"AI goes MAD"

IFT, June 2022

BRYAN ZALDIVAR (IFIC)

"AI goes MAD"

IFT, June 2022

BRYAN ZALDIVAR (IFIC)

A Review on Modern Bayesian Inference

"AI goes MAD"

IFT, June 2022

BRYAN ZALDIVAR (IFIC)

"AI goes MAD"

IFT, June 2022

BRYAN ZALDIVAR (IFIC)

Some open research lines in B.I.

INTRODUCTION

- What is Bayesian Inference?

- What is Bayesian Inference?

  A way to update the "degree of believe" of your hypotheses after seeing some data

- What is Bayesian Inference?

  A way to update the "degree of believe" of your hypotheses after seeing some data

Hyp. A : e.g. A = Higgs mass

$p(A) \Rightarrow$ prior dist.

$p(\text{Data} \mid A) \Rightarrow$ Likelihood of data

- What is Bayesian Inference?

A way to update the "degree of believe" of your hypotheses after seeing some data

$$\underbrace{p(A|\text{Data})}_{\text{Posterior dist.}} = \frac{p(\text{Data}|A)\,p(A)}{p(\text{Data})}$$

Hyp. A : e.g. A = Higgs mass

$p(A) \Rightarrow$ prior dist.

$p(\text{Data}|A) \Rightarrow$ Likehlhood of data

- What is Bayesian Inference?

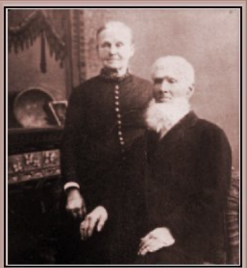
A way to update the "degree of believe" of your hypotheses after seeing some data

$$\underbrace{p(A|\text{Data})}_{\text{Posterior dist.}} = \frac{p(\text{Data}|A)\,p(A)}{p(\text{Data})}$$

Hyp. A : e.g. A = Higgs mass

$p(A) \Rightarrow$ prior dist.

$p(\text{Data}|A) \Rightarrow$ Likelihood of data



Thomas BAYES and wife Catherine HALEY Bayes

- What is Bayesian Inference?

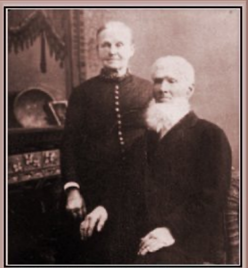
A way to update the "degree of believe" of your hypotheses after seeing some data

$$\underbrace{p(A|\text{Data})}_{\text{Posterior dist.}} = \frac{p(\text{Data}|A)\,p(A)}{p(\text{Data})}$$

Hyp. A : e.g. A = Higgs mass

$p(A) \Rightarrow$ prior dist.

$p(\text{Data}|A) \Rightarrow$ Likelihood of data



Thomas BAYES and wife Catherine HALEY Bayes

As opposed to "Frequentist" approach

$$p(A) = \frac{\#\text{ of occurrences of } A}{\text{Total } \#\text{ of trials}}$$

A : an observable (not a hypothesis)

INTRODUCTION

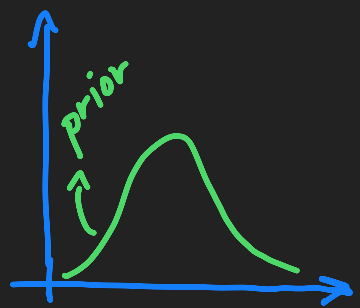
- What is it used for?

- What is it used for?

 1) Making inference on physical parameters

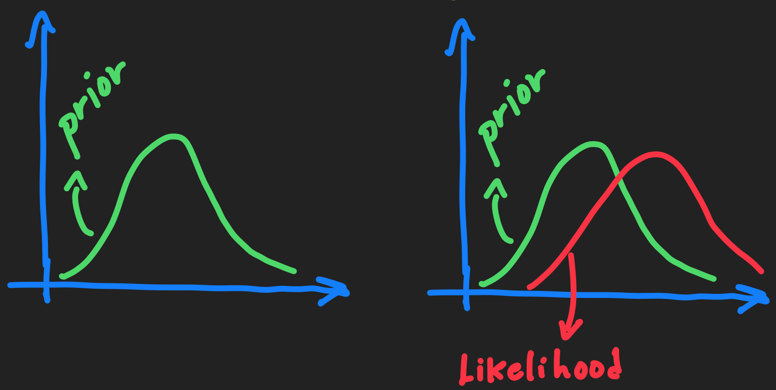
- What is it used for ?

1) Making inference on physical parameters



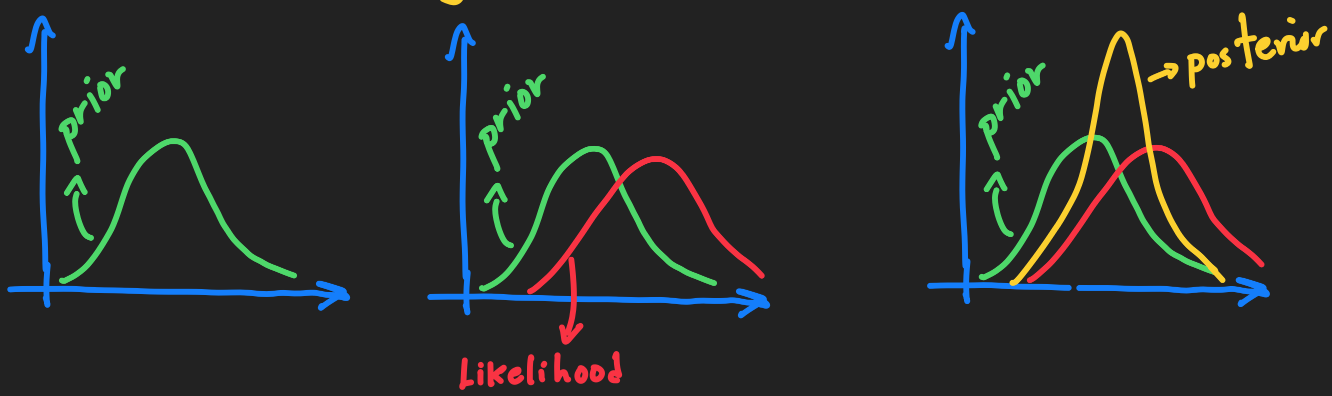
- What is it used for ?

1) Making inference on physical parameters



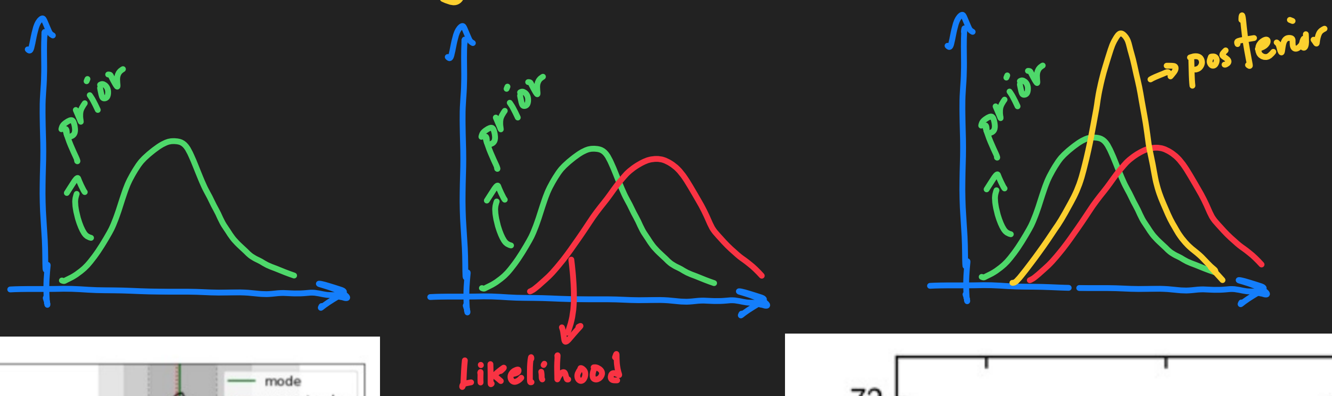
- What is it used for ?

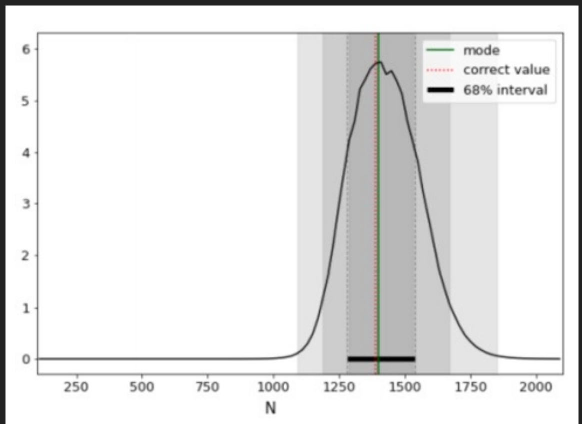
1) Making inference on physical parameters



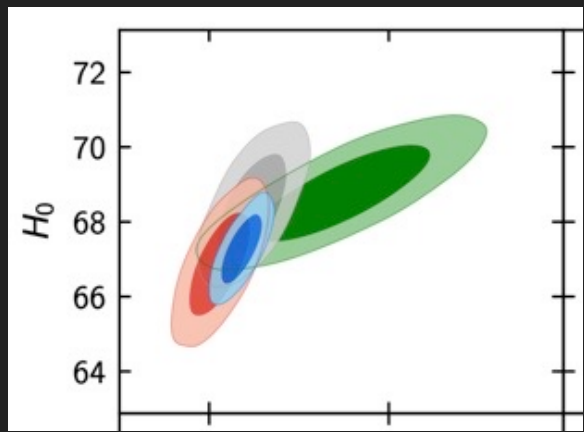
- What is it used for?

1) Making inference on physical parameters





CREDITS: Dimitriou et al, (to appear)

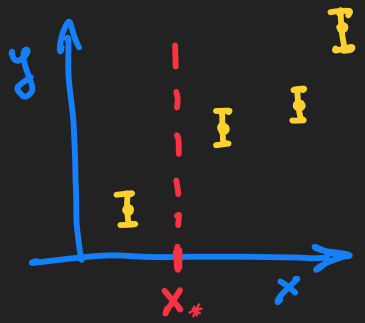


Planck Collaboration 1807.06209

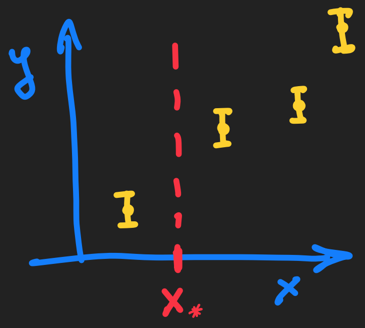
INTRODUCTION

2) Obtaining predictions with uncertainties

2) Obtaining predictions with uncertainties

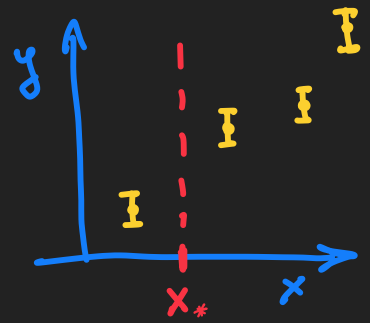


2) Obtaining predictions with uncertainties

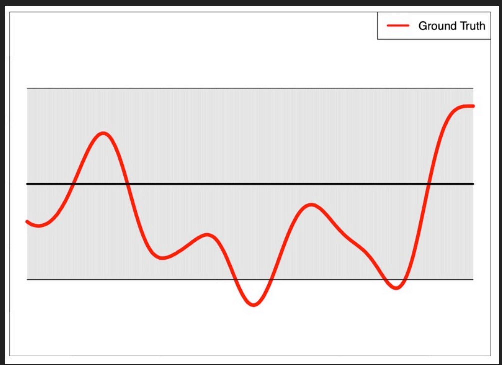


$$P(y_* | x_*, \text{Data}) = \int \underbrace{d\vec{\theta}}_{\text{model parameters}} \begin{pmatrix} \text{Likelihood} \\ \text{of } y_* | x_*, \vec{\theta} \end{pmatrix} \cdot \begin{pmatrix} \text{Posterior of} \\ \vec{\theta} \,|\, \text{Data} \end{pmatrix}$$

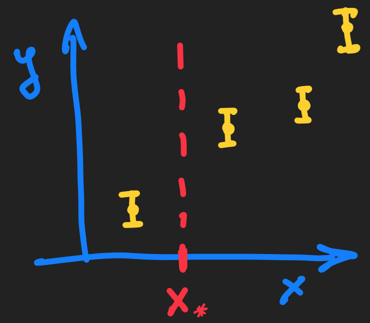
2) Obtaining predictions with uncertainties



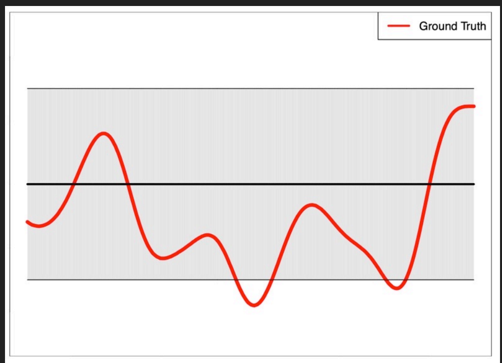
$$P(y_* \mid x_*, \text{Data}) = \int \overset{\text{model parameters}}{d\vec{\theta}} \begin{pmatrix} \text{Likelihood} \\ \text{of } y_* \mid x_*, \vec{\theta} \end{pmatrix} \cdot \begin{pmatrix} \text{Posterior of} \\ \vec{\theta} \mid \text{Data} \end{pmatrix}$$

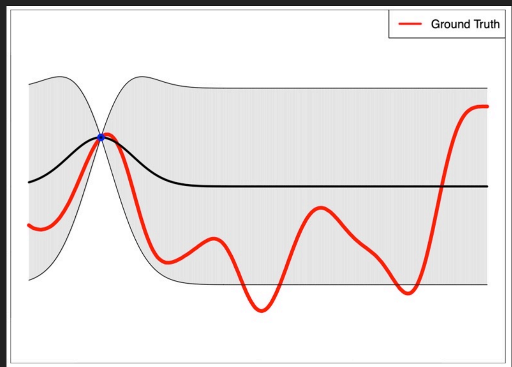


2) Obtaining predictions with uncertainties

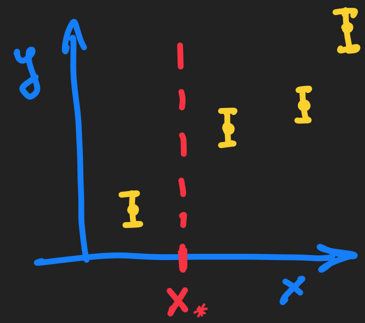


$$P(y_* \mid x_*, \text{Data}) = \int \overset{\text{model parameters}}{d\vec{\theta}} \left(\begin{array}{c}\text{Likelihood} \\ \text{of } y_* \mid x_*, \vec{\theta}\end{array}\right) \cdot \left(\begin{array}{c}\text{Posterior of} \\ \vec{\theta} \mid \text{Data}\end{array}\right)$$

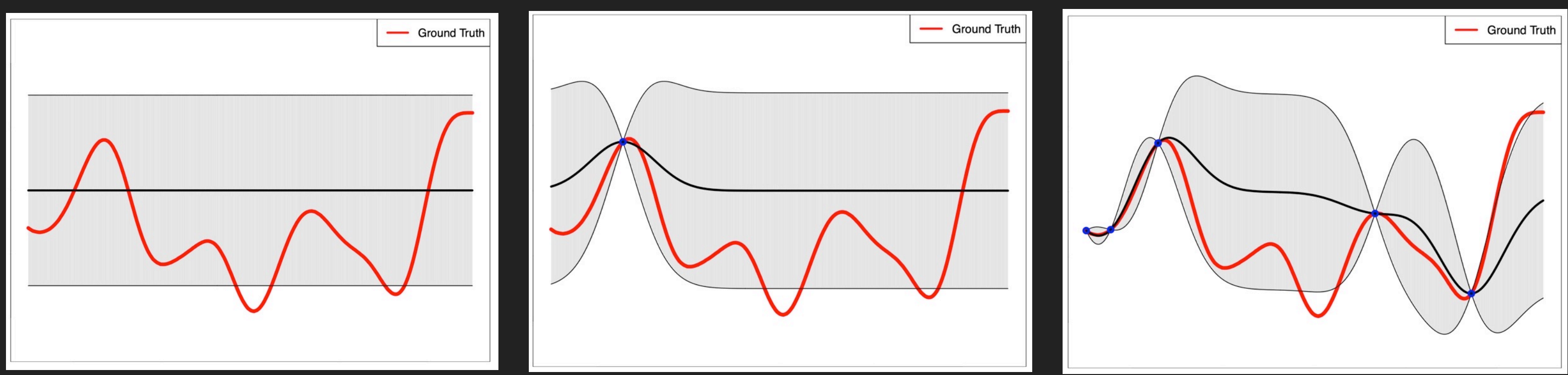




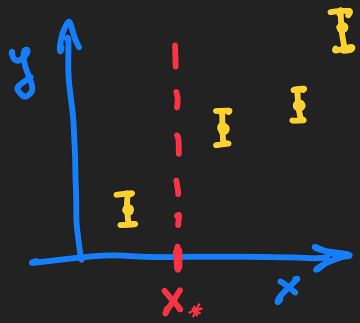
2) Obtaining predictions with uncertainties



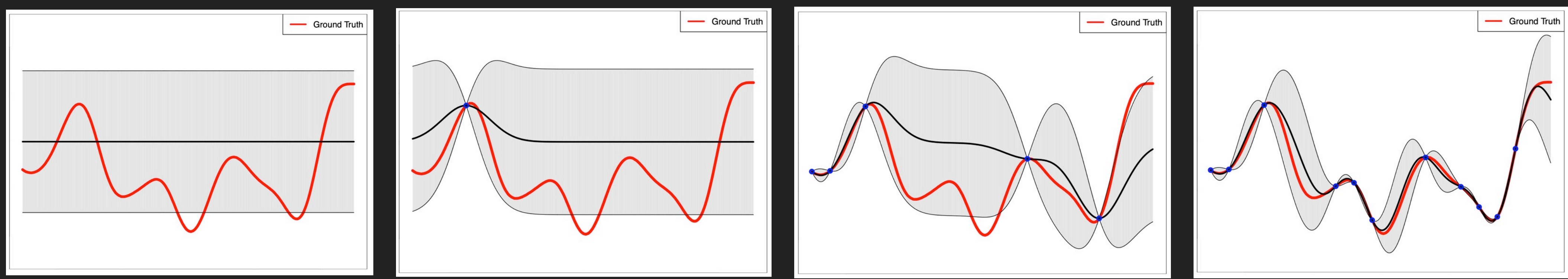
$$P(y_*|x_*,\text{Data}) = \int \overset{\text{model parameters}}{d\vec{\theta}}\,\left(\begin{array}{c}\text{Likelihood}\\ \text{of } y_*|x_*,\vec{\theta}\end{array}\right)\cdot\left(\begin{array}{c}\text{Posterior of}\\ \vec{\theta}\,|\,\text{Data}\end{array}\right)$$



2) Obtaining predictions with uncertainties



$$P(y_* \mid x_*, \text{Data}) = \int \overset{\text{model parameters}}{d\vec{\theta}} \begin{pmatrix} \text{Likelihood} \\ \text{of } y_* \mid x_*, \vec{\theta} \end{pmatrix} \cdot \begin{pmatrix} \text{Posterior of} \\ \vec{\theta} \mid \text{Data} \end{pmatrix}$$



METHODS

METHODS

- Bayesian Inference is computationally challenging!

