational statistics or machine learning….

Existing methods

- **Problem:** In general it is really hard to find a function $h : \mathbb{R}^d \to \mathbb{R}$ with known $E_{X \sim P}[h(X)]$.
- Existing methods focus on simple distributions P such as a Gaussian or a uniform and simple h, such as a polynomial.
- This is severely limiting for computational statistics or machine learning….

***Question:** What are we supposed to do when P is more complicated?*

Stein control variates

- Given the focus of this course, it should be obvious that we can pick:

$$h_\theta(x) = \mathcal{S}_P[g_\theta](x) + \theta_0$$

Stein control variates

- Given the focus of this course, it should be obvious that we can pick:

$$h_\theta(x) = \mathcal{S}_P[g_\theta](x) + \theta_0 \longrightarrow \mathbb{E}_{X\sim P}[h_\theta(X)] = \mathbb{E}_{X\sim P}[\mathcal{S}_P[g_\theta](X)] + \theta_0 = \theta_0$$

Stein control variates

- Given the focus of this course, it should be obvious that we can pick:

$$h_\theta(x) = \mathcal{S}_P[g_\theta](x) + \theta_0 \longrightarrow \mathbb{E}_{X\sim P}[h_\theta(X)] = \mathbb{E}_{X\sim P}[\mathcal{S}_P[g_\theta](X)] + \theta_0 = \theta_0$$

- $\{g_\theta : \theta \in \Theta\}$ can be a family of polynomials, neural networks, an RKHS, etc… so long as this family is a subset of the corresponding Stein class $\mathcal{G}_P$!

Stein control variates

- Given the focus of this course, it should be obvious that we can pick:

$$h_\theta(x) = \mathcal{S}_P[g_\theta](x) + \theta_0 \longrightarrow \mathbb{E}_{X\sim P}[h_\theta(X)] = \mathbb{E}_{X\sim P}[\mathcal{S}_P[g_\theta](X)] + \theta_0 = \theta_0$$

- $\{g_\theta : \theta \in \Theta\}$ can be a family of polynomials, neural networks, an RKHS, etc… so long as this family is a subset of the corresponding Stein class $\mathcal{G}_P$!

- Initial work in Bayesian computation mostly used the equivalent of polynomial-based Stein control variates without realising they were using Stein!

Mira, A., Solgi, R., & Imparato, D. (2013). Zero variance Markov chain Monte Carlo for Bayesian estimators. *Statistics and Computing*, *23*(5), 653–662.

Papamarkou, T., Mira, A., & Girolami, M. (2014). Zero variance differential geometric Markov chain Monte Carlo algorithms. *Bayesian Analysis*, *9*(1), 97–128.

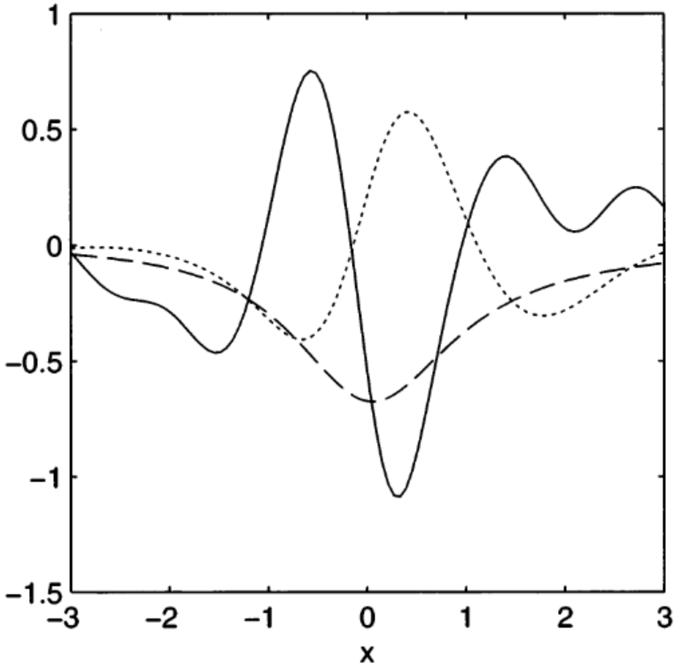
Elements of Stein RKHS

- Recall that we can take an RKHS $\mathscr{H}_k$ and create a new one by applying a Stein operator to functions in the space:

$$\mathcal{S}_P[g](x), \qquad g \in \mathscr{H}_k^d$$

- This leads to the RKHS with kernel k_P given by:

$$k_p(x, x') = \mathcal{S}_P^x \mathcal{S}_P^{x'}[k](x, x')$$



[Oates et al 2017, JRSSB]

Oates, C. J., Girolami, M., & Chopin, N. (2017). Control functionals for Monte Carlo integration. *Journal of the Royal Statistical Society B: Statistical Methodology*, *79*(3), 695–718.

Oates, C. J., Cockayne, J., Briol, F.-X., & Girolami, M. (2019). Convergence rates for a class of estimators based on Stein's identity. *Bernoulli*, *25*(2), 1141–1159.

South, L. F., Karvonen, T., Nemeth, C., Girolami, M., & Oates, C. J. (2022). Semi-exact control functionals from Sard's method. *Biometrika*, *109*, 351–367.

Stein Neural Networks

- We can also take our favourite (sufficiently smooth) neural network $g_\theta : \mathbb{R}^d \to \mathbb{R}^d$ and apply a Stein operator to the output.
- To use the language in this field, we can add a **“Stein layer”**.

