

Foundations of Probabilistic Machine Learning

Neill DF Campbell

Bath ML Summer School, 6 August 2026

Department of Computer Science, University College London

(Slide input/ideas credits: Simon Prince, Carl Henrik Ek, Javier González, Mike Tipping, David MacKay)



Dangers of avoiding probabilistic approaches...

DATA

Which dataset do you want to use?



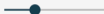
Ratio of training to test data: 50%



Noise: 0



Batch size: 10



REGENERATE

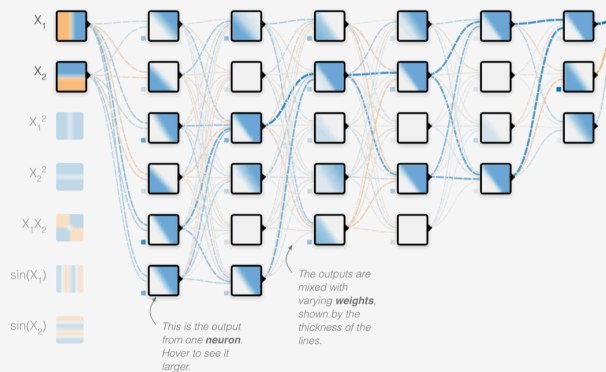
FEATURES

Which properties do you want to feed in?

- X_1
- X_2
- X_1^2
- X_2^2
- $X_1 X_2$
- $\sin(X_1)$
- $\sin(X_2)$

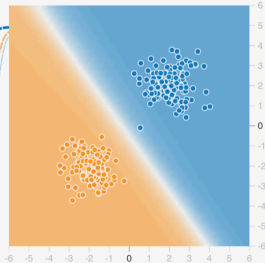
6 HIDDEN LAYERS

6 neurons 6 neurons 5 neurons 5 neurons 4 neurons 3 neurons

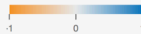


OUTPUT

Test loss 0.002
Training loss 0.001



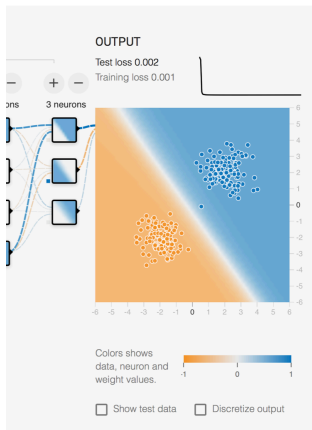
Colors shows data, neuron and weight values.



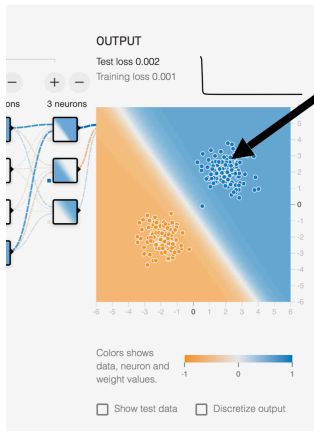
☐ Show test data

☐ Discretize output

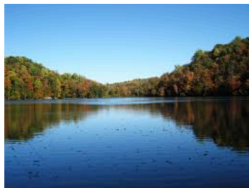
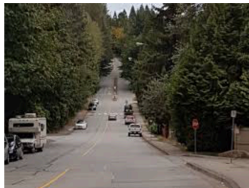
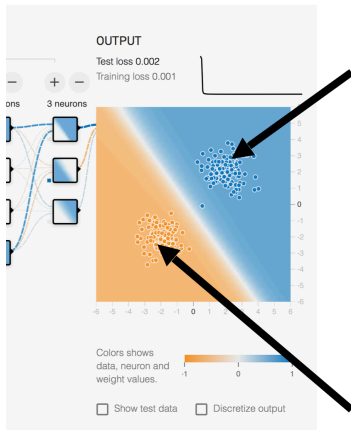
Dangers of avoiding probabilistic approaches...



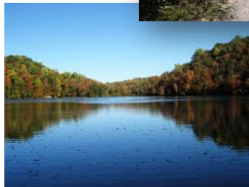
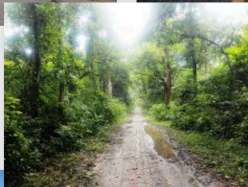
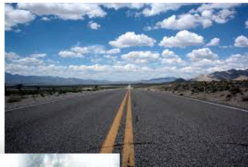
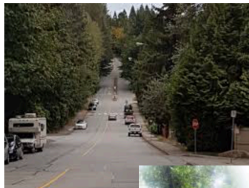
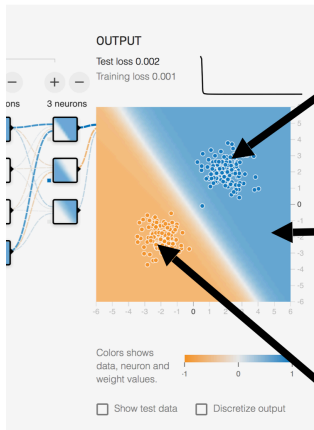
Dangers of avoiding probabilistic approaches...