- Bayesian Inference is computationally challenging!

$$p(\vec{\theta}\,|\,\mathrm{Data}) = p(\mathrm{Data}\,|\,\vec{\theta})\, p(\vec{\theta}) \,/\, p(\mathrm{Data})$$

- Bayesian Inference is computationally challenging!

$$p(\vec{\theta}\,|\,\mathrm{Data}) = p(\mathrm{Data}\,|\,\vec{\theta})\, p(\vec{\theta}) \,/\, p(\mathrm{Data})$$

$$p(\mathrm{Data}) = \int d\vec{\theta}\ p(\mathrm{Data}\,|\,\vec{\theta})\ p(\vec{\theta})$$

- Bayesian Inference is computationally challenging!

$$p(\vec{\theta}\,|\,\text{Data}) = p(\text{Data}\,|\,\vec{\theta})\, p(\vec{\theta}) \,/\, p(\text{Data})$$

$$p(\text{Data}) = \int d\vec{\theta}\;\; p(\text{Data}\,|\,\vec{\theta})\;\, p(\vec{\theta}) \quad \Big\} \text{ typically intractable}$$

- Bayesian Inference is computationally challenging!

$$p(\vec{\theta}\,|\,\text{Data}) = p(\text{Data}\,|\,\vec{\theta})\, p(\vec{\theta}) \,/\, p(\text{Data}) \Longrightarrow \text{Needs to be approximated}$$

$$p(\text{Data}) = \int d\vec{\theta}\; p(\text{Data}\,|\,\vec{\theta})\; p(\vec{\theta}) \quad \Big\} \text{ typically intractable}$$

- Bayesian Inference is computationally challenging!

$$p(\vec{\theta}|\text{Data}) = p(\text{Data}|\vec{\theta})\, p(\vec{\theta}) / p(\text{Data})$$ → Needs to be approximated

$$p(\text{Data}) = \int d\vec{\theta}\; p(\text{Data}|\vec{\theta})\; p(\vec{\theta})$$ } typically intractable

- Approximate methods

- Bayesian Inference is computationally challenging!

$$p(\vec{\theta}\,|\,\text{Data}) = p(\text{Data}\,|\,\vec{\theta})\, p(\vec{\theta}) / p(\text{Data})$$ → Needs to be approximated

$$p(\text{Data}) = \int d\vec{\theta}\; p(\text{Data}\,|\,\vec{\theta})\; p(\vec{\theta})$$ } typically intractable

- Approximate methods

1. Sampling methods

e.g. MCMC,
ABC, ...

- Bayesian Inference is computationally challenging!

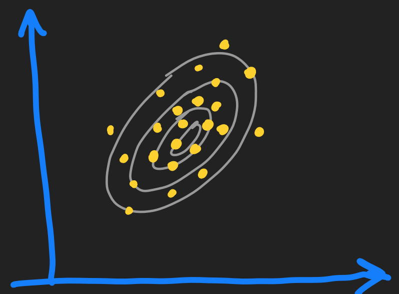
$$p(\vec{\theta}|\text{Data}) = p(\text{Data}|\vec{\theta})\, p(\vec{\theta}) \,/\, p(\text{Data}) \longrightarrow \text{Needs to be approximated}$$

$$p(\text{Data}) = \int d\vec{\theta}\; p(\text{Data}|\vec{\theta})\; p(\vec{\theta}) \quad \Big\} \text{ typically intractable}$$

- Approximate methods

① Sampling methods

e.g. MCMC, ABC, ...



- Bayesian Inference is computationally challenging!

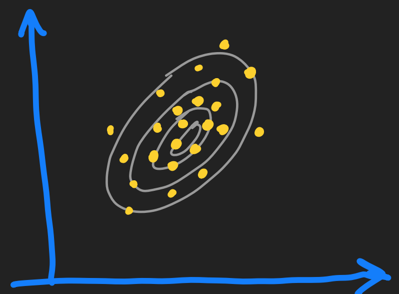
$$p(\vec{\theta}|\mathrm{Data}) = p(\mathrm{Data}|\vec{\theta})\, p(\vec{\theta}) \,/\, p(\mathrm{Data})$$ → Needs to be approximated

$$p(\mathrm{Data}) = \int d\vec{\theta}\; p(\mathrm{Data}|\vec{\theta})\; p(\vec{\theta})$$ } typically intractable

- Approximate methods

① Sampling methods

e.g. MCMC, ABC, ...



② Optimization methods

e.g. Laplace, Variational Inference, N. ratio-estimation, ...

- Bayesian Inference is computationally challenging!

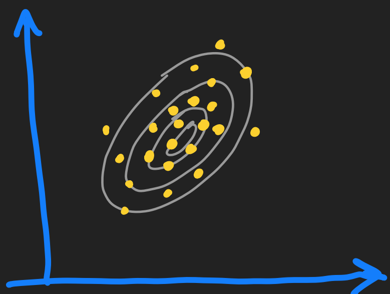
$$p(\vec{\theta}\,|\,\text{Data}) = p(\text{Data}\,|\,\vec{\theta})\, p(\vec{\theta}) \,/\, p(\text{Data}) \longrightarrow \text{Needs to be approximated}$$

$$p(\text{Data}) = \int d\vec{\theta}\; p(\text{Data}\,|\,\vec{\theta})\; p(\vec{\theta}) \quad \Big\}\ \text{typically intractable}$$

- Approximate methods

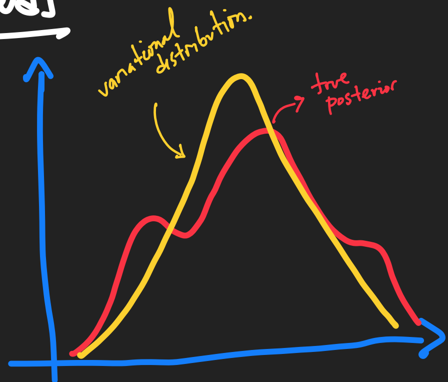
① Sampling methods

e.g. MCMC,
ABC, ...



② Optimization methods

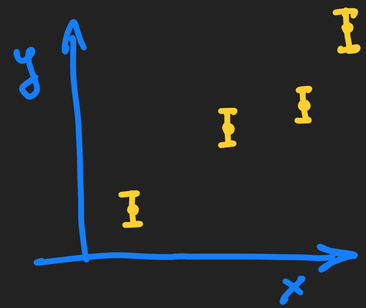
e.g. Laplace,
Variational Inference,
N. ratio-estimation, ...



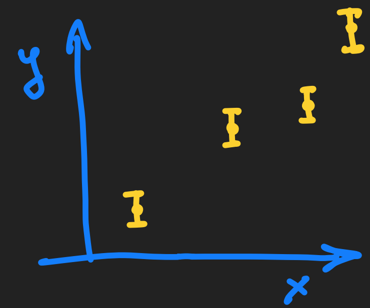
VARIATIONAL INFERENCE

- Data $\{x_i, y_i\}_{i=1}^{N}$

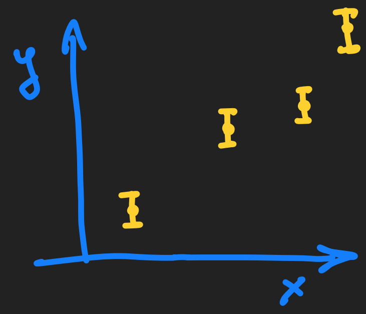
- Data $\{x_i, y_i\}_{i=1}^{N}$



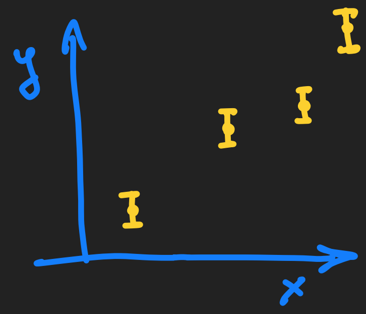
- Data $\{x_i, y_i\}_{i=1}^{N}$
- Fitting function $f(x;\vec{\theta})$



- Data $\{x_i, y_i\}_{i=1}^{N}$
- Fitting function $f(x; \vec{\theta})$
- Likelihood $p(\vec{y} \mid \vec{\theta}) = \prod_{i=1}^{N} N(y_i \mid f(x_i, \vec{\theta}), \underbrace{\sigma_i}_{\text{known uncertainty}})$

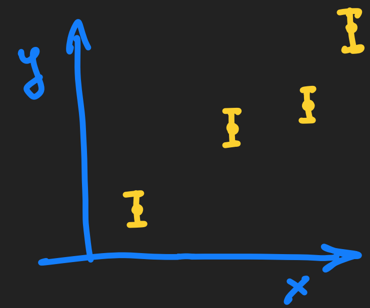


- Data $\{x_i, y_i\}_{i=1}^{N}$
- Fitting function $f(x; \vec{\theta})$



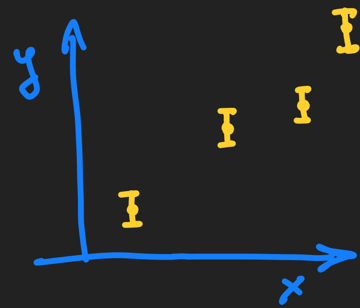
- Likelihood $p(\vec{y}\,|\,\vec{\theta}) = \prod_{i=1}^{N} N(y_i \,|\, f(x_i, \vec{\theta}), \underbrace{\sigma_i}_{\hookrightarrow \text{known uncertainty}})$
- Prior distrib. for $\vec{\theta}$: $p(\vec{\theta}) \rightarrow$ e.g. also Gaussian

- Data $\{x_i, y_i\}_{i=1}^{N}$
- Fitting function $f(x; \vec{\theta})$



- Likelihood $p(\vec{y}|\vec{\theta}) = \prod_{i=1}^{N} \mathcal{N}(y_i | f(x_i, \vec{\theta}), \underbrace{\sigma_i}_{\text{known uncertainty}})$
- Prior distrib. for $\vec{\theta}$: $p(\vec{\theta}) \rightarrow$ e.g. also Gaussian
- Posterior $p(\vec{\theta}|\vec{y})$ not Gaussian in general $\rightarrow$ Typically intractable

- Data $\{x_i, y_i\}_{i=1}^{N}$
- Fitting function $f(x; \vec{\theta})$

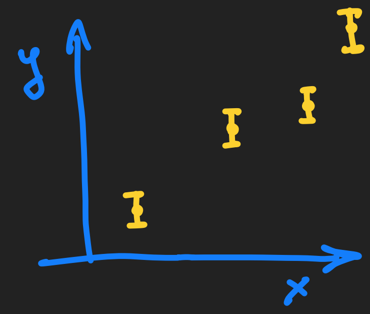


- Likelihood $p(\vec{y}|\vec{\theta}) = \prod_{i=1}^{N} N(y_i | f(x_i, \vec{\theta}), \underbrace{\sigma_i}_{\text{known uncertainty}})$

- Prior distrib. for $\vec{\theta}$: $p(\vec{\theta})$ → e.g. also Gaussian
- Posterior $p(\vec{\theta}|\vec{y})$ not Gaussian in general → Typically intractable
- $p(\vec{\theta}|\vec{y}) \approx q(\vec{\theta}|\vec{\eta})$
  - Tractable
  - Expressive enough

VARIATIONAL INFERENCE

- Data $\{x_i, y_i\}_{i=1}^{N}$ • Fitting function $f(x;\vec{\theta})$



- Likelihood $p(\vec{y}|\vec{\theta}) = \prod_{i=1}^{N} N(y_i | f(x_i, \vec{\theta}), \underbrace{\sigma_i}_{\text{known uncertainty}})$

- Prior distrib. for $\vec{\theta}$: $p(\vec{\theta})$ → e.g. also Gaussian

- Posterior $p(\vec{\theta}|\vec{y})$ not Gaussian in general → Typically intractable

- $p(\vec{\theta}|\vec{y}) \approx q(\vec{\theta}|\vec{\eta})$
  - Tractable
  - Expressive enough

  Need a procedure to optimize parameters $\vec{\eta}$ s.t. this approx. is as good as possible

- Standard procedure: Minimize the KL divergence:

$$KL[q_{\vec{\eta}} \| p] = \int d\vec{\theta} \cdot q_{\vec{\eta}} \cdot \ln\left(\frac{q_{\vec{\eta}}}{p}\right) \geq 0$$

- Standard procedure: Minimize the KL divergence:

$$KL[q_{\vec{\eta}} \,|\, p] = \int d\vec{\theta} \cdot q_{\vec{\eta}} \cdot \ln\left(\frac{q_{\vec{\eta}}}{p}\right) \geq 0$$

↳ true posterior

- Standard procedure: Minimize the KL divergence:

$$KL[q_{\vec{\eta}} \| p] = \int d\vec{\theta} \cdot q_{\vec{\eta}} \cdot \ln\left(\frac{q_{\vec{\eta}}}{p}\right) \geq 0$$

proposed approximation

true posterior

- Standard procedure: Minimize the KL divergence:

$$KL[q_{\vec{\eta}} \| p] = \int d\vec{\theta} \cdot q_{\vec{\eta}} \cdot \ln\left(\frac{q_{\vec{\eta}}}{p}\right) \geq 0$$

proposed approximation

true posterior

Useless since it depends on p

- Standard procedure: Minimize the KL divergence:

$$\underbrace{KL[q_{\vec{\eta}}}_{\text{proposed approximation}} \| \underbrace{p}_{\text{true posterior}}] = \int d\vec{\theta} \cdot q_{\vec{\eta}} \cdot \ln\left(\frac{q_{\vec{\eta}}}{p}\right) \geq 0$$

useless since it depends on p

- Intractability of p typically due to its normalization ("Evidence")

- Standard procedure: Minimize the KL divergence:

$$KL[q_{\vec{\eta}} \| p] = \int d\vec{\theta} \cdot q_{\vec{\eta}} \cdot \ln\left(\frac{q_{\vec{\eta}}}{p}\right) \geq 0$$

proposed approximation

true posterior

useless since it depends on p

- Intractability of p typically due to its normalization ("Evidence")

$$p = \tilde{p}/E$$

- Standard procedure: Minimize the KL divergence:

$$KL[q_{\vec{\eta}} \| p] = \int d\vec{\theta} \cdot q_{\vec{\eta}} \cdot \ln\left(\frac{q_{\vec{\eta}}}{p}\right) \geq 0$$

proposed approximation

true posterior

useless since it depends on p

- Intractability of p typically due to its normalization ("Evidence")

$$p = \tilde{p}/E$$

$$ELBO = \int d\vec{\theta} \cdot q_{\vec{\eta}} \ln\left(\frac{\tilde{p}}{q_{\vec{\eta}}}\right)$$

- Standard procedure: Minimize the KL divergence:

$$KL[q_{\vec{\eta}} \| p] = \int d\vec{\theta} \cdot q_{\vec{\eta}} \cdot \ln\left(\frac{q_{\vec{\eta}}}{p}\right) \geq 0$$

proposed approximation

true posterior

useless since it depends on p

- Intractability of p typically due to its normalization ("Evidence")

$$p = \tilde{p}/E$$

$$\text{ELBO} = \int d\vec{\theta} \cdot q_{\vec{\eta}} \ln\left(\frac{\tilde{p}}{q_{\vec{\eta}}}\right)$$

Maximize wrt $\vec{\eta}$

VARIATIONAL INFERENCE

- Standard procedure: Minimize the KL divergence:

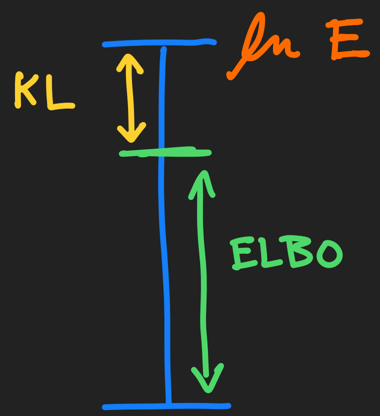
$$KL[q_{\vec{\eta}} \,|\, p] = \int d\vec{\theta} \cdot q_{\vec{\eta}} \cdot \ln\left(\frac{q_{\vec{\eta}}}{p}\right) \geq 0$$

$q_{\vec{\eta}}$: proposed approximation; p: true posterior

Useless since it depends on p

- Intractability of p typically due to its normalization ("Evidence")

$$p = \tilde{p}/E$$



$$\text{ELBO} = \int d\vec{\theta} \cdot q_{\vec{\eta}} \ln\left(\frac{\tilde{p}}{q_{\vec{\eta}}}\right)$$

Maximize wrt $\vec{\eta}$

- Standard procedure: Minimize the KL divergence:

$$KL[q_{\vec{\eta}} \| p] = \int d\vec{\theta} \cdot q_{\vec{\eta}} \cdot \ln\left(\frac{q_{\vec{\eta}}}{p}\right) \geq 0$$

proposed approximation ↳ true posterior

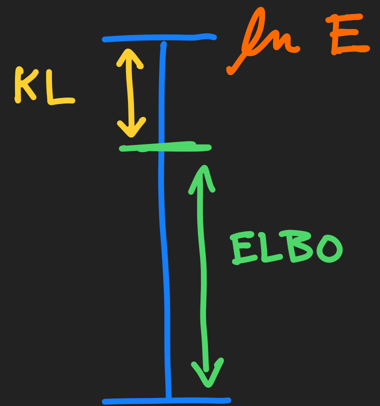
useless since it depends on p

- Intractability of p typically due to its normalization ("Evidence")

$$p = \tilde{p} / E$$

$$\text{ELBO} = \int d\vec{\theta} \cdot q_{\vec{\eta}} \ln\left(\frac{\tilde{p}}{q_{\vec{\eta}}}\right)$$

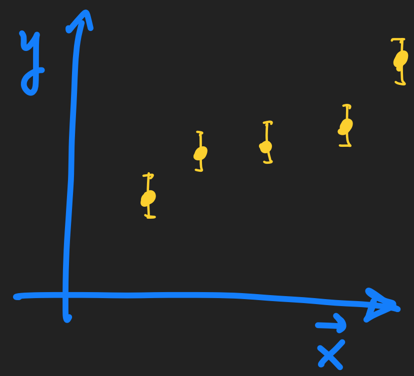
Maximize wrt $\vec{\eta}$



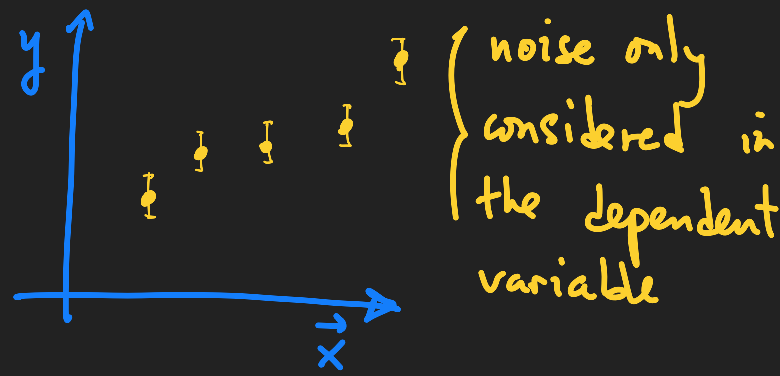
- Bonus:
  Get an approx. to E:
  $\ln E \approx \text{ELBO}$