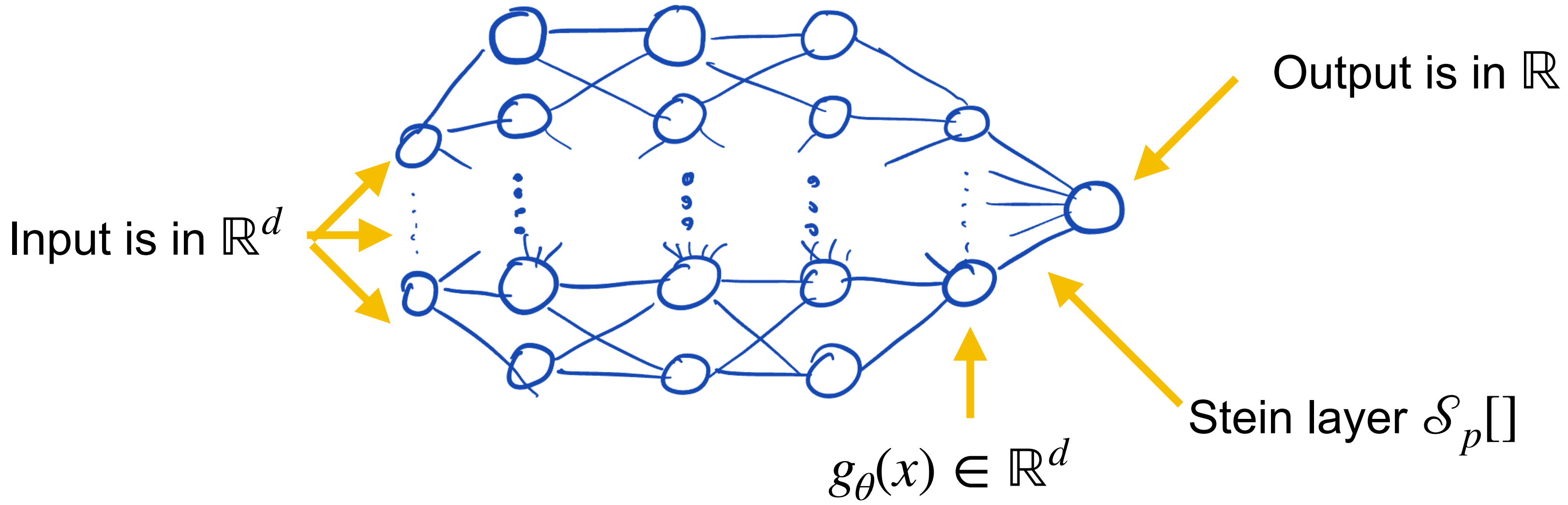
Si, S., Oates, C. J., Duncan, A. B., Carin, L., & Briol, F.-X. (2021). Scalable control variates for Monte Carlo methods via stochastic optimization. *Proceedings of the 14th Conference on Monte Carlo and Quasi-Monte Carlo Methods*.

Sun, Z., Oates, C. J., & Briol, F.-X. (2023). Meta-learning control variates: Variance reduction with limited data. *UAI*, 2047–2057.

Ott, K., Tiemann, M., Hennig, P., & Briol, F.-X. (2023). Bayesian numerical integration with neural networks. *UAI*, 1606–1617.

Stein Neural Networks

- We can also take our favourite (sufficiently smooth) neural network $g_\theta : \mathbb{R}^d \to \mathbb{R}^d$ and apply a Stein operator to the output.
- To use the language in this field, we can add a **"Stein layer"**.



Si, S., Oates, C. J., Duncan, A. B., Carin, L., & Briol, F.-X. (2021). Scalable control variates for Monte Carlo methods via stochastic optimization. *Proceedings of the 14th Conference on Monte Carlo and Quasi-Monte Carlo Methods*.

Sun, Z., Oates, C. J., & Briol, F.-X. (2023). Meta-learning control variates: Variance reduction with limited data. *UAI*, 2047–2057.

Ott, K., Tiemann, M., Hennig, P., & Briol, F.-X. (2023). Bayesian numerical integration with neural networks. *UAI*, 1606–1617.

Stein CVs as empirical risk minimisation

- Clearly a natural objective to choose our control variate is:

$$J(\theta) = \mathrm{Var}[f - h_\theta] = \mathrm{Var}[f - \mathcal{S}_P[g_\theta](x) - \theta_0]$$

Stein CVs as empirical risk minimisation

- Clearly a natural objective to choose our control variate is:

$$J(\theta) = \text{Var}[f - h_\theta] = \text{Var}[f - \mathcal{S}_P[g_\theta](x) - \theta_0]$$

- Since this objective is intractable, we can approximate it with samples:

$$J_m(\theta) = \widehat{\text{Var}_m}[f - h_\theta]$$

Stein CVs as empirical risk minimisation

- Clearly a natural objective to choose our control variate is:

$$J(\theta) = \text{Var}[f - h_\theta] = \text{Var}[f - \mathcal{S}_P[g_\theta](x) - \theta_0]$$

- Since this objective is intractable, we can approximate it with samples:

$$J_m(\theta) = \widehat{\text{Var}_m}[f - h_\theta] = \frac{1}{m}\sum_{j=1}^{m}(f(x_j) - h_\theta(x_j))^2$$

Stein CVs as empirical risk minimisation

- Clearly a natural objective to choose our control variate is:

$$J(\theta) = \mathrm{Var}[f - h_\theta] = \mathrm{Var}[f - \mathcal{S}_P[g_\theta](x) - \theta_0]$$

- Since this objective is intractable, we can approximate it with samples:

$$J_m(\theta) = \widehat{\mathrm{Var}_m}[f - h_\theta] = \frac{1}{m}\sum_{j=1}^{m}(f(x_j) - h_\theta(x_j))^2$$

- We then choose our control variate as follows:

$$\hat{\theta}_m = \arg\min_{\theta\in\Theta} J_m(\theta) \longrightarrow h_{\hat{\theta}_m}(x)$$

Linear Stein CVs

- We note that Stein operators are usually **linear operators,** meaning that

$$\theta \mapsto h_\theta(x)$$

will be linear so long as $\theta \mapsto g_\theta(x)$ is also linear!

Linear Stein CVs

- We note that Stein operators are usually **linear operators,** meaning that

$$\theta \mapsto h_\theta(x)$$

  will be linear so long as $\theta \mapsto g_\theta(x)$ is also linear!

- This is the case for polynomials or for kernels, but not for neural networks.

Linear Stein CVs

- We note that Stein operators are usually **linear operators,** meaning that

$$\theta \mapsto h_\theta(x)$$

will be linear so long as $\theta \mapsto g_\theta(x)$ is also linear!