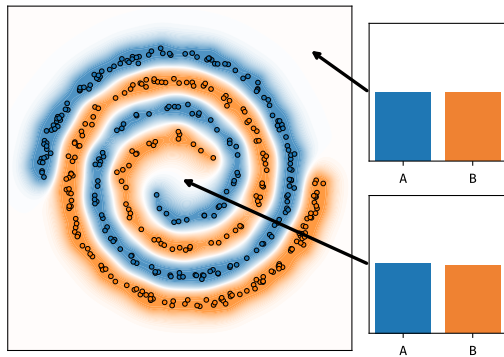
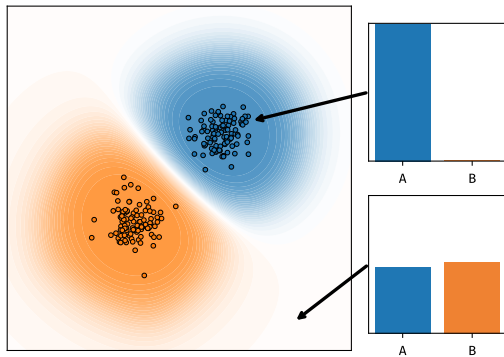
Dangers of avoiding probabilistic approaches...



Dangers of avoiding probabilistic approaches...



What if we use a probabilistic approach?

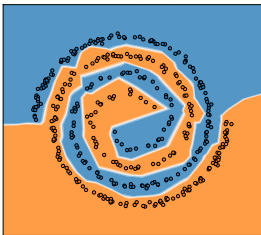


We need to consider the **properties** of
our Machine Learning
models/approaches..

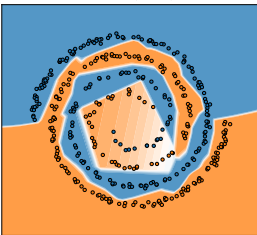
Spiral Dataset with an MLP: Different initialisations (of the weights)..

Same data, varying MLP init, MLP: 32 x 2

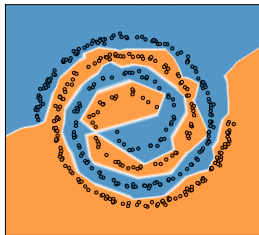
init_seed=0 (acc=1.00)



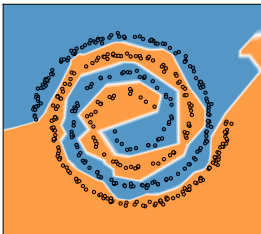
init_seed=1 (acc=0.96)



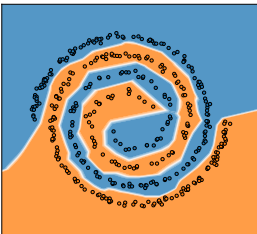
init_seed=2 (acc=1.00)



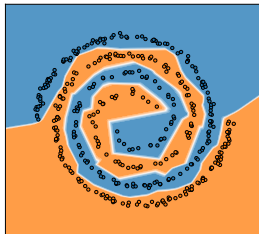
init_seed=3 (acc=1.00)



init_seed=4 (acc=1.00)



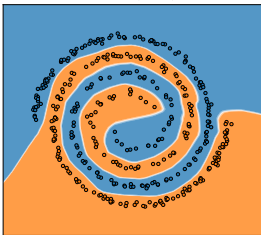
init_seed=5 (acc=1.00)



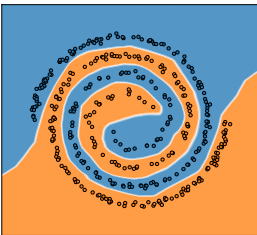
Spiral Dataset with an MLP: Different initialisations (of the weights)..

Same data, varying MLP init, MLP: 64 x 4

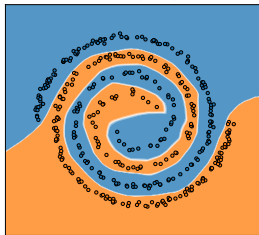
init_seed=0 (acc=1.00)



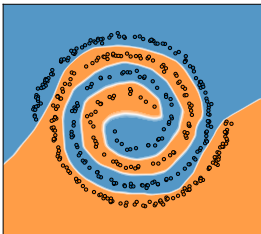
init_seed=1 (acc=1.00)



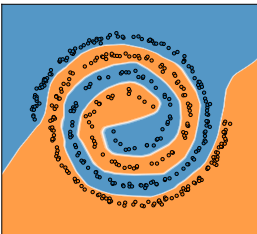
init_seed=2 (acc=1.00)



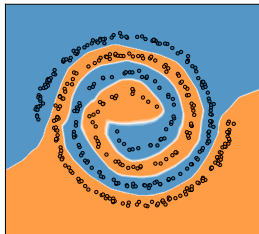
init_seed=3 (acc=1.00)



init_seed=4 (acc=1.00)



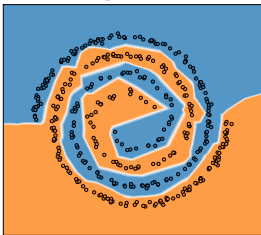
init_seed=5 (acc=1.00)



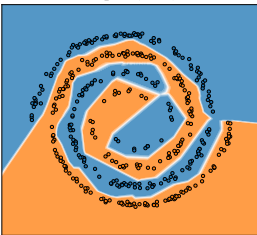
Spiral Dataset with an MLP: Different datasets..