CASE # 1

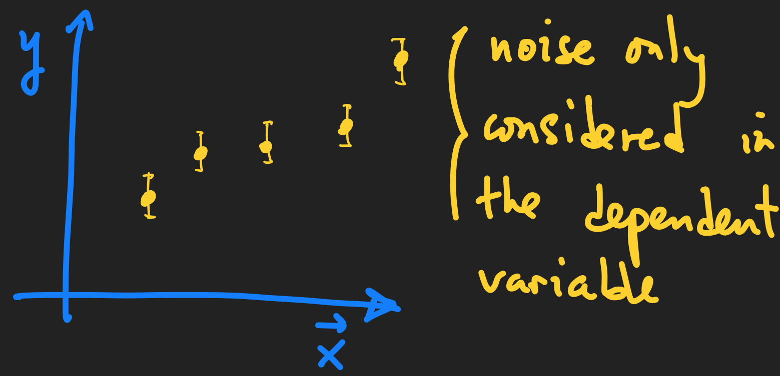
- In a typical data-science problem



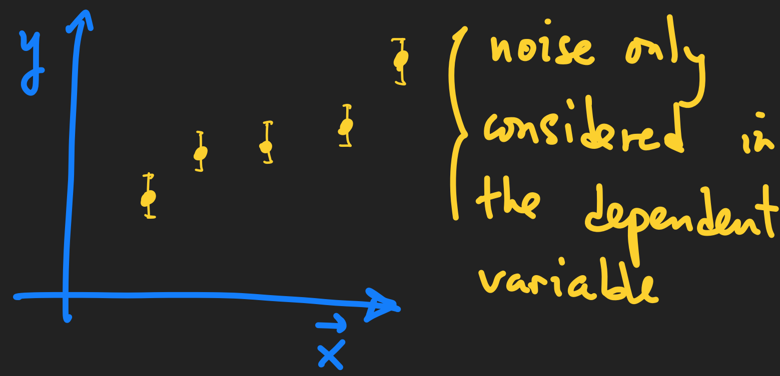
- In a typical data-science problem



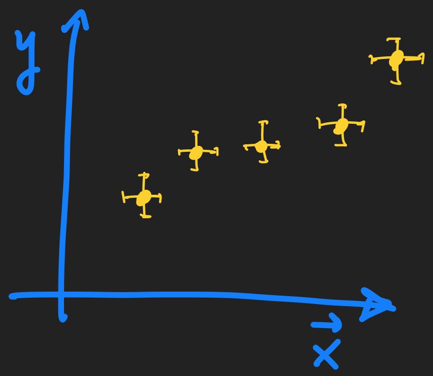
- In a typical data-science problem
- In physics it is quite common to have uncertainties also in the $\vec{x}$ (e.g. instrumental errors)



- In a typical data-science problem

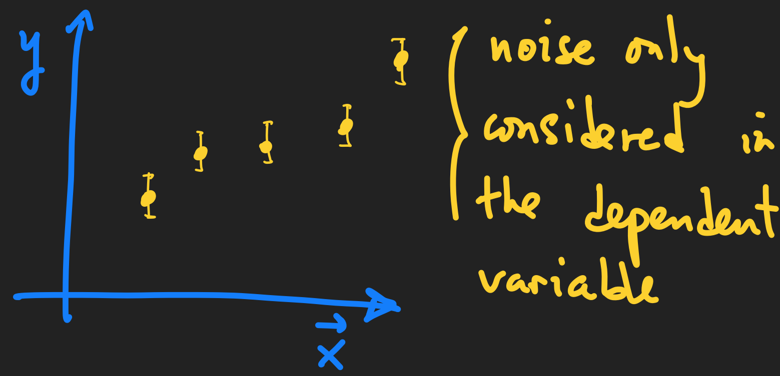


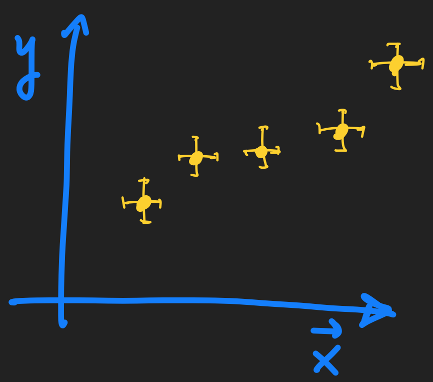
- In physics it is quite common to have uncertainties also in the $\vec{x}$
  (e.g. instrumental errors)



what is the impact on the fitting function?

- In a typical data-science problem
- In physics it is quite common to have uncertainties also in the $\vec{x}$ (e.g. instrumental errors)





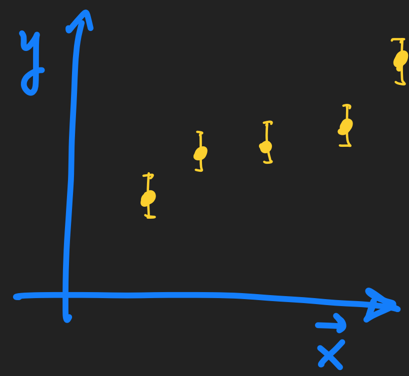
[Numerical Recipes]

e.g. $f(x) = a + bx$

$$\chi^2(a,b) = \sum_{i=1}^{N} \frac{(y_i - f(x_i))^2}{\sigma_{y_i}^2 + b^2 \sigma_{x_i}^2}$$

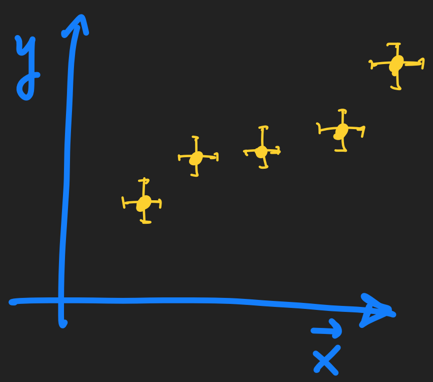
what is the impact on the fitting function?

- In a typical data-science problem



noise only considered in the dependent variable

- In physics it is quite common to have uncertainties also in the $\vec{x}$ (e.g. instrumental errors)



what is the impact on the fitting function?

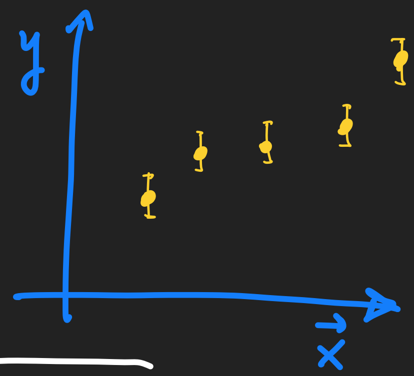
[Numerical Recipes]

e.g. $f(x) = a + bx$

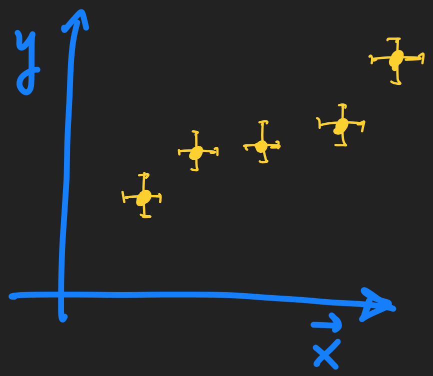
$$\chi^2(a,b) = \sum_{i=1}^{N} \frac{\left(y_i - f(x_i)\right)^2}{\sigma_{y_i}^2 + b^2 \sigma_{x_i}^2}$$

+Many refinements in the literature

- In a typical data-science problem
- In physics it is quite common to have uncertainties also in the $\vec{x}$ (e.g. instrumental errors)



noise only considered in the dependent variable



what is the impact on the fitting function?

[Numerical Recipes]

e.g. $f(x) = a + bx$

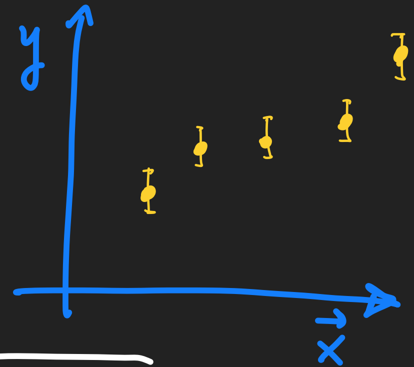
$$\chi^2(a,b) = \sum_{i=1}^{N} \frac{(y_i - f(x_i))^2}{\sigma_{y_i}^2 + b^2\sigma_{x_i}^2}$$

+Many refinements in the literature

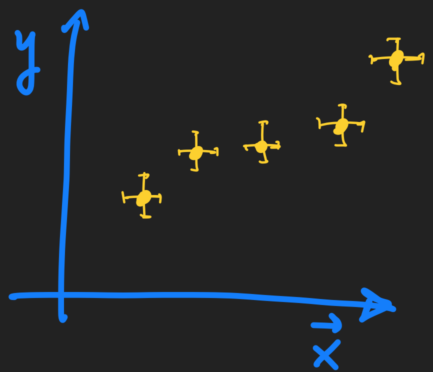
What about classification problems?

(e.g signal-vs-bckg, cats-vs-dogs-vs-donkeys)

- In a typical data-science problem
- In physics it is quite common to have uncertainties also in the $\vec{x}$ (e.g. instrumental errors)



noise only considered in the dependent variable



[Numerical Recipes]

e.g. $f(x) = a + bx$

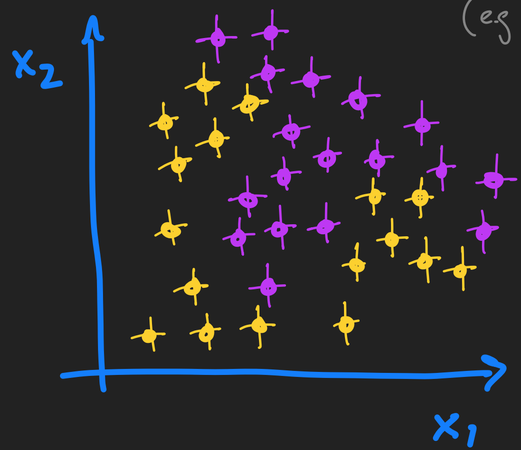
$$\chi^2(a,b) = \sum_{i=1}^{N} \frac{\left(y_i - f(x_i)\right)^2}{\sigma_{y_i}^2 + b^2\sigma_{x_i}^2}$$

what is the impact on the fitting function?

+Many refinements in the literature

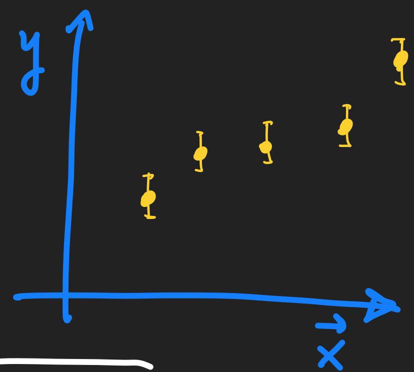
What about classification problems?

(e.g. signal-vs-bckg, cats-vs-dogs-vs-donkeys)

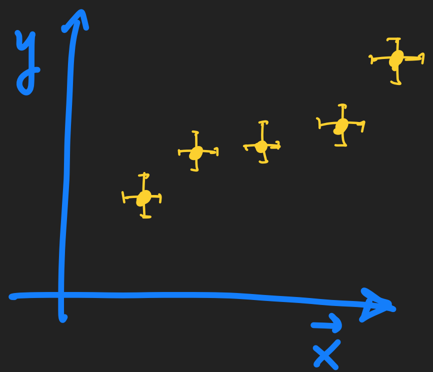


VILLACAMPA, GARRIDO, HERNÁNDEZ, AND BZ, JOURNAL OF ML RESEARCH, 2020

- In a typical data-science problem
- In physics it is quite common to have uncertainties also in the $\vec{x}$ (e.g. instrumental errors)



noise only considered in the dependent variable



[Numerical Recipes]

e.g. $f(x) = a + bx$

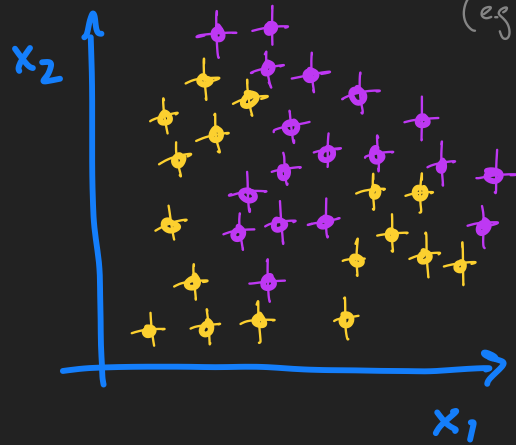
$$\chi^2(a,b) = \sum_{i=1}^{N} \frac{(y_i - f(x_i))^2}{\sigma_{y_i}^2 + b^2\sigma_{x_i}^2}$$

+Many refinements in the literature

what is the impact on the fitting function?

What about classification problems?

(e.g signal-vs-bckg, cats-vs-dogs-vs-donkeys)



Not formally addressed before 2019

VILLACAMPA, GARRIDO, HERNÁNDEZ, AND BZ, JOURNAL OF ML RESEARCH, 2020

- Idea: Each input $\vec{x}_i = \underline{\vec{x}}_i + \Delta\vec{x}_i$

- Idea: Each input $\vec{x}_i = \underline{\vec{x}}_i + \Delta\vec{x}_i$

noisy observation

- Idea: Each input $\vec{x}_i = \underline{\vec{x}}_i + \Delta\vec{x}_i$

noisy observation

actual input (unknown)

- Idea: Each input $\vec{x}_i = \underline{\vec{x}}_i + \Delta\vec{x}_i$

noisy observation

actual input (unknown) ➡ To be modelled

- Idea: Each input $\vec{x}_i = \underline{\vec{x}}_i + \Delta\vec{x}_i$ → Input uncertainty (assumed Gaussian)

noisy observation

actual input (unknown) → To be modelled

- Idea: Each input $\vec{x}_i = \underline{\vec{x}}_i + \Delta\vec{x}_i$ → Input uncertainty (assumed Gaussian)

  noisy observation

  actual input (unknown) → To be modelled

  (in analogy to what we typically do for the dependent variable y)

- Idea: Each input $\vec{x}_i = \underline{\vec{x}}_i + \Delta\vec{x}_i$ → Input uncertainty (assumed Gaussian)

  noisy observation

  actual input (unknown) → To be modelled

  (in analogy to what we typically do for the dependent variable y)

- Likelihood of data $p(Y, X \mid F, \underline{X}) = \prod_i^N p(y_i \mid f(\underline{\vec{x}}_i)) \cdot N(\vec{x}_i \mid \underline{\vec{x}}_i, \Delta\vec{x}_i)$

  Latent variables for the output Y

- Idea: Each input $\vec{x}_i = \underline{\vec{x}}_i + \Delta\vec{x}_i$ → Input uncertainty (assumed Gaussian)

  noisy observation

  actual input (unknown) → To be modelled

  (in analogy to what we typically do for the dependent variable y)

- Likelihood of data $p(Y, X \mid F, \underline{X}) = \prod_i^N p(y_i \mid f(\underline{\vec{x}}_i)) \cdot \mathcal{N}(\vec{x}_i \mid \underline{\vec{x}}_i, \Delta\vec{x}_i)$

  Latent variables for the output Y

- Y modelled as a Gaussian Process (GP)

- Idea: Each input $\vec{x}_i = \underline{\vec{x}}_i + \Delta\vec{x}_i$ → Input uncertainty (assumed Gaussian)

  noisy observation

  actual input (unknown) → To be modelled

  (in analogy to what we typically do for the dependent variable y)

- Likelihood of data $p(Y, X \mid F, \underline{X}) = \prod_i^N p(y_i \mid f(\underline{\vec{x}}_i)) \cdot \mathcal{N}(\vec{x}_i \mid \underline{\vec{x}}_i, \Delta\vec{x}_i)$

  Latent variables for the output Y

- Y modelled as a Gaussian Process (GP) → Very popular Stochastic Process in ML, based on Gaussian distrib. [over functions]

- Idea: Each input $\vec{x}_i = \underline{\vec{x}}_i + \Delta\vec{x}_i$ → Input uncertainty (assumed Gaussian)

  $\vec{x}_i$: noisy observation

  $\underline{\vec{x}}_i$: actual input (unknown) → To be modelled

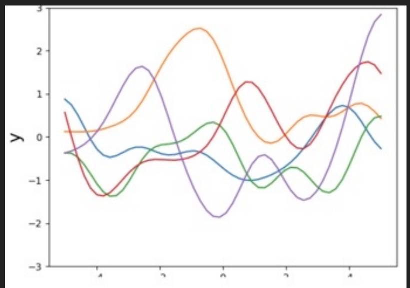
  (in analogy to what we typically do for the dependent variable y)

- Likelihood of data $p(Y, X \mid F, \underline{X}) = \prod_i^N p(y_i \mid f(\underline{\vec{x}}_i)) \cdot N(\vec{x}_i \mid \underline{\vec{x}}_i, \Delta\vec{x}_i)$

  F: Latent variables for the output Y

- Y modelled as a Gaussian Process (GP) → Very popular Stochastic Process in ML, based on Gaussian distrib. [over functions]

  GP prior



- Idea : Each input $\vec{x}_i = \underline{\vec{x}}_i + \Delta\vec{x}_i$ → Input uncertainty (assumed Gaussian)

noisy observation

actual input (unknown) → To be modelled

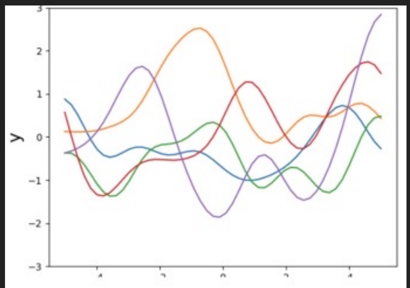
(in analogy to what we typically do for the dependent variable y)

- Likelihood of data $p(Y, X \mid F, \underline{X}) = \prod_i^N p(y_i \mid f(\underline{\vec{x}}_i)) \cdot N(\vec{x}_i \mid \underline{\vec{x}}_i, \Delta\vec{x}_i)$

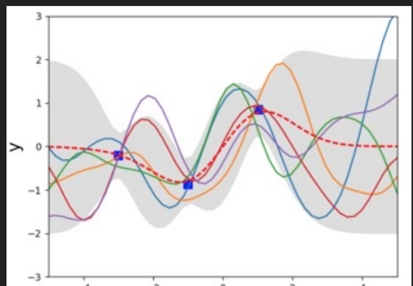
Latent variables for the output Y

- Y modelled as a Gaussian Process (GP) → Very popular Stochastic Process in ML, based on Gaussian distrib. [over functions]

GP prior



GP posterior after 3 obs.



- Idea: Each input $\vec{x}_i = \underline{\vec{x}}_i + \Delta\vec{x}_i$ → Input uncertainty (assumed Gaussian)

  noisy observation

  actual input (unknown) → To be modelled

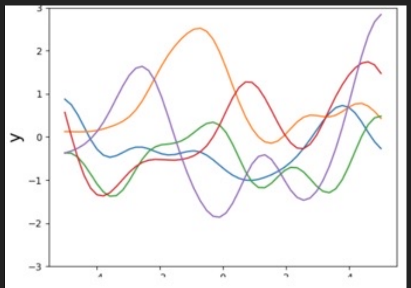
  (in analogy to what we typically do for the dependent variable y)

- Likelihood of data $p(Y, X \mid F, \underline{X}) = \prod_i^N p(y_i \mid f(\underline{\vec{x}}_i)) \cdot \mathcal{N}(\vec{x}_i \mid \underline{\vec{x}}_i, \Delta\vec{x}_i)$

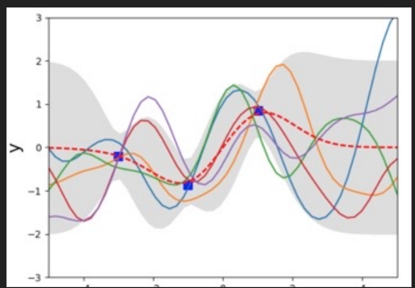
  Latent variables for the output Y

- Y modelled as a Gaussian Process (GP) → Very popular Stochastic Process in ML, based on Gaussian distrib. [over functions]

GP prior



GP posterior after 3 obs.