- This is the case for polynomials or for kernels, but not for neural networks.

- The great advantage of linear Stein CVs is that $\theta \mapsto J_m(\theta)$ becomes a quadratic function in θ and **can hence be solved through a linear system** (we are essentially doing least squares)!

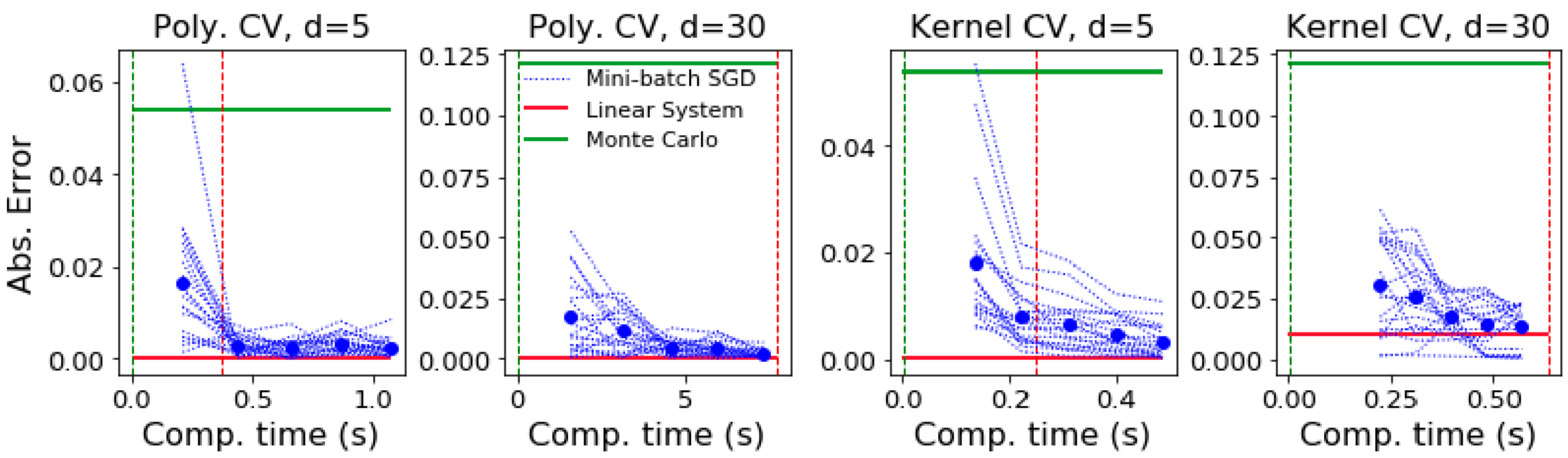
Stochastic optimisation for linear Stein CVs: a toy problem

- Of course another approach is to use gradient-based optimisation, such as stochastic gradient descent…

$$f(x) = x_1 + \ldots + x_d$$

$$P = \mathcal{N}(0, I_d)$$

Half of the samples were used for learning the CV, the other half for the estimator



Si, S., Oates, C. J., Duncan, A. B., Carin, L., & Briol, F.-X. (2021). Scalable control variates for Monte Carlo methods via stochastic optimization. *Proceedings of the 14th Conference on Monte Carlo and Quasi-Monte Carlo Methods*.

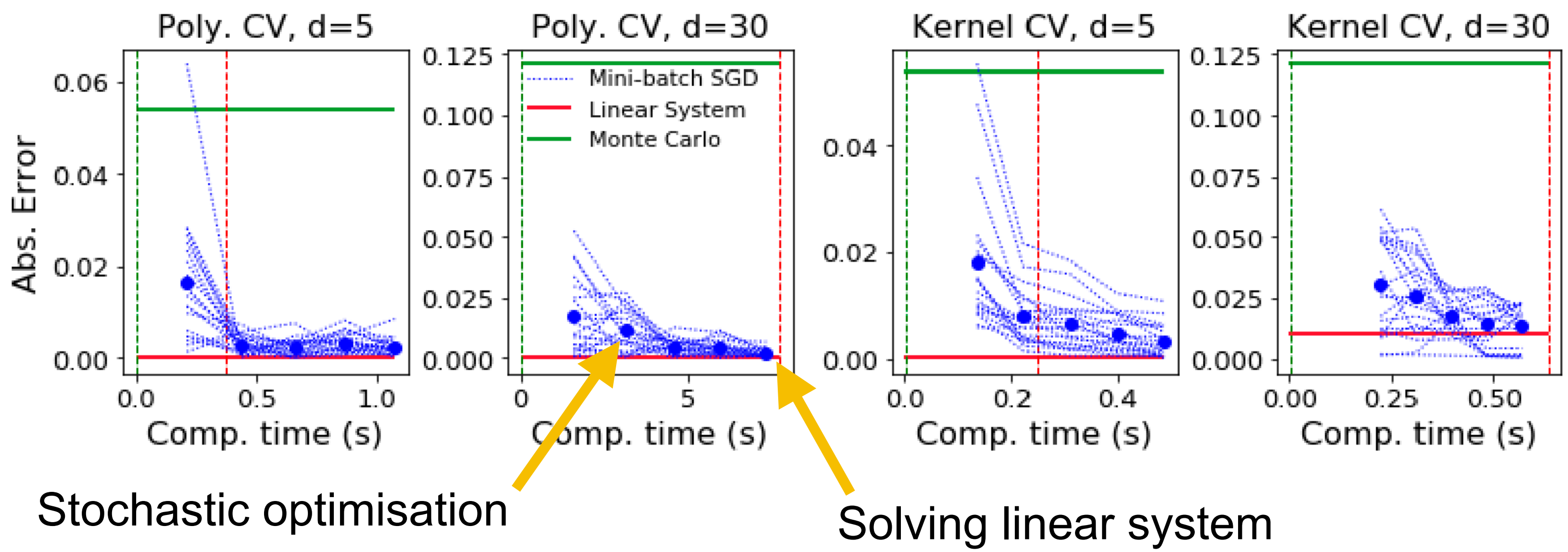
Stochastic optimisation for linear Stein CVs: a toy problem

- Of course another approach is to use gradient-based optimisation, such as stochastic gradient descent…

$$f(x) = x_1 + \ldots + x_d$$

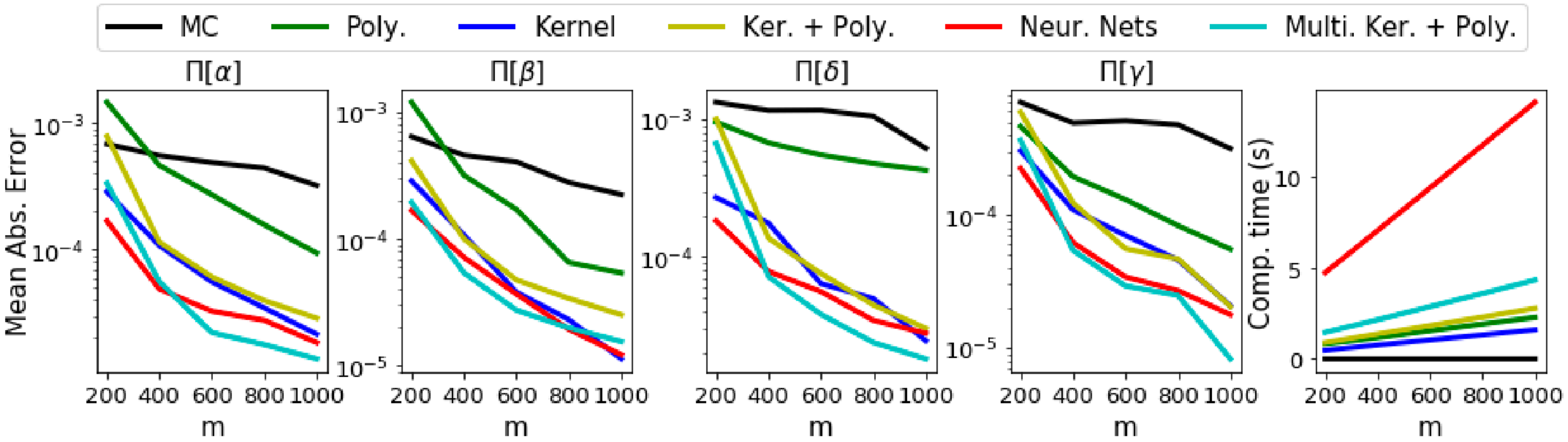
$$P = \mathcal{N}(0, I_d)$$

Half of the samples were used for learning the CV, the other half for the estimator



Si, S., Oates, C. J., Duncan, A. B., Carin, L., & Briol, F.-X. (2021). Scalable control variates for Monte Carlo methods via stochastic optimization. *Proceedings of the 14th Conference on Monte Carlo and Quasi-Monte Carlo Methods*.

Posterior inference for ODE system



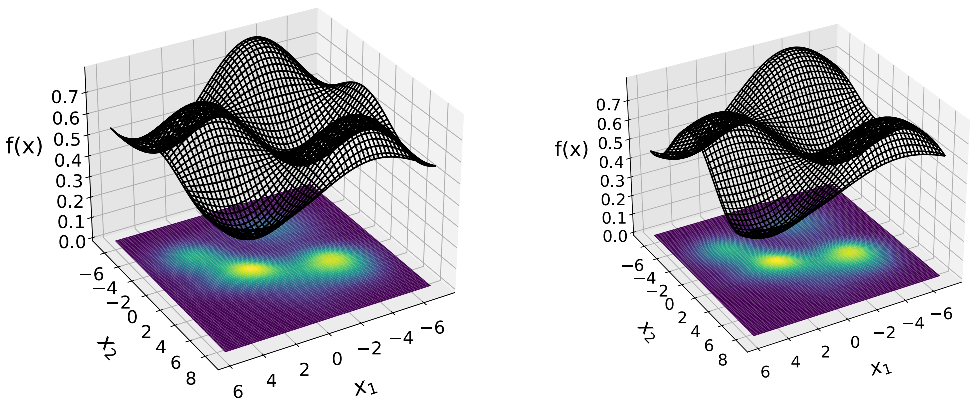
Computing expectations under the posterior for $(\alpha, \beta, \delta, \gamma)$ given some observations of the following Lotka-Volterra ODE system:

(Half of the samples were used for learning the CV, the other half for the estimator)

$$\dot{x} = \alpha x - \beta x y \qquad \dot{y} = \delta x y - \gamma y$$

Si, S., Oates, C. J., Duncan, A. B., Carin, L., & Briol, F.-X. (2021). Scalable control variates for Monte Carlo methods via stochastic optimization. *Proceedings of the 14th Conference on Monte Carlo and Quasi-Monte Carlo Methods*.

Multiple related integrals



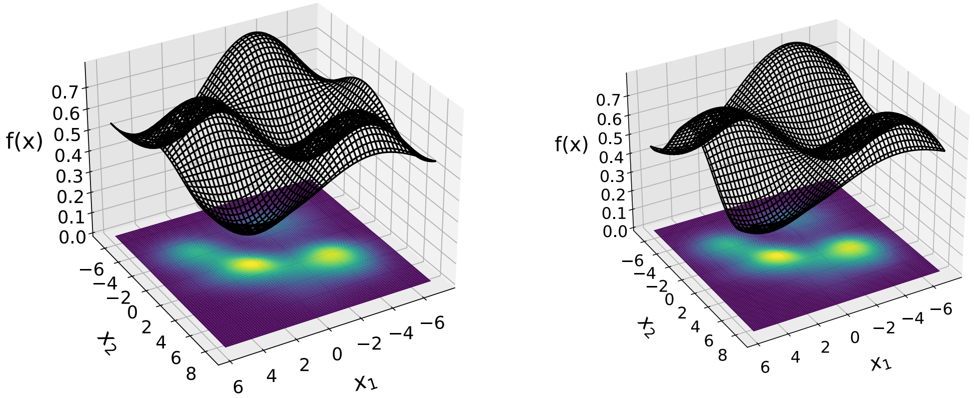
- In some situations, we have to estimate several integrals either sequentially or simultaneously:

$$\mathbb{E}_{X\sim P_1}[f_1(X)], \ldots, \mathbb{E}_{X\sim P_T}[f_T(X)]$$

Sun, Z., Barp, A., & Briol, F.-X. (2023). Vector-valued control variates. *ICML*, 32819–32846.