- GP give analytical predictions in regression problems

- Idea: Each input $\vec{x}_i = \underline{\vec{x}}_i + \Delta\vec{x}_i$ → Input uncertainty (assumed Gaussian)

  noisy observation

  actual input (unknown) → To be modelled

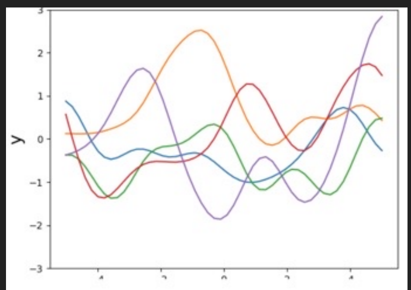
  (in analogy to what we typically do for the dependent variable y)

- Likelihood of data $p(Y, X \mid F, \underline{X}) = \prod_i^N p(y_i \mid f(\underline{\vec{x}}_i)) \cdot N(\vec{x}_i \mid \underline{\vec{x}}_i, \Delta\vec{x}_i)$

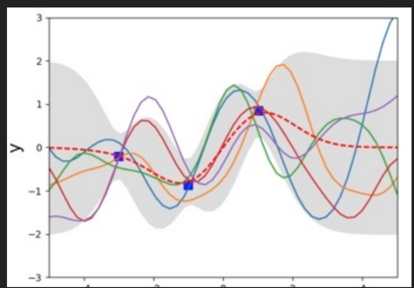
  Latent variables for the output Y

- Y modelled as a Gaussian Process (GP) → Very popular Stochastic Process in ML, based on Gaussian distrib. [over functions]

GP prior



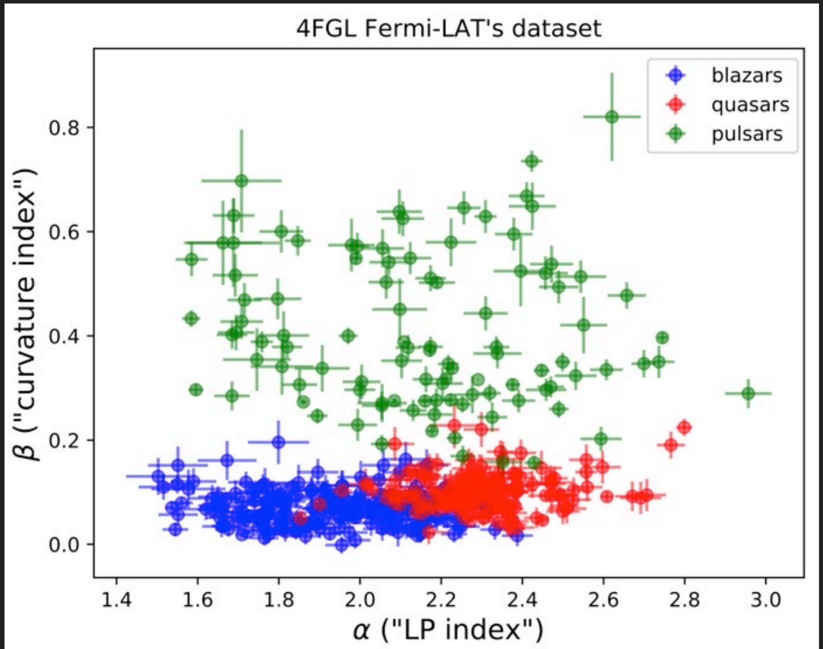
GP posterior after 3 obs.



- GP give analytical predictions in regression problems
- For classification the posterior should be approximated
  - ↳ nowadays typically using Variational Inference

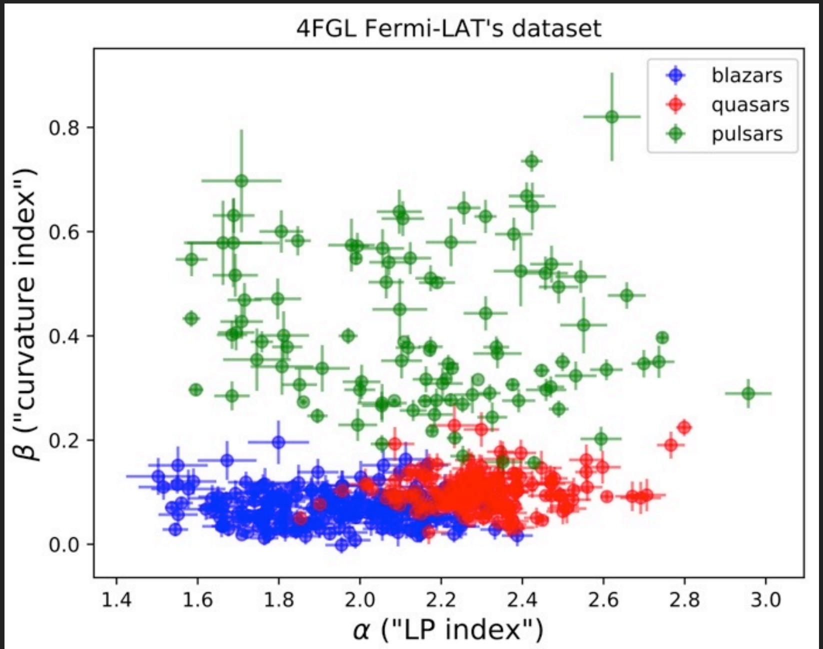
- Tested the benefits of this approach in several synthetic and real datasets,

- Tested the benefits of this approach in several synthetic and real datasets, e.g. →



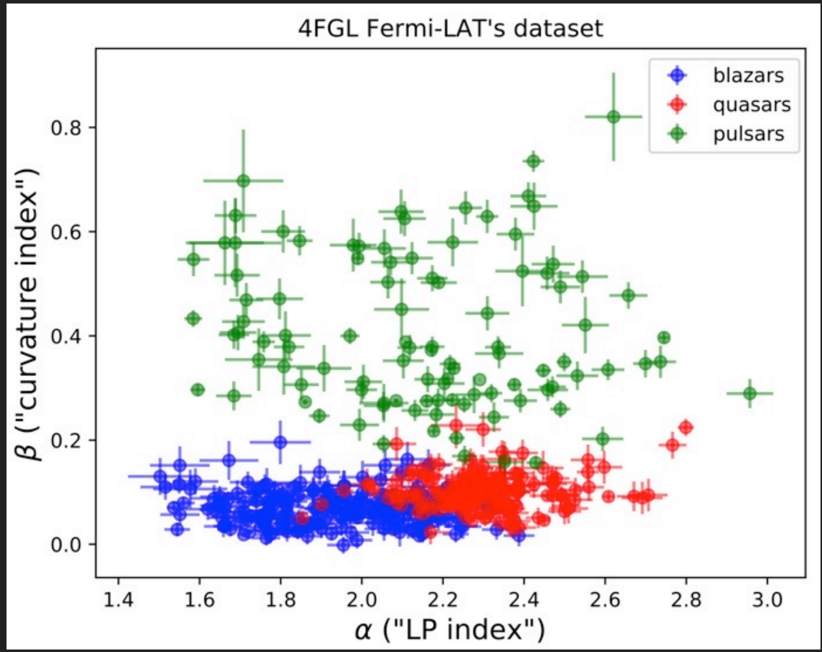
- Tested the benefits of this approach in several synthetic and real datasets, e.g. →

Classifying among 3 types of astro. sources from latest Fermi Catalog



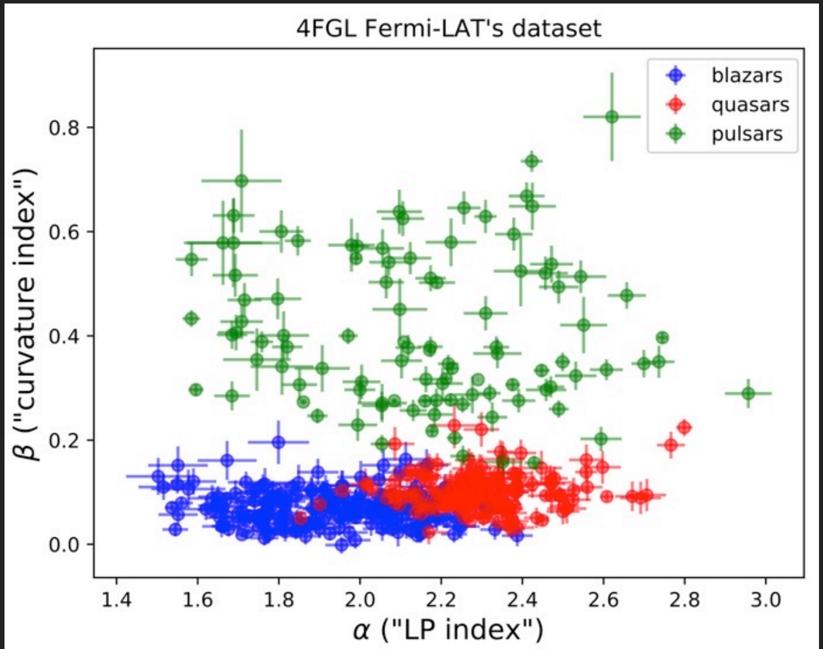
- Tested the benefits of this approach in several synthetic and real datasets, e.g. →

Classifying among 3 types of astro. sources from latest Fermi Catalog



- Considering 6 representative attributes of the point-like sources

- Tested the benefits of this approach in several synthetic and real datasets, e.g. →



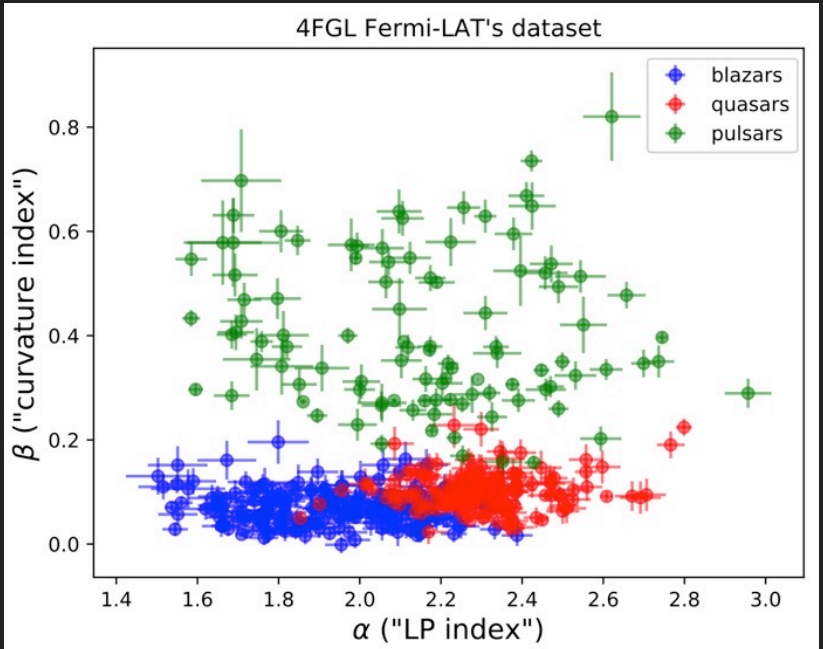
Classifying among 3 types of astro. sources from latest Fermi Catalog

- Considering 6 representative attributes of the point-like sources

	NOISELESS	VARIANT #1	VARIANT #2	VARIANT #3
	MGP	NIMGP	NIMGP_{NN}	NIMGP_{FO}
NLL	0.377±0.0194	**0.246±0.0097**	0.261±0.011	0.292±0.0158
Test error	0.075±0.0043	0.088±0.0038	0.082±0.0041	**0.071±0.0038**

- Tested the benefits of this approach in several synthetic and real datasets, e.g. →

Classifying among 3 types of astro. sources from latest Fermi Catalog



- Considering 6 representative attributes of the point-like sources

	NOISELESS MGP	VARIANT #1 NIMGP	VARIANT #2 NIMGP_{NN}	VARIANT #3 NIMGP_{FO}
NLL	0.377±0.0194	**0.246±0.0097**	0.261±0.011	0.292±0.0158
Test error	0.075±0.0043	0.088±0.0038	0.082±0.0041	**0.071±0.0038**

Take-home message:

- No clear benefit in Accuracy
- Clear advantage for predictive distribution!

CASE #2

SANTANA, BZ, HERNANDEZ, INTERNATIONALCONFERENCE ON ML 2022, (2110.07618)

- What happens in typical Variational Inference for very complicated models? (e.g. deep neural networks)

- What happens in typical Variational Inference for very complicated models? (e.g. deep neural networks)

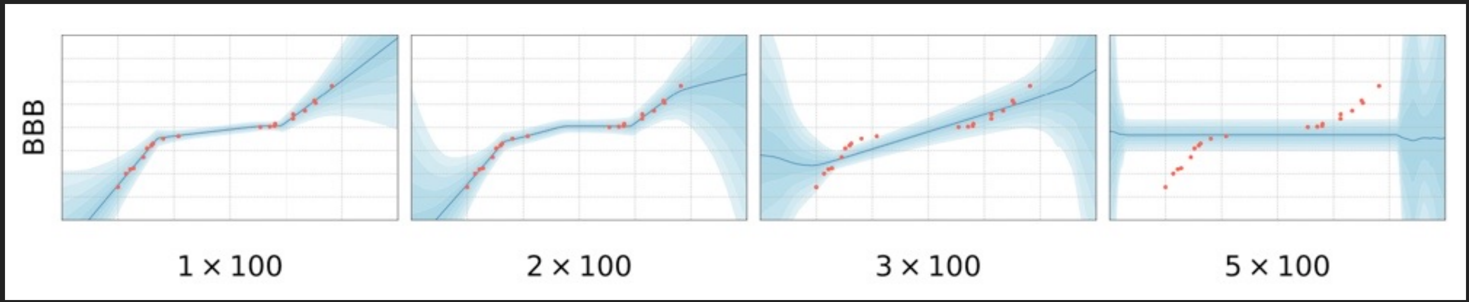
Difficult to keep trade-off $q(\vec{\theta})$

Tractability vs. Expressivity

- What happens in typical Variational Inference for very complicated models? (e.g. deep neural networks)

Difficult to keep trade-off $q(\vec{\theta})$

Tractability vs. Expressivity

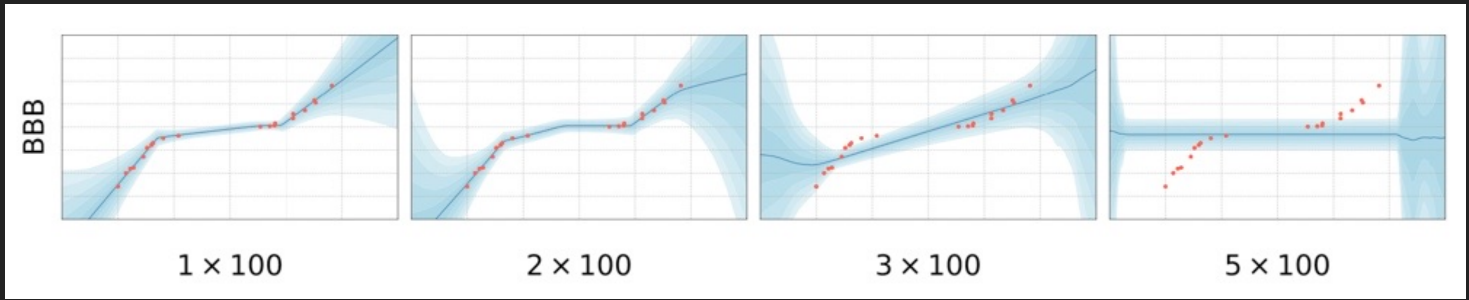


1903.05779

- What happens in typical Variational Inference for very complicated models? (e.g. deep neural networks)

Difficult to keep trade-off $\mathcal{F}(\vec{\theta})$

Tractability vs. Expressivity

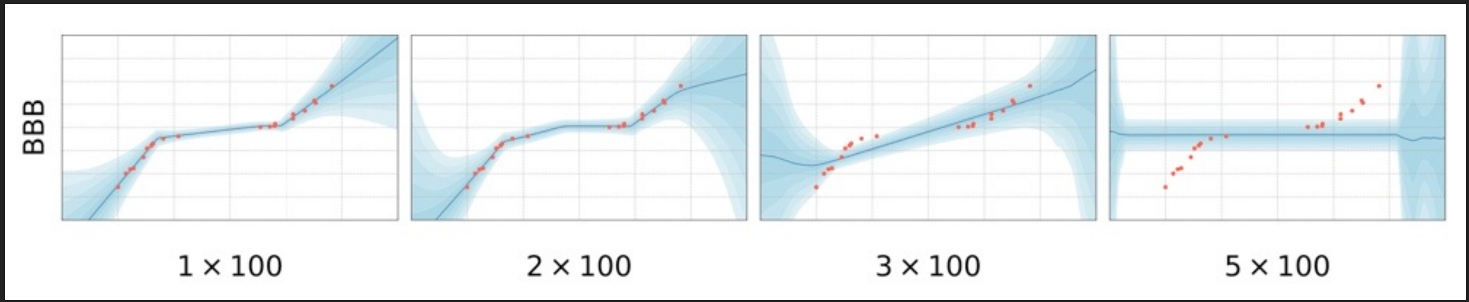


1903.05779

- ML community now considering doing inference in "function space" instead of parameter space

- What happens in typical Variational Inference for very complicated models? (e.g. deep neural networks)

Difficult to keep trade-off $\mathcal{F}(\vec{\theta})$
Tractability vs. Expressivity



1903.05779

- ML community now considering doing inference in "function space" instead of parameter space

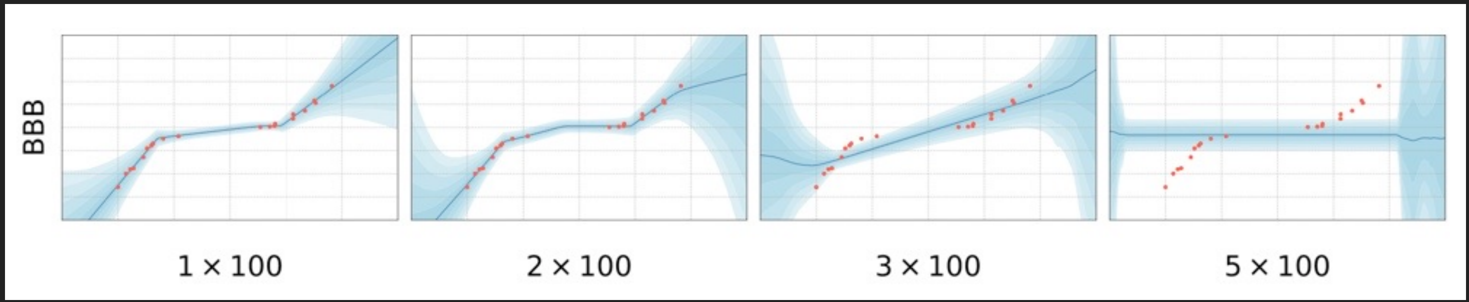
- Priors over functions (GP, "Implicit Processes")

$\{f(x_1), f(x_2), \dots, f(x_N)\}$

- What happens in typical Variational Inference for very complicated models? (e.g. deep neural networks)

Difficult to keep trade-off $\mathcal{F}(\vec{\theta})$

Tractability vs. Expressivity



1903.05779

- ML community now considering doing inference in "function space" instead of parameter space
- Priors over functions (GP, "Implicit Processes")

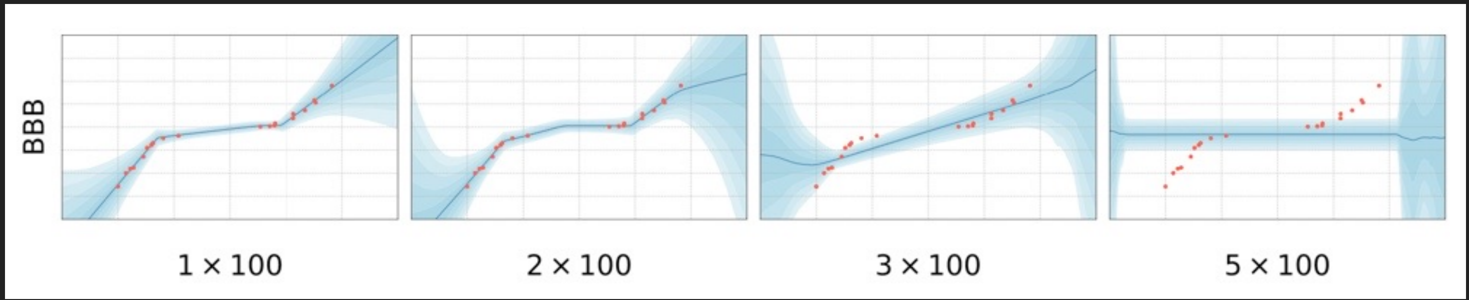
$\{f(x_1), f(x_2), \ldots, f(x_N)\}$

IP defined by "Implicit distributions" [cannot be evaluated, but can be sampled]

- What happens in typical Variational Inference for very complicated models? (e.g. deep neural networks)

Difficult to keep trade-off $q(\vec{\theta})$

Tractability vs. Expressivity



1903.05779

- ML community now considering doing inference in "function space" instead of parameter space
- Priors over functions (GP, "Implicit Processes")

$\{f(x_1), f(x_2), \dots, f(x_N)\}$

IP defined by "Implicit distributions" [cannot be evaluated, but can be sampled]

e.g. Physics simulators:

$$f(x) = g_\theta(x, z) \quad , \quad z \sim p(z)$$

encodes physics

Latent variables

remaining randomness

SANTANA, BZ, HERNANDEZ, INTERNATIONALCONFERENCE ON ML 2022, (2110.07618)

- V.I. procedures in function space are computationally challenging

  ELBO$[\, q \mid \tilde{p} \,]$ now for two implicit distributions over functions!

- V.I. procedures in function space are computationally challenging

  ELBO$[q \,|\, \tilde{p}]$ now for two implicit distributions over functions!

IP priors given by neural networks with (x, z) inputs

$z \to$ stochastic

- V.I. procedures in function space are computationally challenging

  ELBO$[q \,|\, \tilde{p}]$ now for two implicit distributions over functions!

- Useful intuition from standard VI for Gaussian Processes

IP priors given by neural networks with (x, z) inputs

$z \rightarrow$ stochastic

- V.I. procedures in function space are computationally challenging

  ELBO$[q \mid \tilde{p}]$ now for two implicit distributions over functions!

- Useful intuition from standard VI for Gaussian Processes

- $\text{ELBO} \supset \mathbb{E}_q\left[\ln \frac{q}{\text{prior}}\right] = ?$

IP priors given by neural networks with (x, z) inputs

$z \to$ stochastic

- V.I. procedures in function space are computationally challenging

  ELBO[$q \mid \tilde{p}$] now for two implicit distributions over functions!

- Useful intuition from standard VI for Gaussian Processes

- ELBO $\supset \mathbb{E}_q \left[\ln \frac{q}{\text{prior}} \right] =$?

$\ln \frac{q}{\text{prior}} \approx D_{w_*}$ } a "discriminator" (e.g. neural net) trained to distinguish samples from q and prior

IP priors given by neural networks with (x, z) inputs

z ↳ stochastic

- V.I. procedures in function space are computationally challenging
  ELBO$[q \,|\, \tilde{p}]$ now for two implicit distributions over functions!

IP priors given by neural networks with (x, z) inputs
z → stochastic

- Useful intuition from standard VI for Gaussian Processes

- $\text{ELBO} \supset \mathbb{E}_q \left[\ln \frac{q}{\text{prior}} \right] = ?$

$\ln \frac{q}{\text{prior}} \approx D_{w_*}$ } a "discriminator" (e.g. neural net) trained to distinguish samples from q and prior

- Results

- V.I. procedures in function space are computationally challenging

  $ELBO[q \mid \tilde{p}]$ now for two implicit distributions over functions!

IP priors given by neural networks with (x, z) inputs

z → stochastic

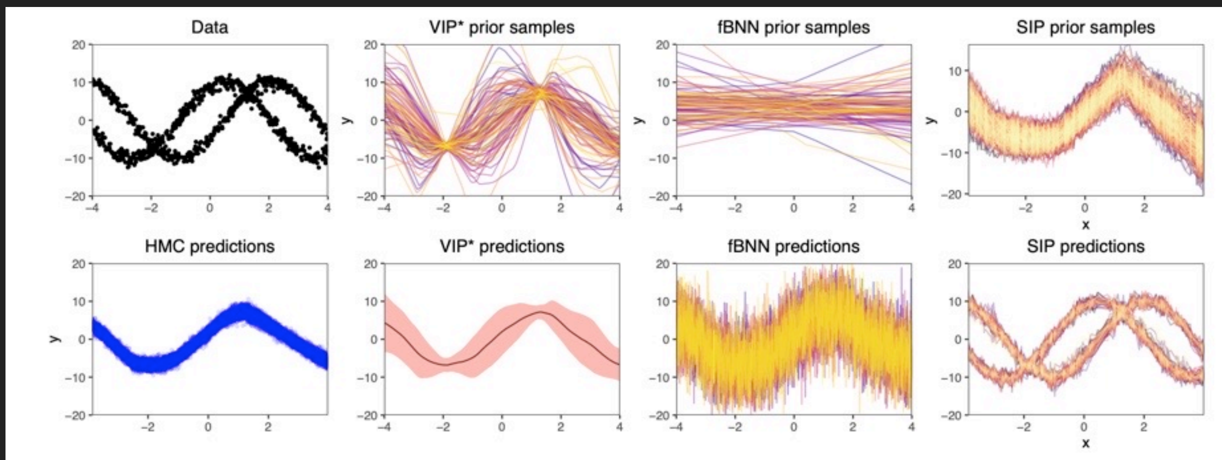
- Useful intuition from standard VI for Gaussian Processes

- $ELBO \supset \mathbb{E}_q\left[\ln \frac{q}{prior}\right] = ?$

$$\ln \frac{q}{prior} \approx D_{w_*}$$

a "discriminator" (e.g. neural net) trained to distinguish samples from q and prior

- Results



- V.I. procedures in function space are computationally challenging

  ELBO$[q \mid \tilde{p}]$ now for two implicit distributions over functions !

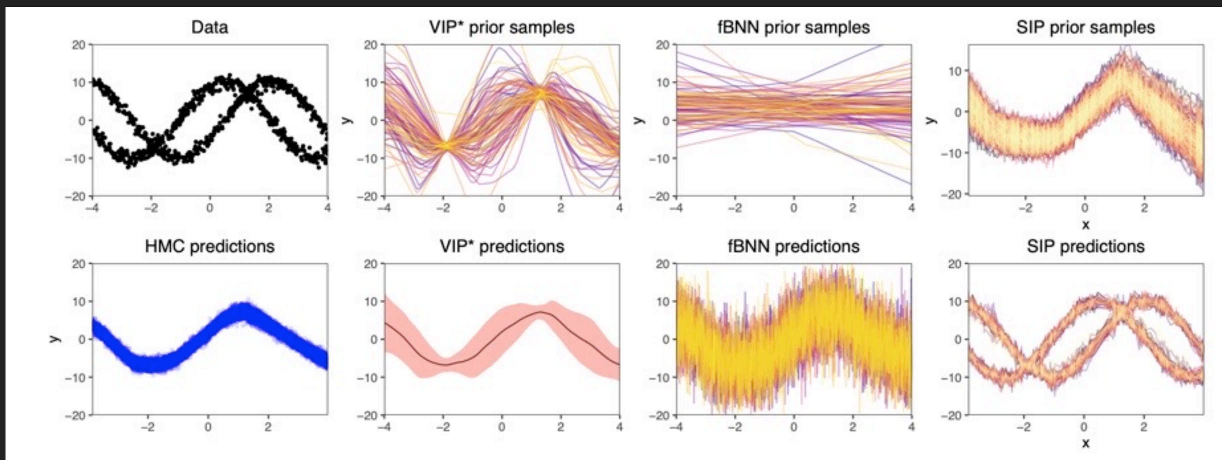
- Useful intuition from standard VI for Gaussian Processes

- ELBO $\supset \mathbb{E}_q \left[\ln \frac{q}{\text{prior}} \right] = ?$ $\quad \ln \frac{q}{\text{prior}} \approx D_{w_*}$ $\Big\}$ a "discriminator" (e.g. neural net) trained to distinguish samples from q and prior

IP priors given by neural networks with (x, z) inputs

z → stochastic

- Results



"Golden truth" | Existing method #1 | Existing method #2 | Our method

- V.I. procedures in function space are computationally challenging

  ELBO$[q \,|\, \tilde{p}]$ now for two implicit distributions over functions!

- Useful intuition from standard VI for Gaussian Processes

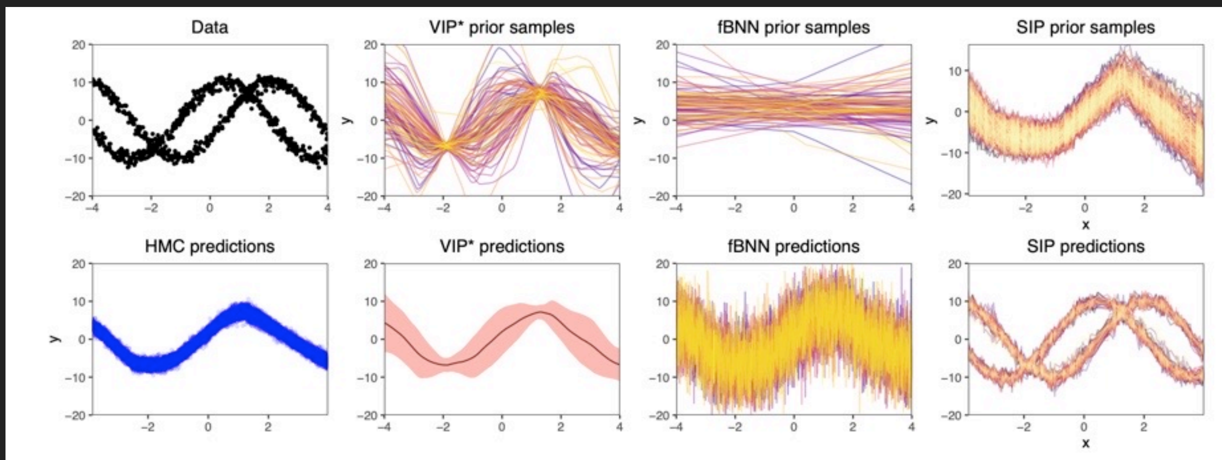
- $\text{ELBO} \supset \mathbb{E}_q\left[\ln \frac{q}{\text{prior}}\right] = ?$

  $\ln \frac{q}{\text{prior}} \approx D_{w_*}$ } a "discriminator" (e.g. neural net) trained to distinguish samples from q and prior

IP priors given by neural networks with (x, z) inputs

z ↳ stochastic

- Results



"Golden truth" | Existing method #1 | Existing method #2 | Our method

+ Our method performs better in general for a collection of other real datasets

CASE #3

OTHER ACTIVE RESEARCH BRANCH IN VARIATIONAL INFERENCE

Consider the following mathematical identity ("reweighting" - Lattice QCD jargon)

$$\mathbb{E}_p[f(\theta)] = \int d\theta\, p(\theta)\, f(\theta) = \int d\theta\, q(\theta)\, \frac{p(\theta)}{q(\theta)}\, f(\theta) = \mathbb{E}_q\Big[\underbrace{\frac{p}{q}}_{\text{"weights"}} f(\theta)\Big]$$

Consider the following mathematical identity ("reweighting" - Lattice QCD jargon)

$$\mathbb{E}_p[f(\theta)] = \int d\theta\, p(\theta)\, f(\theta) = \int d\theta\, q(\theta)\, \frac{p(\theta)}{q(\theta)}\, f(\theta) = \mathbb{E}_q\Big[\underbrace{\frac{p}{q}}_{\text{"weights"}}\, f(\theta)\Big]$$

- This is the basis of "Importance Sampling"

Consider the following mathematical identity ("reweighting" - Lattice QCD jargon)

$$\mathbb{E}_p[f(\theta)] = \int d\theta\, p(\theta)\, f(\theta) = \int d\theta\, q(\theta)\, \frac{p(\theta)}{q(\theta)}\, f(\theta) = \mathbb{E}_q\Big[\underbrace{\frac{p}{q}}_{\text{"weights"}} f(\theta)\Big]$$

- This is the basis of "Importance Sampling" $\Rightarrow$ computing $\mathbb{E}$ under complicated p.

Consider the following mathematical identity ("reweighting" - Lattice QCD jargon)

$$\mathbb{E}_p[f(\theta)] = \int d\theta\, p(\theta)\, f(\theta) = \int d\theta\, q(\theta)\, \frac{p(\theta)}{q(\theta)}\, f(\theta) = \mathbb{E}_q\Big[\underbrace{\frac{p}{q}}_{\text{"weights"}}\, f(\theta)\Big]$$

- This is the basis of "Importance Sampling" ⇒ computing $\mathbb{E}$ under complicated p.
- Useful for V.I. ⇒

Consider the following mathematical identity ("reweighting" - Lattice QCD jargon)

$$\mathbb{E}_p[f(\theta)] = \int d\theta\, p(\theta)\, f(\theta) = \int d\theta\, q(\theta)\, \frac{p(\theta)}{q(\theta)}\, f(\theta) = \mathbb{E}_q\Big[\underbrace{\frac{p}{q}}_{\text{"weights"}}\, f(\theta)\Big]$$

- This is the basis of "Importance Sampling" ⇒ computing $\mathbb{E}$ under complicated p.
- Useful for V.I. ⇒ posterior ≈ $q(\theta)$ s.t. $p(y_*|x_*) = \mathbb{E}_p[\,p(y_*|x_*,\theta)\,]$

$$\approx \mathbb{E}_q[\quad " \quad]$$

Consider the following mathematical identity ("reweighting" - Lattice QCD jargon)

$$\mathbb{E}_p[f(\theta)] = \int d\theta\, p(\theta)\, f(\theta) = \int d\theta\, q(\theta)\, \frac{p(\theta)}{q(\theta)}\, f(\theta) = \mathbb{E}_q\Big[\underbrace{\frac{p}{q}}_{\text{"weights"}}\, f(\theta)\Big]$$

- This is the basis of "Importance Sampling" $\Rightarrow$ computing $\mathbb{E}$ under complicated p.
- Useful for V.I. $\Rightarrow$ posterior $\approx q(\theta)$ s.t. $p(y_*|x_*) = \mathbb{E}_p[\,p(y_*|x_*,\theta)\,]$ $\approx \mathbb{E}_q[\quad " \quad]$

V.I. is inherently biased

Consider the following mathematical identity ("reweighting" - Lattice QCD jargon)

$$\mathbb{E}_p[f(\theta)] = \int d\theta\, p(\theta)\, f(\theta) = \int d\theta\, q(\theta)\, \frac{p(\theta)}{q(\theta)}\, f(\theta) = \mathbb{E}_q\Big[\underbrace{\frac{p}{q}}_{\text{"weights"}}\, f(\theta)\Big]$$

- This is the basis of "Importance Sampling" ⇒ computing $\mathbb{E}$ under complicated p.
- Useful for V.I. ⇒ posterior $\approx q(\theta)$ s.t. $p(y_*|x_*) = \mathbb{E}_p[\,p(y_*|x_*,\theta)\,]$
  $\approx \mathbb{E}_q[\quad " \quad]$

V.I. is inherently biased

⇒ Reweighting helps "de-biasing" V.I

(see Jordan et al, 2106.15980
and refs. there in)

Consider the following mathematical identity ("reweighting" - Lattice QCD jargon)

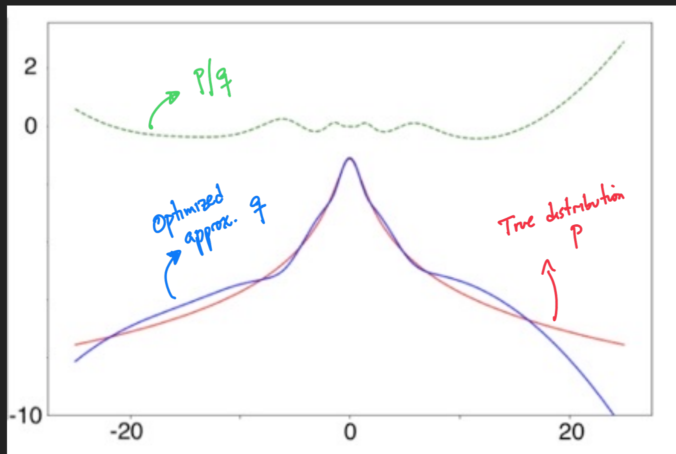
$$\mathbb{E}_p[f(\theta)] = \int d\theta\, p(\theta)\, f(\theta) = \int d\theta\, q(\theta)\, \frac{p(\theta)}{q(\theta)}\, f(\theta) = \mathbb{E}_q\Big[\underbrace{\frac{p}{q}}_{\text{"weights"}}\, f(\theta)\Big]$$

- This is the basis of "Importance Sampling" ➡ computing $\mathbb{E}$ under complicated p.
- Useful for V.I. ➡ posterior $\approx q(\theta)$ s.t. $p(y_*|x_*) = \mathbb{E}_p[p(y_*|x_*,\theta)] \approx \mathbb{E}_q[\quad " \quad]$

V.I. is inherently biased

➡ Reweighting helps "de-biasing" V.I

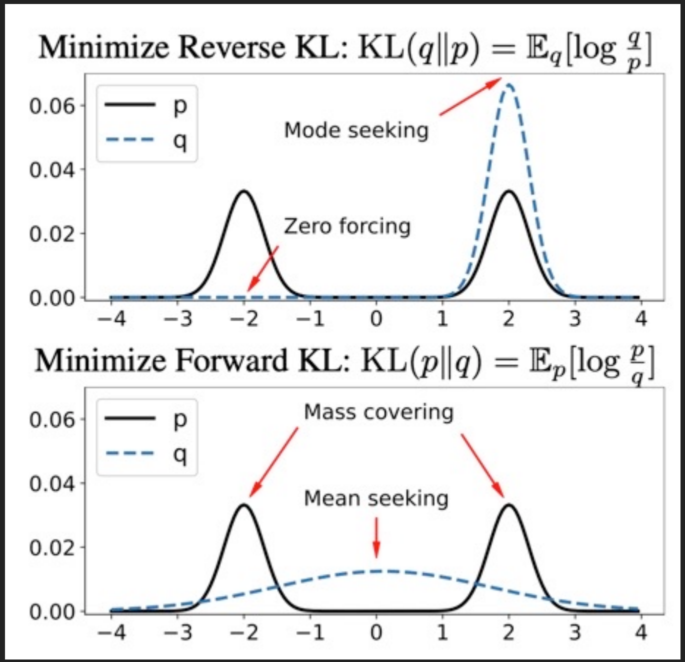
(see Jordan et al, 2106.15980 and refs. there in)