Sun, Z., Oates, C. J., & Briol, F.-X. (2023). Meta-learning control variates: Variance reduction with limited data. *UAI*, 2047–2057.

Multiple related integrals



- In some situations, we have to estimate several integrals either sequentially or simultaneously:

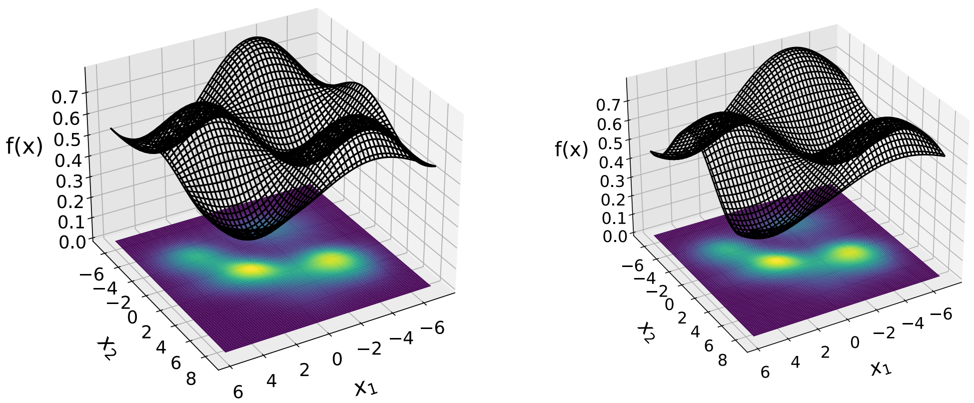
$$\mathbb{E}_{X\sim P_1}[f_1(X)], \ldots, \mathbb{E}_{X\sim P_T}[f_T(X)]$$

- These could be estimated separately, but sharing information across tasks can significantly improve the accuracy.

Sun, Z., Barp, A., & Briol, F.-X. (2023). Vector-valued control variates. *ICML*, 32819–32846.

Sun, Z., Oates, C. J., & Briol, F.-X. (2023). Meta-learning control variates: Variance reduction with limited data. *UAI*, 2047–2057.

Multiple related integrals



- In some situations, we have to estimate several integrals either sequentially or simultaneously:

$$\mathbb{E}_{X\sim P_1}[f_1(X)], \ldots, \mathbb{E}_{X\sim P_T}[f_T(X)]$$

- These could be estimated separately, but sharing information across tasks can significantly improve the accuracy.

- Thankfully Stein's method can be extended to vector-valued functions to create control variates suitable for tackling this task!

Sun, Z., Barp, A., & Briol, F.-X. (2023). Vector-valued control variates. *ICML*, 32819–32846.

Sun, Z., Oates, C. J., & Briol, F.-X. (2023). Meta-learning control variates: Variance reduction with limited data. *UAI*, 2047–2057.

Overview: numerical integration with Stein's method

- The accuracy of Monte Carlo methods can be significantly improved through control variates, but finding a good control variate can be very hard.

Overview: numerical integration with Stein’s method

- The accuracy of Monte Carlo methods can be significantly improved through control variates, but finding a good control variate can be very hard.

→ Stein’s method allows us to create very flexible classes of control variates for a very broad variety of applications!

Stein’s method as a computational tool

Beyond Euclidean domains

A computational tool beyond Euclidean spaces…

- Recall our favourite Stein operator:

$$\mathcal{T}[g](x) := \langle \nabla_x \log p(x), g(x) \rangle + \langle \nabla, g(x) \rangle$$

A computational tool beyond Euclidean spaces…

- Recall our favourite Stein operator:

$$\mathcal{T}[g](x) := \langle \nabla_x \log p(x), g(x) \rangle + \langle \nabla, g(x) \rangle$$

- This is only valid when the domain/data space is $\mathcal{X} = \mathbb{R}^d$…!

A computational tool beyond Euclidean spaces…

- Recall our favourite Stein operator:

$$\mathcal{T}[g](x) := \langle \nabla_x \log p(x), g(x) \rangle + \langle \nabla, g(x) \rangle$$

- This is only valid when the domain/data space is $\mathcal{X} = \mathbb{R}^d$…!
- But often we want to do statistics with data which is in $\mathbb{R}^d$; e.g. categorical data, count data, manifold-valued data, functional data…

A computational tool beyond Euclidean spaces…

- Recall our favourite Stein operator:

$$\mathcal{T}[g](x) := \langle \nabla_x \log p(x), g(x) \rangle + \langle \nabla, g(x) \rangle$$

- This is only valid when the domain/data space is $\mathcal{X} = \mathbb{R}^d$…!
- But often we want to do statistics with data which is in $\mathbb{R}^d$; e.g. categorical data, count data, manifold-valued data, functional data…

None of the tools we have seen so far work….

Stein on bounded subsets of Euclidean space

- The defining property of the Langevin Stein operator is:

$$\mathbb{E}_{X \sim P}\left[\mathcal{T}[g](X)\right] = \int_{\mathbb{R}^d} \mathcal{T}[g](x)p(x)dx = 0$$

Stein on bounded subsets of Euclidean space

- The defining property of the Langevin Stein operator is:

$$\mathbb{E}_{X\sim P}\left[\mathcal{T}[g](X)\right] = \int_{\mathbb{R}^d} \mathcal{T}[g](x)p(x)dx = 0$$

- But what if instead we have a model defined only on positive values:

$$\int_{\mathbb{R}^d_+} \mathcal{T}[g](x)p(x)dx \neq 0$$

Stein on bounded subsets of Euclidean space

- The defining property of the Langevin Stein operator is:

$$\mathbb{E}_{X\sim P}\left[\mathcal{T}[g](X)\right] = \int_{\mathbb{R}^d} \mathcal{T}[g](x)p(x)dx = 0$$

- But what if instead we have a model defined only on positive values:

$$\int_{\mathbb{R}^d_+} \mathcal{T}[g](x)p(x)dx \neq 0$$

- There are plenty of cases where our models/data does not have full support, but where $\mathcal{X} \subset \mathbb{R}^d$ and this is a **strict subset**.