OTHER ACTIVE RESEARCH BRANCH IN VARIATIONAL INFERENCE

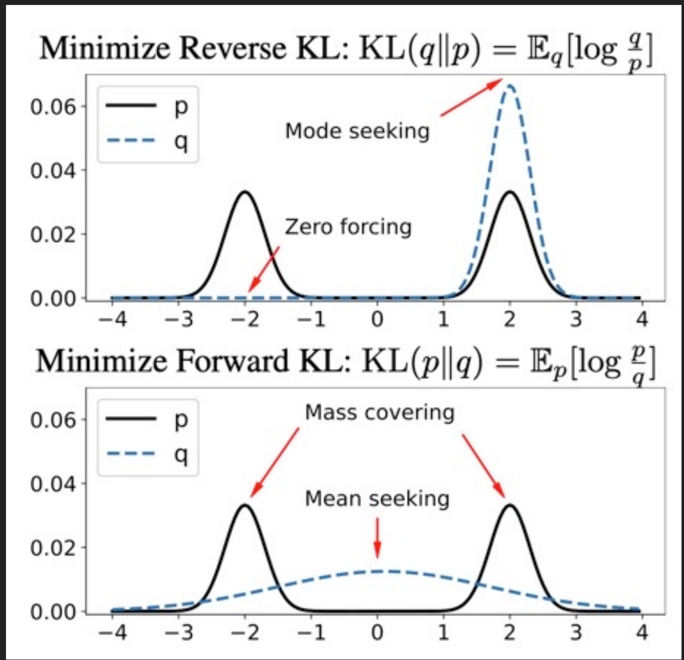
- Challenges here :

- Challenges here:



- Challenges here :

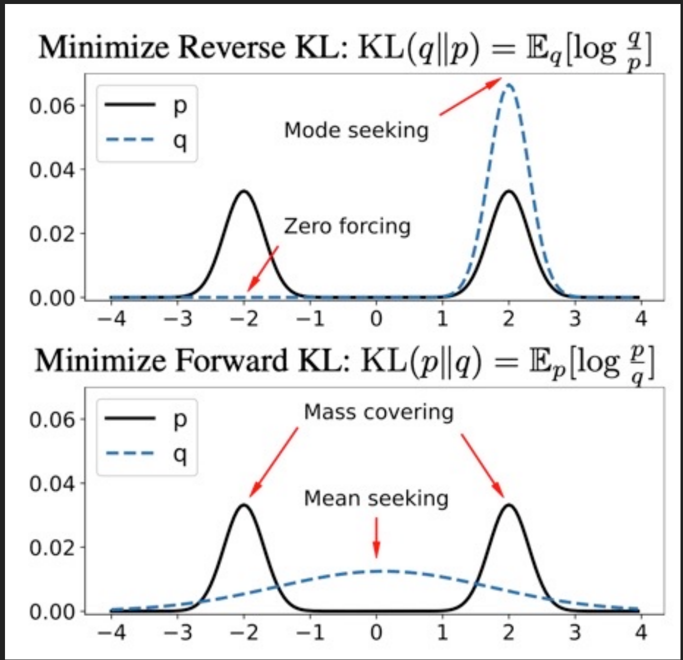
weights have typically very large variance



- Challenges here:

weights have typically very large variance

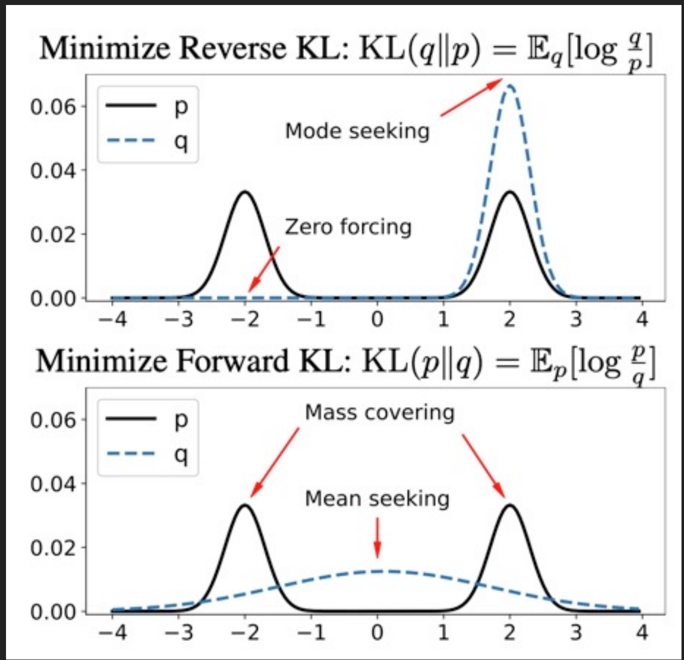
smaller variance, but computationally very costly



- Challenges here:

weights have typically very large variance

smaller variance, but computationally very costly

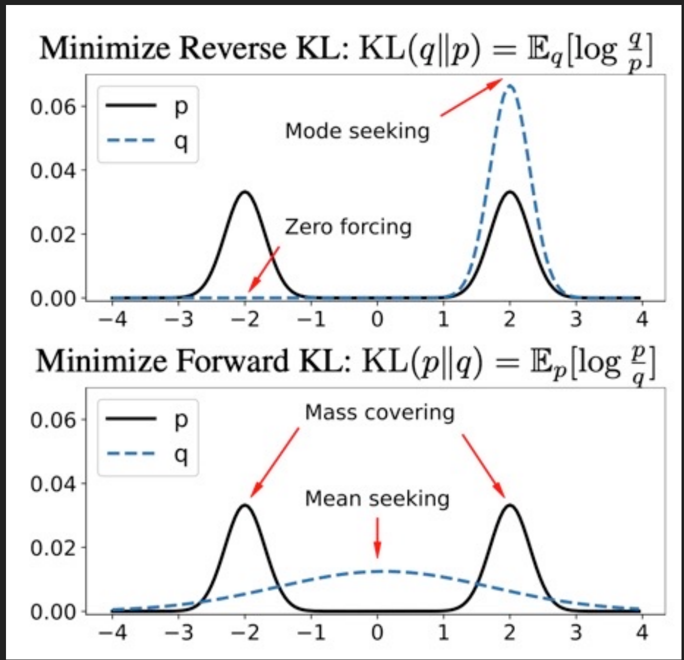


* Computation-Quality trade-off

- Challenges here:

weights have typically very large variance

smaller variance, but computationally very costly



* Computation - Quality trade-off

* Other "distance measures" are definitely worth exploring!

[A. Dimitriou, A. Ramos, G. Telo and BZ work in progress]

CASE # 4

- Idea of MCMC

- Idea of MCMC

  * Get samples $\vec{\theta}_1, \vec{\theta}_2, \ldots, \vec{\theta}_t, \vec{\theta}_{t+1}, \ldots$

    from proposal dist. $q(\vec{\theta}_{t+1} | \vec{\theta}_t)$

- Idea of MCMC

  * Get samples $\vec{\theta}_1, \vec{\theta}_2, \ldots, \vec{\theta}_t, \vec{\theta}_{t+1}, \ldots$ from proposal dist. $q(\vec{\theta}_{t+1} | \vec{\theta}_t)$ $\Big\}$ Markov chain

- Idea of MCMC

  * Get samples $\vec{\theta}_1, \vec{\theta}_2, \dots, \vec{\theta}_t, \vec{\theta}_{t+1}, \dots$ from proposal dist. $q(\vec{\theta}_{t+1} | \vec{\theta}_t)$ } Markov chain

  * Accept/Reject $\vec{\theta}_{t+1}$ according to some criterion (typically ensuring good properties of the Markov chain)

- **Idea of MCMC**

  * Get samples $\vec{\theta}_1, \vec{\theta}_2, \dots, \vec{\theta}_t, \vec{\theta}_{t+1}, \dots$ from proposal dist. $q(\vec{\theta}_{t+1} | \vec{\theta}_t)$ } Markov chain

  * Accept/Reject $\vec{\theta}_{t+1}$ according to some criterion (typically ensuring good properties of the Markov chain)

$$p(\vec{\theta} | \text{Data}) \equiv \frac{f(\vec{\theta})}{p(\text{Data})}$$

- Idea of MCMC

  * Get samples $\vec{\theta}_1, \vec{\theta}_2, \dots, \vec{\theta}_t, \vec{\theta}_{t+1}, \dots$ from proposal dist. $q(\vec{\theta}_{t+1} | \vec{\theta}_t)$ } Markov chain

  * Accept/Reject $\vec{\theta}_{t+1}$ according to some criterion (typically ensuring good properties of the Markov chain)

$$p(\vec{\theta}|\text{Data}) = \frac{f(\vec{\theta})}{p(\text{Data})}$$

$$r = \frac{f(\vec{\theta}_{t+1})}{f(\vec{\theta}_t)}$$

$$u \sim \text{unif}[0,1]$$

- Idea of MCMC

  * Get samples $\vec{\theta}_1, \vec{\theta}_2, \dots, \vec{\theta}_t, \vec{\theta}_{t+1}, \dots$ from proposal dist. $q(\vec{\theta}_{t+1} | \vec{\theta}_t)$ } Markov chain

  * Accept/Reject $\vec{\theta}_{t+1}$ according to some criterion (typically ensuring good properties of the Markov chain)

$$p(\vec{\theta}|\text{Data}) \equiv \frac{f(\vec{\theta})}{p(\text{Data})}$$

$$r = \frac{f(\vec{\theta}_{t+1})}{f(\vec{\theta}_t)}$$

$$u \sim \text{unif}[0,1]$$

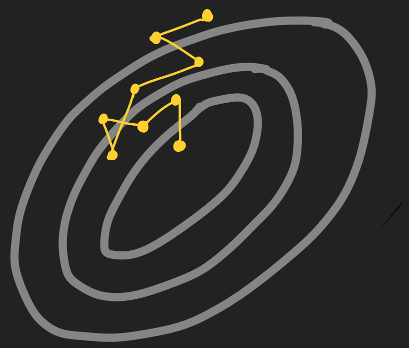
$u \le r \Rightarrow$ Accept $\vec{\theta}_{t+1}$

$u > r \Rightarrow$ Reject it

HYBRID (A.K.A. HAMILTON) MONTE CARLO

- Random walk behaviour
  (e.g. Metropolis-Hastings)

- Random walk behaviour (e.g. Metropolis-Hastings)



- Random walk behaviour
  (e.g. Metropolis-Hastings)

Would be convenient to have guidance on where to move next !

- Random walk behaviour (e.g. Metropolis-Hastings)

Would be convenient to have guidance on where to move next!

- Hamiltonian dynamics at work

- Random walk behaviour (e.g. Metropolis-Hastings)

Would be convenient to have guidance on where to move next!

- Hamiltonian dynamics at work

  * $\vec{\theta} \equiv \vec{q} \rightarrow (\vec{q}, \vec{p})$

- Random walk behaviour (e.g. Metropolis-Hastings)

Would be convenient to have guidance on where to move next!

- Hamiltonian dynamics at work

  * $\vec{\theta} \equiv \vec{q} \rightarrow (\vec{q}, \vec{p})$

  * $H(\vec{q}, \vec{p}) = K(\vec{p}) + V(\vec{q})$

HYBRID (A.K.A. HAMILTON) MONTE CARLO

- Random walk behaviour (e.g. Metropolis-Hastings)

Would be convenient to have guidance on where to move next!

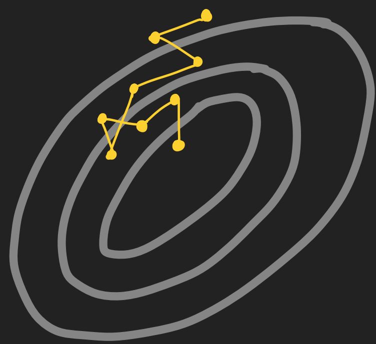
- Hamiltonian dynamics at work

* $\vec{\theta} \equiv \vec{q} \to (\vec{q}, \vec{p})$

* $H(\vec{q}, \vec{p}) = K(\vec{p}) + \overbrace{V(\vec{q})}^{= \ln p(\vec{q}, \text{Data})}$

HYBRID (A.K.A. HAMILTON) MONTE CARLO

- Random walk behaviour (e.g. Metropolis-Hastings)



Would be convenient to have guidance on where to move next!

- Hamiltonian dynamics at work

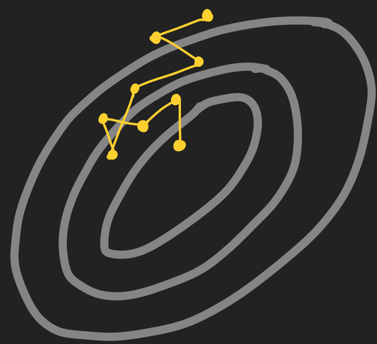
  * $\vec{\theta} \equiv \vec{q} \rightarrow (\vec{q}, \vec{p})$

  * $H(\vec{q}, \vec{p}) = K(\vec{p}) + \underbrace{V(\vec{q})}_{= \ln p(\vec{q},\, \mathrm{Data})}$

$$\frac{dq}{dt} = \frac{\partial H}{\partial p}, \quad \frac{dp}{dt} = -\frac{\partial H}{\partial q}$$

HYBRID (A.K.A. HAMILTON) MONTE CARLO

- Random walk behaviour
  (e.g. Metropolis-Hastings)



Would be convenient to have guidance on where to move next!

- Hamiltonian dynamics at work

  * $\vec{\theta} \equiv \vec{q} \rightarrow (\vec{q}, \vec{p})$

  * $H(\vec{q}, \vec{p}) = K(\vec{p}) + \overbrace{V(\vec{q})}^{= \ln p(\vec{q}, \text{Data})}$

$$\frac{dq}{dt} = \frac{\partial H}{\partial p}, \quad \frac{dp}{dt} = -\frac{\partial H}{\partial q}$$

to be solved numerically by a good algorithm

HYBRID (A.K.A. HAMILTON) MONTE CARLO

- Random walk behaviour (e.g. Metropolis–Hastings)

Would be convenient to have guidance on where to move next!

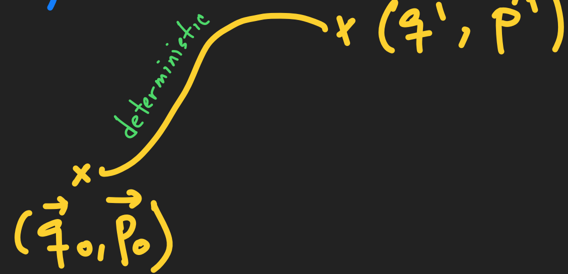
- Hamiltonian dynamics at work

  * $\vec{\theta} \equiv \vec{q} \rightarrow (\vec{q}, \vec{p})$

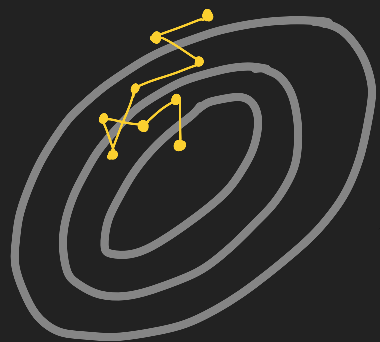
  * $H(\vec{q}, \vec{p}) = K(\vec{p}) + \overbrace{V(\vec{q})}^{= \ln p(\vec{q}, \text{Data})}$

$$\frac{dq}{dt} = \frac{\partial H}{\partial p}, \quad \frac{dp}{dt} = -\frac{\partial H}{\partial q}$$

to be solved numerically by a good algorithm



- Random walk behaviour (e.g. Metropolis-Hastings)



Would be convenient to have guidance on where to move next !

- Hamiltonian dynamics at work

  * $\vec{\theta} \equiv \vec{q} \rightarrow (\vec{q}, \vec{p})$

  * $H(\vec{q}, \vec{p}) = K(\vec{p}) + \underbrace{V(\vec{q})}_{= \ln p(\vec{q}, \text{Data})}$

$$\frac{dq}{dt} = \frac{\partial H}{\partial p}, \quad \frac{dp}{dt} = -\frac{\partial H}{\partial q}$$

to be solved numerically by a good algorithm

$(\vec{q}_0, \vec{p}_0) \xrightarrow{\text{deterministic}} (\vec{q}\,', \vec{p}\,')$

Probability of accepting

$$\min\left(1, \frac{e^{-H'}}{e^{-H_0}}\right)$$

- The predictive distribution contain "hyper-parameters"

- The predictive distribution contain "hyper-parameters"

$$p(y_* | x_*, \text{Data}) = \int d\vec{\theta}^{\text{ model parameters}} \begin{pmatrix} \text{Likelihood} \\ \text{of } y_* | x_*, \vec{\theta} \end{pmatrix} \cdot \begin{pmatrix} \text{Posterior of} \\ \vec{\theta} \mid \text{Data} \end{pmatrix}$$

- The predictive distribution contain "hyper-parameters"

$$P(y_* | x_*, \text{Data}) = \int \overset{\text{model parameters}}{d\vec{\theta}} \left(\begin{matrix} \text{Likelihood} \\ \text{of } y_* | x_*, \vec{\theta} \end{matrix} \right) \cdot \left(\begin{matrix} \text{Posterior of} \\ \vec{\theta} \,|\, \text{Data} \end{matrix} \right) \Bigg\} \propto P(\text{Data} | \vec{\theta}, \underbrace{\vec{\alpha}}_{\text{Likelihood hyper-parameters}}) \, P(\vec{\theta} \,|\, \underbrace{\vec{\beta}}_{\text{prior hyper-parameters}})$$

- The predictive distribution contain "hyper-parameters"

model parameters

Likelihood hyper-parameters

prior hyper-parameters

$$p(y_* | x_*, \text{Data}) = \int d\vec{\theta} \begin{pmatrix} \text{Likelihood} \\ \text{of } y_* | x_*, \vec{\theta} \end{pmatrix} \cdot \underbrace{\begin{pmatrix} \text{Posterior of} \\ \vec{\theta} \mid \text{Data} \end{pmatrix}}_{\propto\, p(\text{Data} | \vec{\theta}, \vec{\alpha})\, p(\vec{\theta} | \vec{\beta})}$$

e.g. prior $= N(\vec{\theta} \,|\, \vec{\mu}, \sigma \cdot \mathbb{1})$

- The predictive distribution contain "hyper-parameters"

$$P(y_*|x_*,\text{Data}) = \int \underbrace{d\vec{\theta}}_{\text{model parameters}} \underbrace{\left(\text{Likelihood of } y_* | x_*, \vec{\theta}\right)}_{P(y_*|x_*,\vec{\theta},\vec{\alpha})} \cdot \left\{\left(\text{Posterior of } \vec{\theta} \mid \text{Data}\right)\right\} \propto P(\text{Data}|\vec{\theta}, \underbrace{\vec{\alpha}}_{\text{Likelihood hyper-parameters}}) \, P(\vec{\theta} \mid \underbrace{\vec{\beta}}_{\text{prior hyper-parameters}})$$

e.g. $\text{prior} = N(\vec{\theta} \mid \vec{\mu}, \sigma \cdot \mathbb{1})$

- The predictive distribution contain "hyper-parameters"

model parameters

Likelihood hyper-parameters

prior hyper-parameters

$$P(y_*|x_*,\text{Data}) = \int d\vec{\theta}\,\underbrace{\left(\text{Likelihood of } y_*|x_*,\vec{\theta}\right)}_{P(y_*|x_*,\vec{\theta},\vec{\alpha})}\cdot\left(\begin{matrix}\text{Posterior of}\\ \vec{\theta}\,|\,\text{Data}\end{matrix}\right)$$

$$\text{Posterior of } \vec{\theta}\,|\,\text{Data} \propto P(\text{Data}|\vec{\theta},\vec{\alpha})\,P(\vec{\theta}\,|\,\vec{\beta})$$

e.g. $\text{prior} = N(\vec{\theta}\,|\,\vec{\mu},\sigma\cdot\mathbb{1})$

- How to optimize these hyper-parameters ?

- The predictive distribution contain "hyper-parameters"

$$P(y_*|x_*,\text{Data}) = \int \overset{\text{model parameters}}{d\vec{\theta}} \underbrace{\begin{pmatrix}\text{Likelihood} \\ \text{of } y_*|x_*,\vec{\theta}\end{pmatrix}}_{P(y_*|x_*,\vec{\theta},\vec{\alpha})} \cdot \left.\begin{pmatrix}\text{Posterior of} \\ \vec{\theta}\,|\,\text{Data}\end{pmatrix}\right\} \propto P(\text{Data}|\vec{\theta},\underset{\text{Likelihood hyper-parameters}}{\vec{\alpha}})\, P(\vec{\theta}\,|\underset{\text{prior hyper-parameters}}{\vec{\beta}})$$

e.g. $\text{prior} = N(\vec{\theta}\,|\,\vec{\mu},\sigma\cdot\mathbb{1})$

- How to optimize these hyper-parameters ?

1) Grid / random search

- The predictive distribution contain "hyper-parameters"

model parameters

Likelihood hyper-parameters

prior hyper-parameters

$$P(y_*|x_*,\text{Data}) = \int d\vec{\theta} \, \underbrace{\begin{pmatrix}\text{Likelihood} \\ \text{of } y_* | x_*, \vec{\theta}\end{pmatrix}}_{P(y_*|x_*,\vec{\theta},\vec{\alpha})} \cdot \left\{\begin{pmatrix}\text{Posterior of} \\ \vec{\theta} \,|\, \text{Data}\end{pmatrix}\right\} \propto P(\text{Data}|\vec{\theta},\vec{\alpha})\, P(\vec{\theta}\,|\,\vec{\beta})$$

e.g. prior $= N(\vec{\theta}\,|\,\vec{\mu}, \sigma \cdot \mathbb{1})$

- How to optimize these hyper-parameters ?

1) Grid / random search $\Leftarrow$ curse of dimensionality

- The predictive distribution contain "hyper-parameters"

model parameters

Likelihood hyper-parameters

prior hyper-parameters

$$P(y_*|x_*,\text{Data}) = \int d\vec{\theta} \; \underbrace{\begin{pmatrix}\text{Likelihood} \\ \text{of } y_* | x_*, \vec{\theta}\end{pmatrix}}_{P(y_*|x_*,\vec{\theta},\vec{\alpha})} \cdot \left.\begin{pmatrix}\text{Posterior of} \\ \vec{\theta} \,|\, \text{Data}\end{pmatrix}\right\} \propto P(\text{Data}|\vec{\theta},\vec{\alpha})\, P(\vec{\theta}\,|\,\vec{\beta})$$

e.g. prior $= N(\vec{\theta}\,|\,\vec{\mu}, \sigma \cdot \mathbb{1})$

- How to optimize these hyper-parameters ?

1) Grid / random search ⟵ curse of dimensionality

2) "Bayesian optimisation" (new proposal based on previous evaluations)

- The predictive distribution contain "hyper-parameters"

model parameters

Likelihood hyper-parameters

prior hyper-parameters

$$P(y_*|x_*,\text{Data}) = \int d\vec{\theta}\, \underbrace{\begin{pmatrix}\text{Likelihood}\\ \text{of } y_*|x_*,\vec{\theta}\end{pmatrix}}_{P(y_*|x_*,\vec{\theta},\vec{\alpha})} \cdot \left.\begin{pmatrix}\text{Posterior of}\\ \vec{\theta}\,|\,\text{Data}\end{pmatrix}\right\} \propto P(\text{Data}|\vec{\theta},\vec{\alpha})\, P(\vec{\theta}\,|\,\vec{\beta})$$

e.g. $\text{prior} = N(\vec{\theta}\,|\,\vec{\mu}, \sigma\cdot\mathbb{1})$

- How to optimize these hyper-parameters?
  1) Grid / random search ⇐ curse of dimensionality
  2) "Bayesian optimisation" (new proposal based on previous evaluations) ⇐ Depends on extra tuning parameters

- The predictive distribution contain "hyper-parameters"

$$p(y_*|x_*,\text{Data}) = \int \underset{\text{model parameters}}{d\vec{\theta}} \underbrace{\begin{pmatrix}\text{Likelihood} \\ \text{of } y_* | x_*, \vec{\theta}\end{pmatrix}}_{p(y_*|x_*,\vec{\theta},\vec{\alpha})} \cdot \left\{ \begin{pmatrix}\text{Posterior of} \\ \vec{\theta} \mid \text{Data}\end{pmatrix} \right\} \propto p(\text{Data}|\vec{\theta},\underset{\text{Likelihood hyper-parameters}}{\vec{\alpha}})\, p(\vec{\theta}\,|\underset{\text{prior hyper-parameters}}{\vec{\beta}})$$

e.g. $\text{prior} = N(\vec{\theta}\,|\,\vec{\mu}, \sigma\cdot\mathbb{1})$

- How to optimize these hyper-parameters?
  1) Grid / random search ← curse of dimensionality
  2) "Bayesian optimisation" (new proposal based on previous evaluations) ← Depends on extra tuning parameters
  3) Maximizing an approximation of the model Evidence

- The predictive distribution contain "hyper-parameters"

$$p(y_*|x_*,\mathrm{Data}) = \int \overset{\text{model parameters}}{d\vec{\theta}}\; \underbrace{\begin{pmatrix}\text{Likelihood}\\ \text{of } y_* | x_*, \vec{\theta}\end{pmatrix}}_{p(y_*|x_*,\vec{\theta},\vec{\alpha})} \cdot \left.\begin{pmatrix}\text{Posterior of}\\ \vec{\theta}\ |\ \mathrm{Data}\end{pmatrix}\right\} \propto p(\mathrm{Data}|\vec{\theta},\underset{\text{Likelihood hyper-parameters}}{\vec{\alpha}})\, p(\vec{\theta}\,|\underset{\text{prior hyper-parameters}}{\vec{\beta}})$$

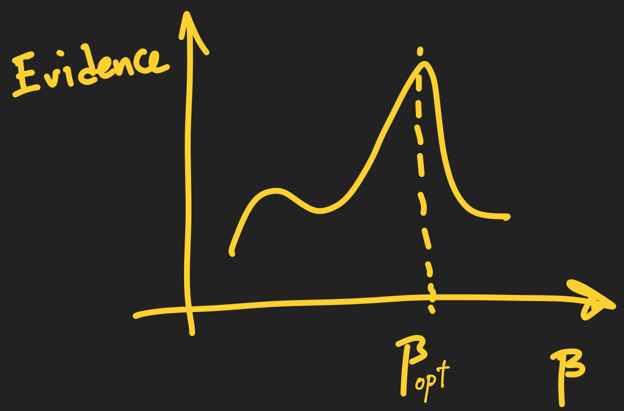
e.g. $\text{prior} = N(\vec{\theta}\,|\,\vec{\mu}, \sigma \cdot \mathbb{1})$

- How to optimize these hyper-parameters ?

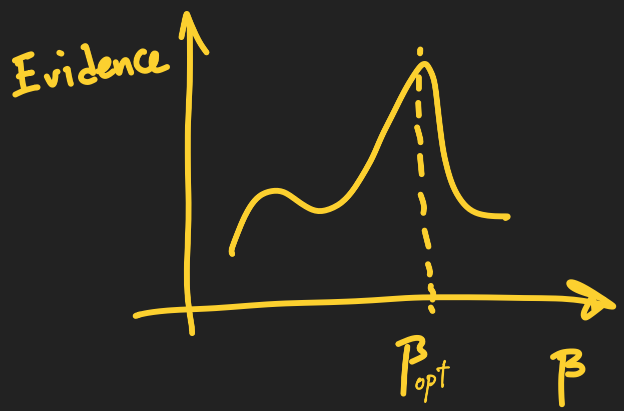
1) Grid / random search ⟵ Curse of dimensionality
2) "Bayesian optimisation" (new proposal based on previous evaluations) ⟵ Depends on extra tuning parameters
3) Maximizing an approximation of the model Evidence ⟵
   - No guarantees for robust predictions
   - Contribution to the predictive variance ?

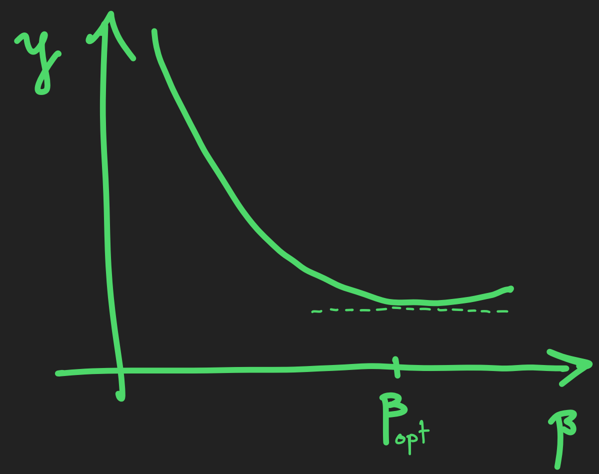
- What does it mean a good value for the hyper-parameters?

- What does it mean a good value for the hyper-parameters?
  - The one maximizing the model evidence?

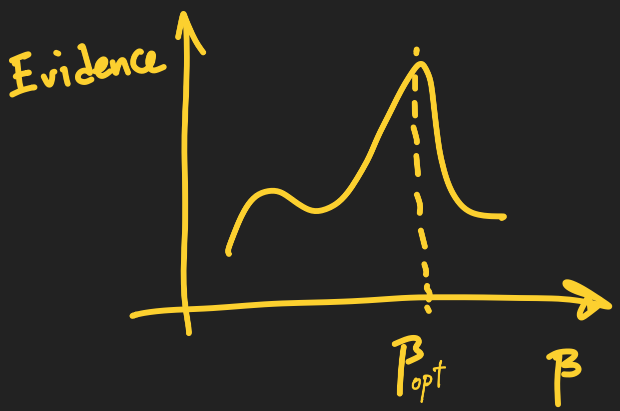


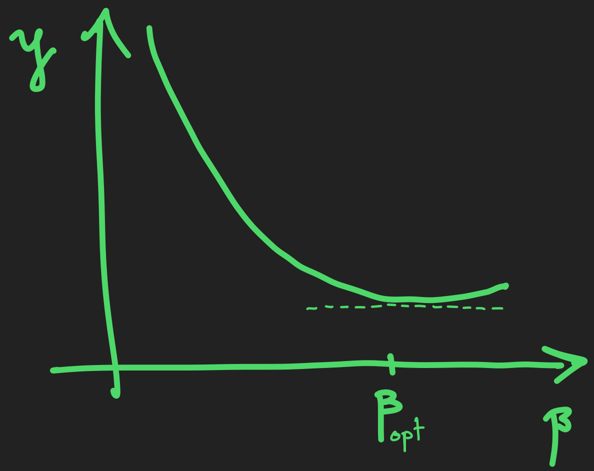
- What does it mean a good value for the hyper-parameters?
  - The one maximizing the model evidence?
  - The one ensuring robust predictions?





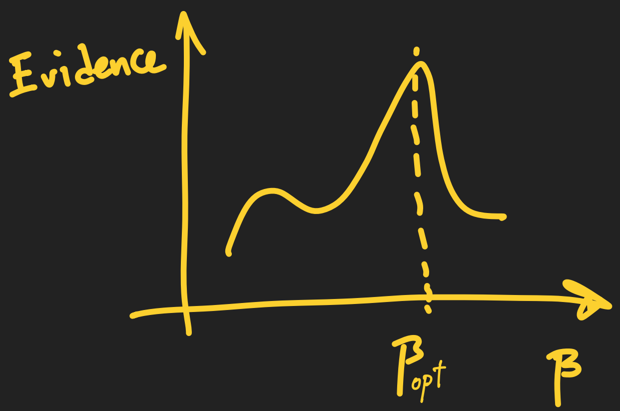
- What does it mean a good value for the hyper-parameters?
  - The one maximizing the model evidence?
  - The one ensuring robust predictions?

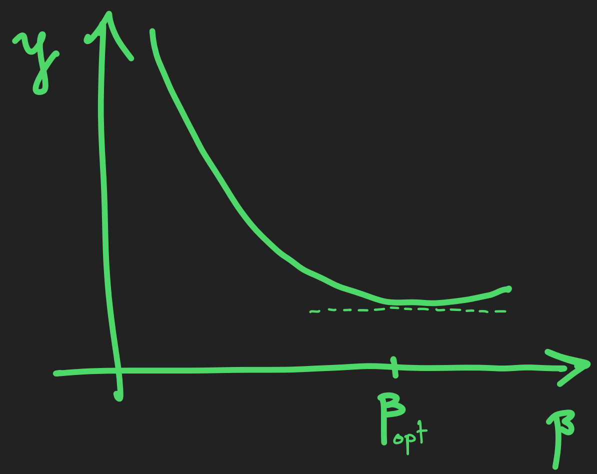




Depends on your scope!

- What does it mean a good value for the hyper-parameters?
  - The one maximizing the model evidence?
  - The one ensuring robust predictions?

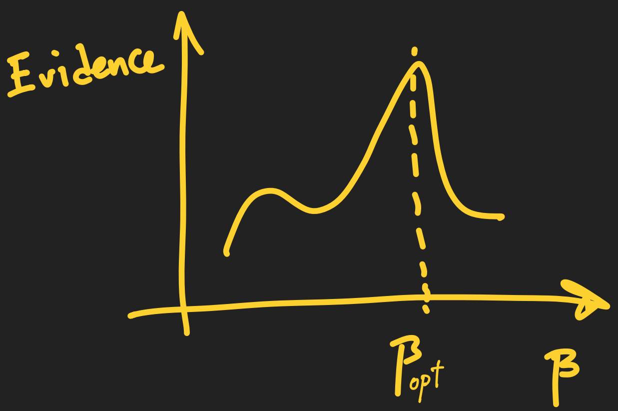


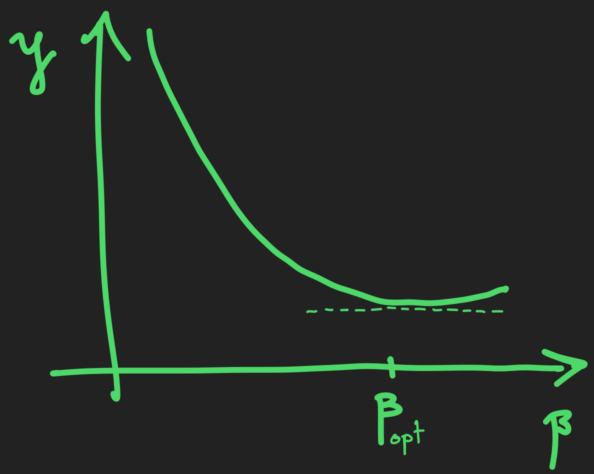


Depends on your scope!

- Here we follow the second approach

- What does it mean a good value for the hyper-parameters?
  - The one maximizing the model evidence?
  - The one ensuring robust predictions?

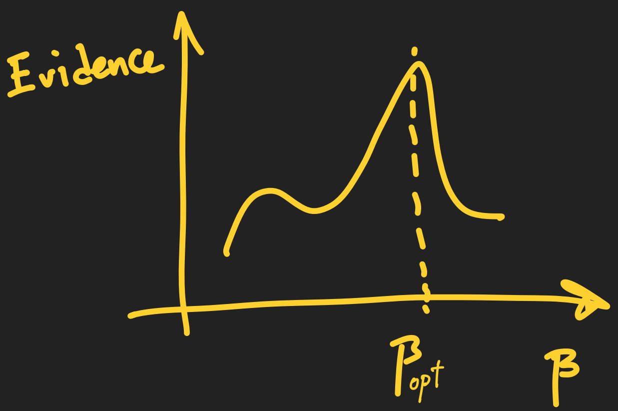


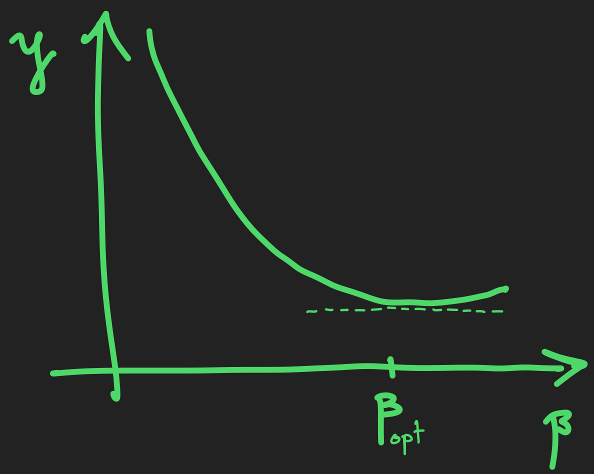


Depends on your scope!

- Here we follow the second approach
- Develop a methodology for estimating uncertainties associated with our ignorance about the hyper-p values

- What does it mean a good value for the hyper-parameters?
  - The one maximizing the model evidence?
  - The one ensuring robust predictions?





Depends on your scope!

Not addressed so far in the literature

- Here we follow the second approach
- Develop a methodology for estimating uncertainties associated with our ignorance about the hyper-p values

- Idea: Compute the derivative of the predictive distribution w.r.t. the hyper-parameters $\Rightarrow$ if very small, then predictions are robust

- Idea: Compute the derivative of the predictive distribution w.r.t. the hyper-parameters ⇒ if very small, then predictions are robust

$$P(y_* | x_*, \text{Data}) \overset{\lim N_s \to \infty}{=} \frac{1}{N_s} \sum_{s=1}^{N_s} P(y_* | x_*, \vec{\theta}_s, \vec{\alpha})$$

- Idea: Compute the derivative of the predictive distribution w.r.t. the hyper-parameters ⇒ if very small, then predictions are robust

$$P(y_*|x_*,\text{Data}) \overset{\lim N_s\to\infty}{=} \frac{1}{N_s}\sum_{s=1}^{N_s} P(y_*|x_*,\vec{\theta}_s,\vec{\alpha}) \quad \text{with } \vec{\theta}_s \sim \text{Posterior}(\vec{\theta}\,|\,\text{Data},\vec{\alpha},\vec{\beta})$$

MCMC

- Idea: Compute the derivative of the predictive distribution w.r.t. the hyper-parameters ⇒ if very small, then predictions are robust

$$P(y_*|x_*, \mathrm{Data}) \overset{\lim N_s \to \infty}{=} \frac{1}{N_s} \sum_{s=1}^{N_s} P(y_* | x_*, \vec{\theta}_s, \vec{\alpha}) \quad \text{with } \vec{\theta}_s \sim \mathrm{Posterior}(\vec{\theta} \,|\, \mathrm{Data}, \vec{\alpha}, \vec{\beta})$$

MCMC

explicit dependence on hyper-parameters is lost!