Stein on bounded subsets of Euclidean space

- A straightforward solution in this case is to use a modified RKHS as the Stein space:

$$\tilde{k}(x, x') = \delta(x)k(x, x')\delta(x')$$

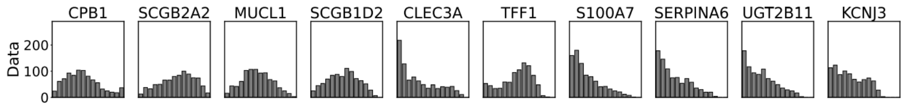
- Where we enforce that the kernel vanishes on the boundary:

$$\delta(x) = 0 \quad \text{for} \quad x \in \partial\mathcal{X}$$

Oates, C. J., Cockayne, J., Briol, F.-X., & Girolami, M. (2019). Convergence rates for a class of estimators based on Stein's identity. *Bernoulli*, *25*(2), 1141–1159.

Williams, D. J., & Liu, S. (2023). Approximate Stein Classes for truncated density estimation. *International Conference on Machine Learning*.

Stein on discrete spaces



[Matsubara et al., 2024+]

$$\mathcal{X} = \mathcal{S}_1 \times \ldots \times \mathcal{S}_d$$

$\mathcal{S}_i$ is a countable ordered set

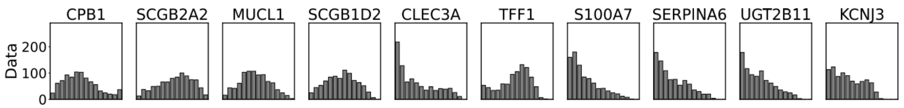
$$S_P[g](x) = \left\langle \frac{\nabla^- p(x)}{p(x)}, g(x) \right\rangle + \langle \nabla^+, g(x) \rangle$$

Yang, J., Liu, Q., Rao, V., & Neville, J. (2018). Goodness-of-fit testing for discrete distributions via Stein discrepancy. *ICML*.

Shi, J., Zhou, Y., Hwang, J., Titsias, M. K., & Mackey, L. (2022). Gradient estimation with discrete Stein operators. *NeurIPS*.

Matsubara, T., Knoblauch, J., Briol, F.-X., & Oates, C. J. (2024+). Generalised Bayesian inference for discrete intractable likelihood. *JASA (to appear)*.

Stein on discrete spaces



[Matsubara et al., 2024+]

$$\mathscr{X} = \mathscr{S}_1 \times \ldots \times \mathscr{S}_d$$

$\mathscr{S}_i$ is a countable ordered set

$$S_P[g](x) = \left\langle \frac{\nabla^- p(x)}{p(x)}, g(x) \right\rangle + \langle \nabla^+, g(x) \rangle$$

$$\nabla^- g(x) = \begin{bmatrix} g(x^{1-}) - g(x) \\ \cdots \\ g(x^{d-}) - g(x) \end{bmatrix} \qquad \nabla^+ g(x) = \begin{bmatrix} g(x^{1+}) - g(x) \\ \cdots \\ g(x^{d+}) - g(x) \end{bmatrix}$$

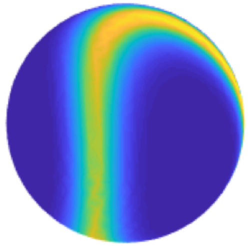
Yang, J., Liu, Q., Rao, V., & Neville, J. (2018). Goodness-of-fit testing for discrete distributions via Stein discrepancy. *ICML*.

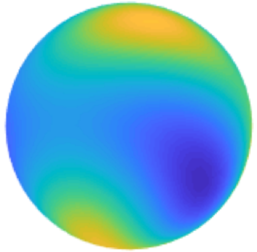
Shi, J., Zhou, Y., Hwang, J., Titsias, M. K., & Mackey, L. (2022). Gradient estimation with discrete Stein operators. *NeurIPS*.

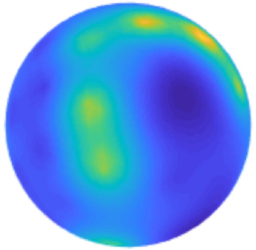
Matsubara, T., Knoblauch, J., Briol, F.-X., & Oates, C. J. (2024+). Generalised Bayesian inference for discrete intractable likelihood. *JASA (to appear)*.

Stein on manifolds

[Barp et al., 2022]







- Sometimes we also want to consider data on manifolds (e.g. spheres, positive definite matrices, etc…)

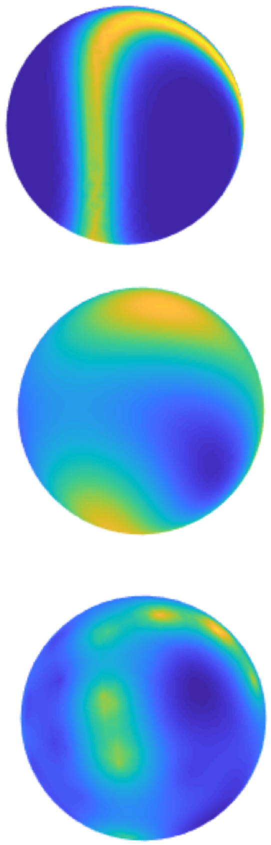
Xu, W., & Matsuda, T. (2020). A Stein goodness-of-fit test for directional distributions. *AISTATS*.

Xu, W., & Matsuda, T. (2021). Interpretable Stein goodness-of-fit tests on Riemannian manifolds. *ICML*.

Barp, A., Oates, C. J., Porcu, E., & Girolami, M. (2022). A Riemannian–Stein kernel method. *Bernoulli*, *28*(4), 2181–2208.

Stein on manifolds

[Barp et al., 2022]



- Sometimes we also want to consider data on manifolds (e.g. spheres, positive definite matrices, etc…)

- Once again the generator approach comes to the rescue: we just need a Markov process defined on this space….