- Idea: Compute the derivative of the predictive distribution w.r.t. the hyper-parameters ⟹ if very small, then predictions are robust

$$p(y_*|x_*,\text{Data}) \overset{\lim N_s\to\infty}{=} \frac{1}{N_s}\sum_{s=1}^{N_s} p(y_*|x_*,\vec{\theta}_s,\vec{\alpha}) \quad \text{with } \vec{\theta}_s \sim \text{Posterior}(\vec{\theta}\,|\,\text{Data},\vec{\alpha},\vec{\beta})$$

MCMC

explicit dependence on hyper-parameters is lost!

$$\frac{\partial p(y_*|x_*,\text{Data})}{\partial \vec{\beta}} = \,?$$

One possible way : Numerical Stochastic Perturbation Theory
using HMC (borrowed from Lattice QCD)

One possible way using HMC: Numerical Stochastic Perturbation Theory (borrowed from Lattice QCD)

- Expand in Taylor series

hyper-parameter

$$\vec{\theta} = \vec{\theta}^{(0)} + \vec{\theta}^{(1)} \cdot \delta\beta + \dots$$

$$\vec{p} = \vec{p}^{(0)} + \vec{p}^{(1)} \cdot \delta\beta + \dots$$

One possible way using HMC: Numerical Stochastic Perturbation Theory (borrowed from Lattice QCD)

- Expand in Taylor series

hyper-parameter

$$\vec{\theta} = \vec{\theta}^{(0)} + \vec{\theta}^{(1)} \cdot \delta\beta + \ldots$$

$$\vec{p} = \vec{p}^{(0)} + \vec{p}^{(1)} \cdot \delta\beta + \ldots$$

$$\left\{ \begin{aligned} \dot{\vec{\theta}} &= \frac{\partial H}{\partial \vec{p}} \\ \dot{\vec{p}} &= -\frac{\partial H}{\partial \vec{\theta}} \end{aligned} \right.$$

One possible way : Numerical Stochastic Perturbation Theory
using HMC (borrowed from Lattice QCD)

- Expand in Taylor series

$$\vec{\theta} = \vec{\theta}^{(0)} + \vec{\theta}^{(1)} \cdot \delta \beta + \dots$$

hyper-parameter

$$\vec{p} = \vec{p}^{(0)} + \vec{p}^{(1)} \cdot \delta \beta + \dots$$

$$\begin{cases} \dot{\vec{\theta}} = \dfrac{\partial H}{\partial \vec{p}} \\ \dot{\vec{p}} = -\dfrac{\partial H}{\partial \vec{\theta}} \end{cases}$$

$$\frac{\partial H}{\partial \vec{p}} = \left(\frac{\partial H}{\partial \vec{p}}\right)^{(0)} + \left(\frac{\partial H}{\partial \vec{p}}\right)^{(1)} \cdot \delta \beta + \dots$$

One possible way: Numerical Stochastic Perturbation Theory using HMC (borrowed from Lattice QCD)

- Expand in Taylor series

hyper-parameter

$$\vec{\theta} = \vec{\theta}^{(0)} + \vec{\theta}^{(1)} \cdot \delta\beta + \dots$$

$$\vec{p} = \vec{p}^{(0)} + \vec{p}^{(1)} \cdot \delta\beta + \dots$$

$$\left\{ \begin{aligned} \dot{\vec{\theta}} &= \frac{\partial H}{\partial \vec{p}} \\ \dot{\vec{p}} &= -\frac{\partial H}{\partial \vec{\theta}} \end{aligned} \right.$$

$$\frac{\partial H}{\partial \vec{p}} = \left(\frac{\partial H}{\partial \vec{p}}\right)^{(0)} + \left(\frac{\partial H}{\partial \vec{p}}\right)^{(1)} \cdot \delta\beta + \dots \quad \left(\text{similarly with } \frac{\partial H}{\partial \vec{\theta}}\right)$$

One possible way: Numerical Stochastic Perturbation Theory
using HMC (borrowed from Lattice QCD)

- Expand in Taylor series

$$\vec{\theta} = \vec{\theta}^{(0)} + \vec{\theta}^{(1)} \cdot \delta\beta + \dots$$

hyper-parameter

$$\vec{p} = \vec{p}^{(0)} + \vec{p}^{(1)} \cdot \delta\beta + \dots$$

$$\left\{ \begin{aligned} \dot{\vec{\theta}} &= \frac{\partial H}{\partial \vec{p}} \\ \dot{\vec{p}} &= -\frac{\partial H}{\partial \vec{\theta}} \end{aligned} \right.$$

$$\frac{\partial H}{\partial \vec{p}} = \left(\frac{\partial H}{\partial \vec{p}}\right)^{(0)} + \left(\frac{\partial H}{\partial \vec{p}}\right)^{(1)} \cdot \delta\beta + \dots$$ (similarly with $\frac{\partial H}{\partial \vec{\theta}}$)

- Results in a tower of Hamilton eqs. up to given truncation order

One possible way : Numerical Stochastic Perturbation Theory
using HMC (borrowed from Lattice QCD)

- Expand in Taylor series

$$\vec{\theta} = \vec{\theta}^{(0)} + \vec{\theta}^{(1)} \cdot \delta\beta + \ldots$$

hyper-parameter

$$\vec{p} = \vec{p}^{(0)} + \vec{p}^{(1)} \cdot \delta\beta + \ldots$$

$$\left\{ \begin{array}{l} \dot{\vec{\theta}} = \dfrac{\partial H}{\partial \vec{p}} \\ \dot{\vec{p}} = -\dfrac{\partial H}{\partial \vec{\theta}} \end{array} \right.$$

$$\frac{\partial H}{\partial \vec{p}} = \left(\frac{\partial H}{\partial \vec{p}}\right)^{(0)} + \left(\frac{\partial H}{\partial \vec{p}}\right)^{(1)} \cdot \delta\beta + \ldots$$ (similarly with $\dfrac{\partial H}{\partial \vec{\theta}}$)

- Results in a tower of Hamilton eqs. up to given truncation order

$$\left. \begin{array}{l} \dot{\vec{\theta}}^{(n)} = \left(\dfrac{\partial H}{\partial \vec{p}}\right)^{(n)} \\ \dot{\vec{p}}^{(n)} = -\left(\dfrac{\partial H}{\partial \vec{\theta}}\right)^{(n)} \end{array} \right\}$$

$n = 1, \ldots, N_{trunc}$

One possible way: Numerical Stochastic Perturbation Theory using HMC (borrowed from Lattice QCD)

$$\left\{ \dot{\vec{\theta}} = \frac{\partial H}{\partial \vec{p}} \quad \dot{\vec{p}} = -\frac{\partial H}{\partial \vec{\theta}} \right.$$

- Expand in Taylor series

$$\vec{\theta} = \vec{\theta}^{(0)} + \vec{\theta}^{(1)} \cdot \delta\beta + \ldots$$

hyper-parameter

$$\vec{p} = \vec{p}^{(0)} + \vec{p}^{(1)} \cdot \delta\beta + \ldots$$

$$\frac{\partial H}{\partial \vec{p}} = \left(\frac{\partial H}{\partial \vec{p}}\right)^{(0)} + \left(\frac{\partial H}{\partial \vec{p}}\right)^{(1)} \cdot \delta\beta + \ldots \quad \left(\text{similarly with } \frac{\partial H}{\partial \vec{\theta}}\right)$$

- Results in a tower of Hamilton eqs. up to given truncation order

$$\left.\begin{aligned} \dot{\vec{\theta}}^{(n)} &= \left(\frac{\partial H}{\partial \vec{p}}\right)^{(n)} \\ \dot{\vec{p}}^{(n)} &= -\left(\frac{\partial H}{\partial \vec{\theta}}\right)^{(n)} \end{aligned}\right\} \quad n = 1, \ldots, N_{trunc}$$

$$\Rightarrow \quad p(y_* | x_*, \text{Data}) \equiv P_*$$

$$P_* = P_*^{(0)} + P^{(1)} \cdot \delta\beta + \ldots$$

One possible way : Numerical Stochastic Perturbation Theory
using HMC (borrowed from Lattice QCD)

- Expand in Taylor series

$$\vec{\theta} = \vec{\theta}^{(0)} + \vec{\theta}^{(1)} \cdot \delta \underbrace{\beta}_{\text{hyper-parameter}} + \dots$$

$$\vec{p} = \vec{p}^{(0)} + \vec{p}^{(1)} \cdot \delta \beta + \dots$$

$$\left\{ \begin{aligned} \dot{\vec{\theta}} &= \frac{\partial H}{\partial \vec{p}} \\ \dot{\vec{p}} &= -\frac{\partial H}{\partial \vec{\theta}} \end{aligned} \right.$$

$$\frac{\partial H}{\partial \vec{p}} = \left(\frac{\partial H}{\partial \vec{p}}\right)^{(0)} + \left(\frac{\partial H}{\partial \vec{p}}\right)^{(1)} \cdot \delta \beta + \dots \quad \left(\text{similarly with } \frac{\partial H}{\partial \vec{\theta}}\right)$$

- Results in a tower of Hamilton eqs. up to given truncation order

$$\left.\begin{aligned} \dot{\vec{\theta}}^{(n)} &= \left(\frac{\partial H}{\partial \vec{p}}\right)^{(n)} \\ \dot{\vec{p}}^{(n)} &= -\left(\frac{\partial H}{\partial \vec{\theta}}\right)^{(n)} \end{aligned}\right\} \quad n = 1, \dots, N_{trunc}$$

$$\Rightarrow \quad p(y_* | x_*, \text{Data}) \equiv P_*$$

$$P_* = P_*^{(0)} + \underline{\underline{P^{(1)}}} \cdot \delta \beta + \dots$$

this is what we are searching for

This can be implemented using a single HMC simulation!

This can be implemented using a single HMC simulation!

[with N_{pert} coupled Hamilton eqs.]

This can be implemented using a single HMC simulation!

[with N_{pert} coupled Hamilton eqs.]

the magic is possible thanks to the concept of "Automatic Differentiation" and "Hyper-dual #s"

(see A. Ramos, 1809.01289)

This can be implemented using a single HMC simulation!

[with N_{pert} coupled Hamilton eqs.]

the magic is possible thanks to the concept of "Automatic Differentiation" and "Hyper-dual #s"

(see A. Ramos, 1809.01289)

- Toy example:

This can be implemented using a single HMC simulation!

[with N_{pert} coupled Hamilton eqs.]

the magic is possible thanks to the concept of "Automatic Differentiation" and "Hyper-dual #s"

(see A. Ramos, 1809.01289)

- Toy example:
  - Gaussian data generated by

$$f(x) = x^3 + x^2 + x + 1$$

This can be implemented using a single HMC simulation!

[with N_{pert} coupled Hamilton eqs.]

the magic is possible thanks to the concept of "Automatic Differentiation" and "Hyper-dual #s"

(see A. Ramos, 1809.01289)

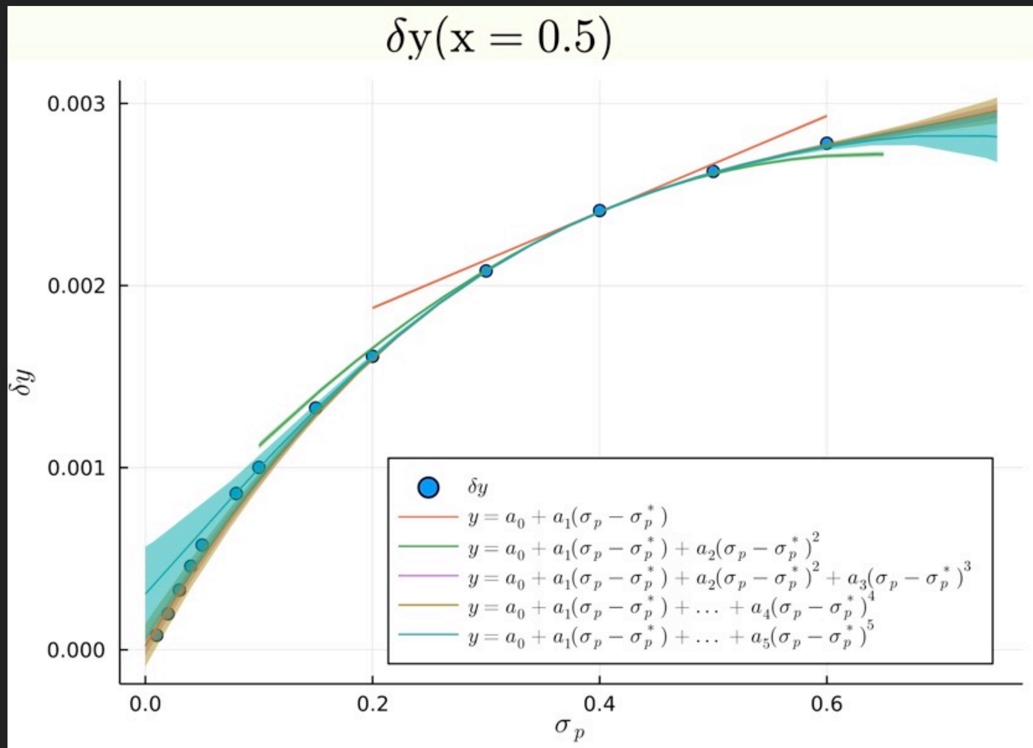
- Toy example:
  - Gaussian data generated by $f(x) = x^3 + x^2 + x + 1$
  - Model is a 1-hidden layer Bayesian Neural net with prior variance $= \sigma_p^2$

This can be implemented using a single HMC simulation!

[with Npert coupled Hamilton eqs.]

the magic is possible thanks to the concept of "Automatic Differentiation" and "Hyper-dual #s"

(see A. Ramos, 1809.01289)



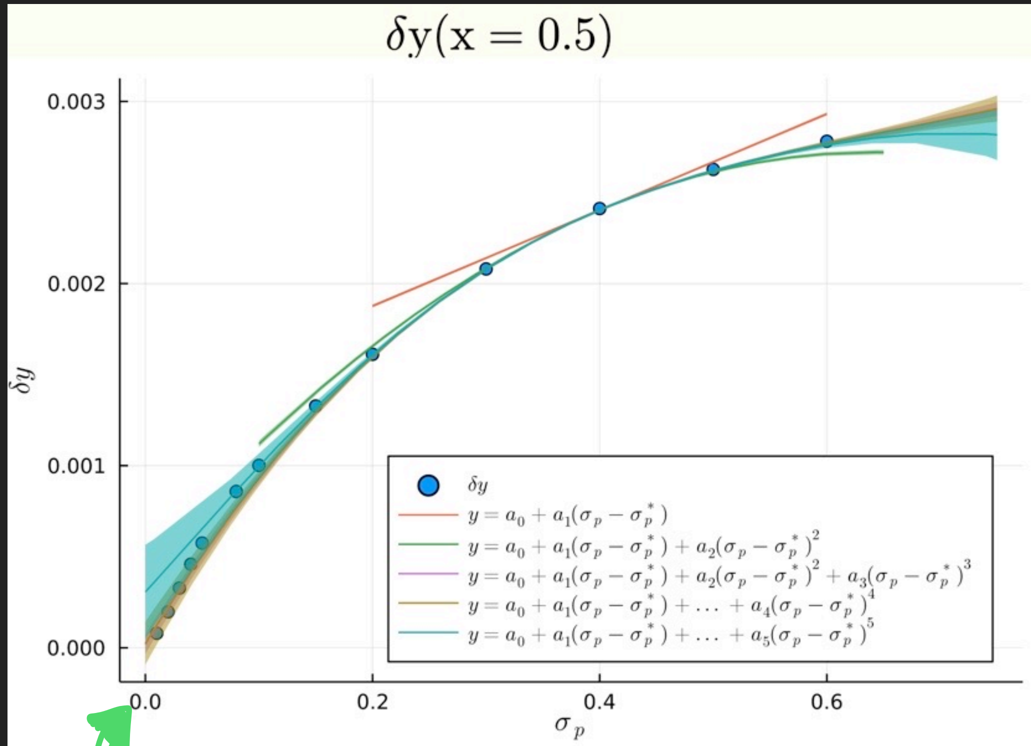
- Toy example:
  - Gaussian data generated by $f(x) = x^3 + x^2 + x + 1$
  - Model is a 1-hidden layer Bayesian Neural net with prior variance $= \sigma_p^2$

This can be implemented using a single HMC simulation!

[with Npert coupled Hamilton eqs.]

the magic is possible thanks to the concept of "Automatic Differentiation" and "Hyper-dual #s"

(see A. Ramos, 1809.01289)

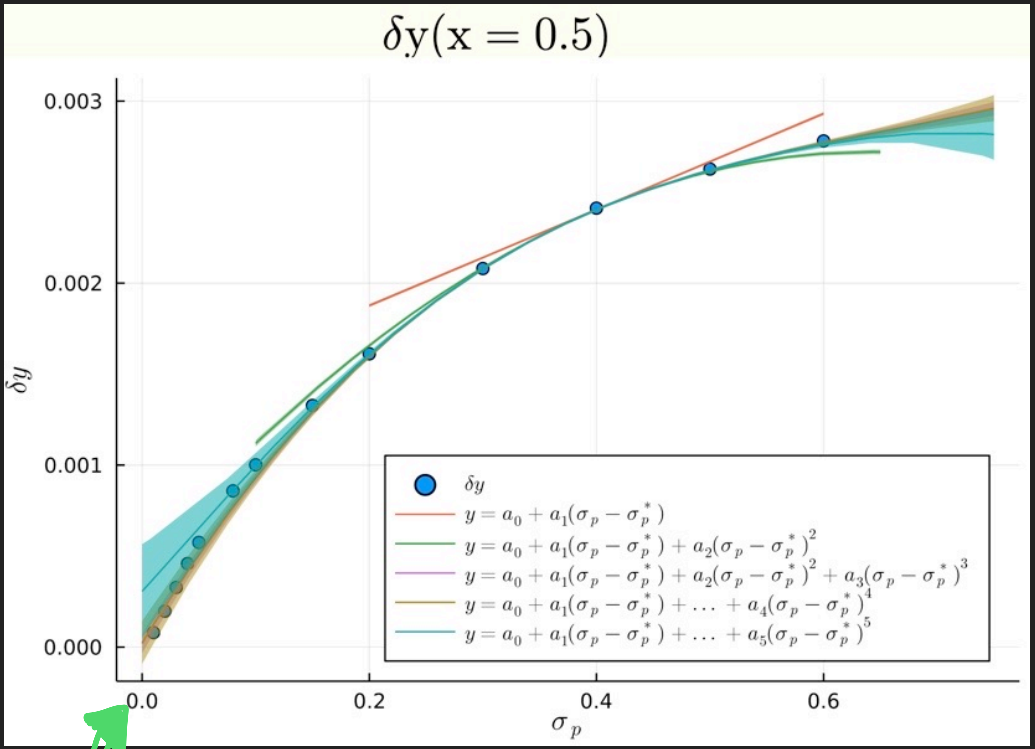


Results: The expansion captures successfully the actual behavior!

- Toy example:
  - Gaussian data generated by $f(x) = x^3 + x^2 + x + 1$
  - Model is a 1-hidden layer Bayesian Neural net with prior variance $= \sigma_p^2$

This can be implemented using a single HMC simulation!
[with Npert coupled Hamilton eqs.]

the magic is possible thanks to the concept of "Automatic Differentiation" and "Hyper-dual #s"
(see A. Ramos, 1809.01289)



- Toy example:
  - Gaussian data generated by $f(x) = x^3 + x^2 + x + 1$
  - Model is a 1-hidden layer Bayesian Neural net with prior variance $= \sigma_p^2$

Results: The expansion captures successfully the actual behavior!

Clearly this works

FINAL TAKE-HOME MESSAGE

FINAL TAKE-HOME MESSAGE

Bayesian Inference is in a golden era nowadays

FINAL TAKE-HOME MESSAGE

Bayesian Inference is in a golden era nowadays

Plenty of opportunities for two-way contributions

FINAL TAKE-HOME MESSAGE

Bayesian Inference is in a golden era nowadays

Plenty of opportunities for two-way contributions

$\boxed{\text{Physics, Biology, etc}}$ $\boxed{\text{Bayesian Inference, Machine learning}}$

FINAL TAKE-HOME MESSAGE

Bayesian Inference is in a golden era nowadays

Plenty of opportunities for two-way contributions

Physics, Biology, etc

Bayesian Inference, Machine learning

FINAL TAKE-HOME MESSAGE

Bayesian Inference is in a golden era nowadays

Plenty of opportunities for two-way contributions

Physics, Biology, etc ⇄ Bayesian Inference, Machine Learning