- There are abundant choices available from physics and computational chemistry literatures!

Xu, W., & Matsuda, T. (2020). A Stein goodness-of-fit test for directional distributions. *AISTATS*.

Xu, W., & Matsuda, T. (2021). Interpretable Stein goodness-of-fit tests on Riemannian manifolds. *ICML*.

Barp, A., Oates, C. J., Porcu, E., & Girolami, M. (2022). A Riemannian–Stein kernel method. *Bernoulli*, *28*(4), 2181–2208.

Stein on function spaces

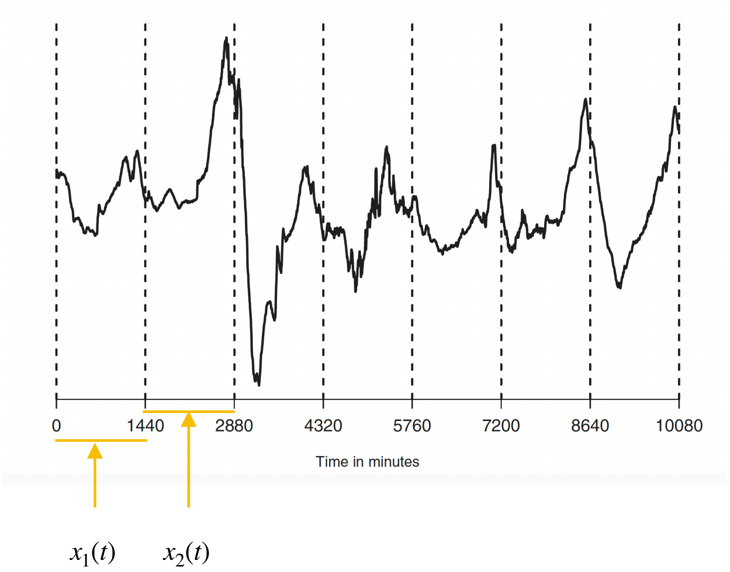
- $\mathscr{X}$ is a space of function (e.g. time series, spatial measurements, etc…).

Wynne, G., Kasprzak, M., & Duncan, A. B. (2024+). A spectral representation of kernel Stein discrepancy with application to goodness-of-fit tests for measures on infinite dimensional Hilbert spaces. *Bernoulli (to appear)*.

Stein on function spaces

- $\mathscr{X}$ is a space of function (e.g. time series, spatial measurements, etc…).

Magnetic field in Honolulu

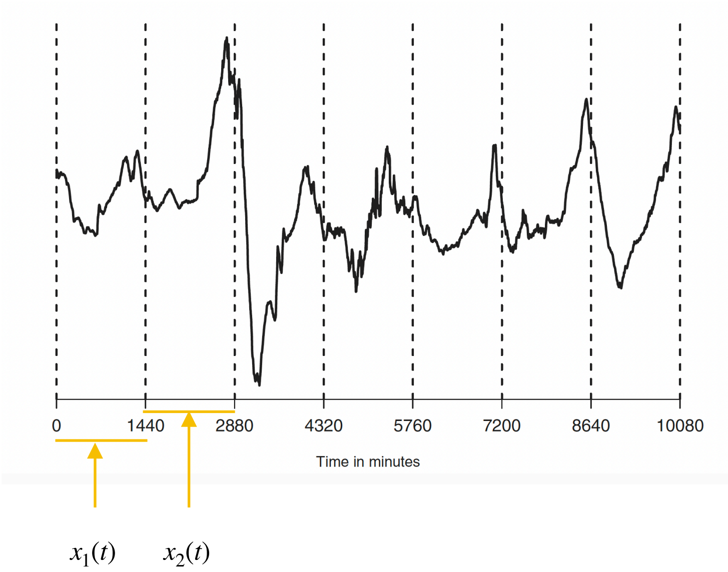


Wynne, G., Kasprzak, M., & Duncan, A. B. (2024+). A spectral representation of kernel Stein discrepancy with application to goodness-of-fit tests for measures on infinite dimensional Hilbert spaces. *Bernoulli (to appear)*.

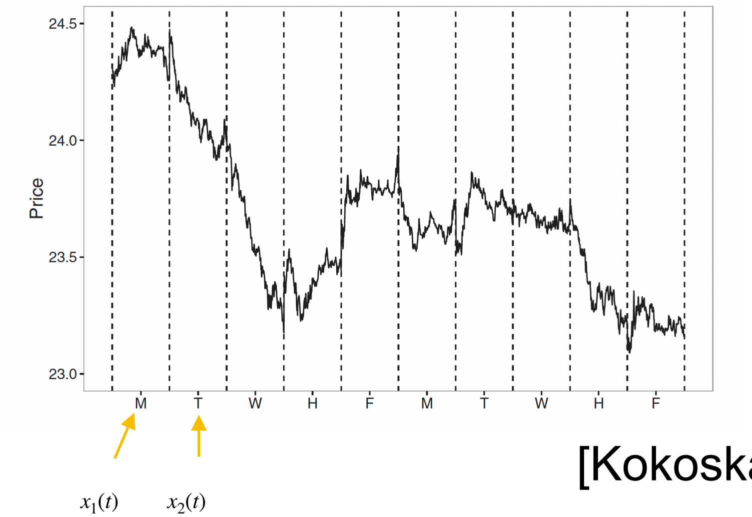
Stein on function spaces

- $\mathscr{X}$ is a space of function (e.g. time series, spatial measurements, etc…).

Magnetic field in Honolulu



Microsoft stock price in May 2006



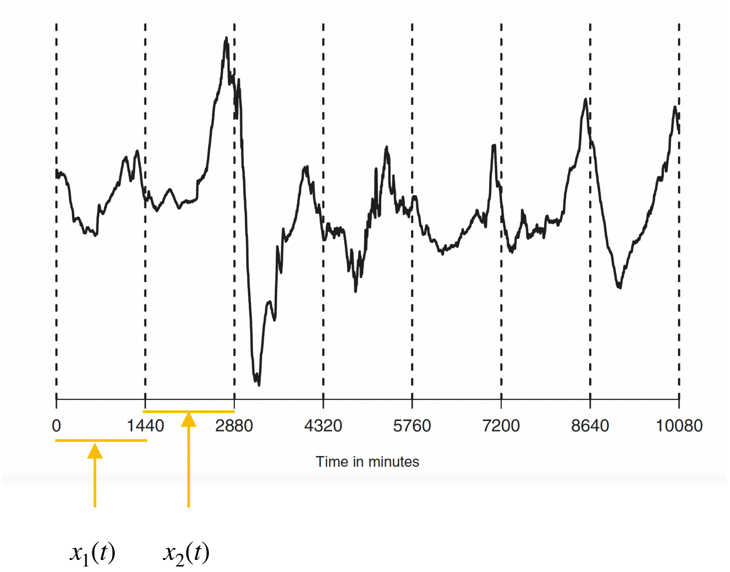
[Kokoska, 2017]

Wynne, G., Kasprzak, M., & Duncan, A. B. (2024+). A spectral representation of kernel Stein discrepancy with application to goodness-of-fit tests for measures on infinite dimensional Hilbert spaces. *Bernoulli (to appear)*.

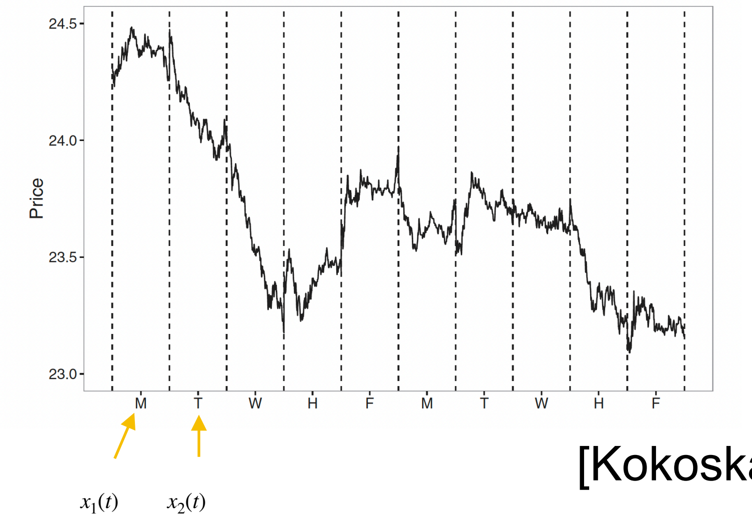
Stein on function spaces

- $\mathscr{X}$ is a space of function (e.g. time series, spatial measurements, etc…).

Magnetic field in Honolulu



Microsoft stock price in May 2006



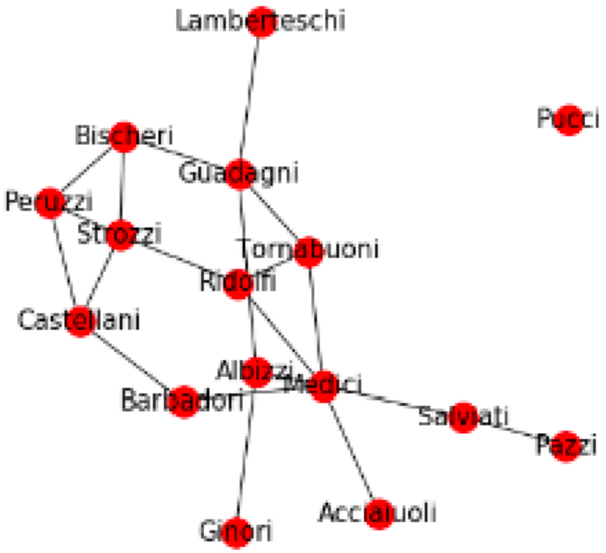
[Kokoska, 2017]

- Once again the generator approach comes to the rescue - we can use the generator of a Wiener process with a carefully selected kernel.

Wynne, G., Kasprzak, M., & Duncan, A. B. (2024+). A spectral representation of kernel Stein discrepancy with application to goodness-of-fit tests for measures on infinite dimensional Hilbert spaces. *Bernoulli (to appear)*.

Stein on graphs

- A lot less straightforward to write on a slide, but is based on the generator approach of Barbour…
- The exact operator is based on Glauber dynamics which allows you to simulate on the space of graphs.



[Xu & Reinert 2022]

Xu, W., & Reinert, G. (2021). A Stein goodness of fit test for exponential random graph models. *AISTATS*.

Xu, W., & Reinert, G. (2022). AgraSSt: Approximate graph Stein statistics for interpretable assessment of implicit graph generators. *NeurIPS*.

Stein's method as a computational tool

The end

Outline (updated)

- What is Stein’s method, and why should you care…
- Computational tools based on Stein’s method.
- Some nice (new) algorithms!

Conclusions/Take-Away

- Stein’s method gives us a new **characterisation of distributions** which is particularly **convenient from a computational viewpoint**!

Conclusions/Take-Away

- Stein’s method gives us a new **characterisation of distributions** which is particularly **convenient from a computational viewpoint**!

- The most useful tool is the **kernel Stein discrepancy (KSD)**, a discrepancy which is computable in most settings of interest in computational statistics and machine learning!

Conclusions/Take-Away

- Stein's method gives us a new **characterisation of distributions** which is particularly **convenient from a computational viewpoint**!
- The most useful tool is the **kernel Stein discrepancy (KSD)**, a discrepancy which is computable in most settings of interest in computational statistics and machine learning!
- Stein's method has now touched most areas in these fields…!

Conclusions/Take-Away

- Stein's method gives us a new **characterisation of distributions** which is particularly **convenient from a computational viewpoint**!
- The most useful tool is the **kernel Stein discrepancy (KSD)**, a discrepancy which is computable in most settings of interest in computational statistics and machine learning!
- Stein's method has now touched most areas in these fields…!

Statistical Science
2023, Vol. 38, No. 1, 120–139
https://doi.org/10.1214/22-STS863

Stein's Method Meets Computational Statistics: A Review of Some Recent Developments