Varying dataset, fixed MLP initialization

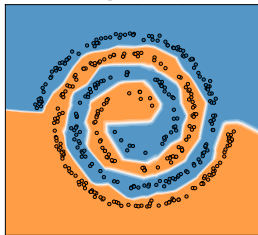
data_seed=0 (acc=1.00)



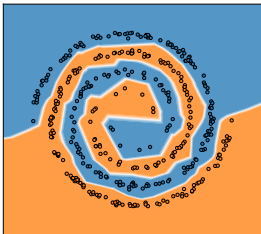
data_seed=1 (acc=1.00)



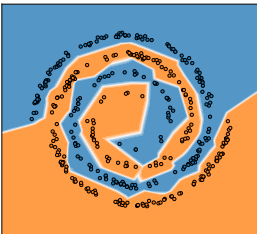
data_seed=2 (acc=1.00)



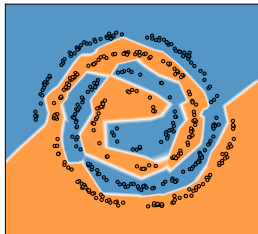
data_seed=3 (acc=1.00)



data_seed=4 (acc=1.00)

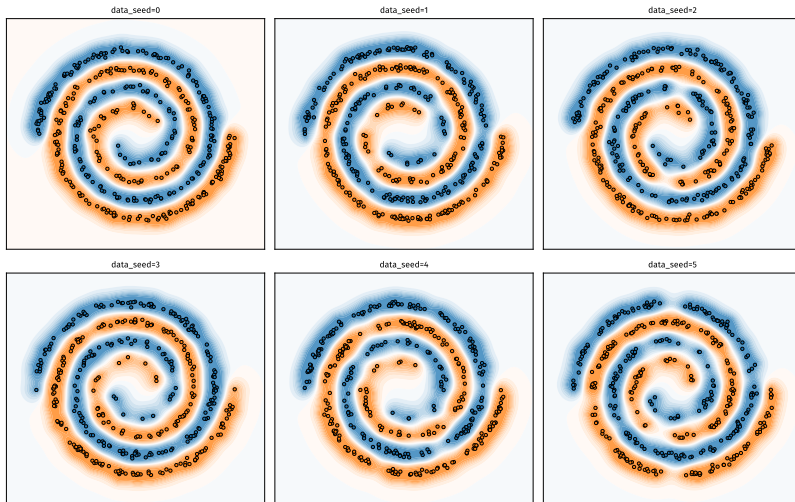


data_seed=5 (acc=1.00)



Spiral Dataset with a Bayesian (GP) model: Different datasets..

Varying dataset, GP classifier (RBF kernel, Bernoulli likelihood)



What do you hope to take away from this session?

Common Questions?

What questions do we have about ML?

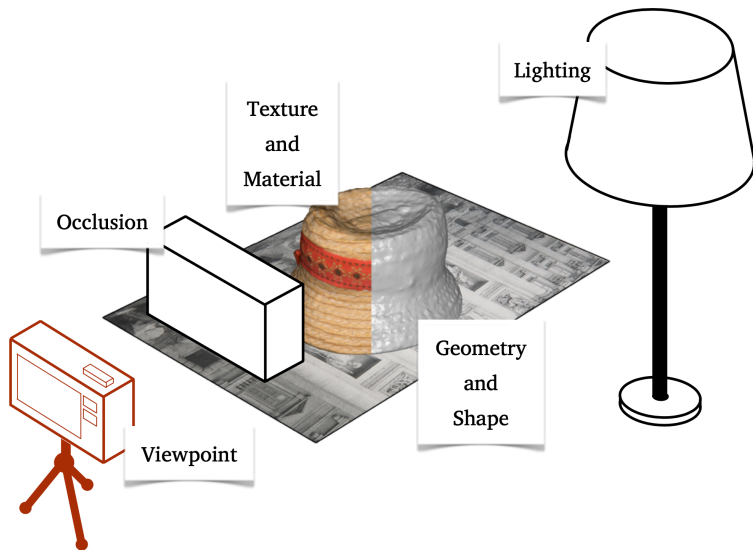
- Can I use ML to solve x ?
- What does ML actually do?
- Isn't ML just the same as y ?
- Can I replace myself/my research team with ML?
- How much data do I need?
- Can I just use Deep Learning/Generative AI/ChatGPT?
- Surely Deep Learning/Generative AI/ChatGPT is all hype?
- Can any of this be used for science/engineering?

Things to consider..

- Are all datasets equal or how to choose your data?
- Gotchas: What Machine Learning can and can't do for you.
- Average vs worst case Machine Learning
- Machine Learning and Causality
- Trade-offs in Machine Learning

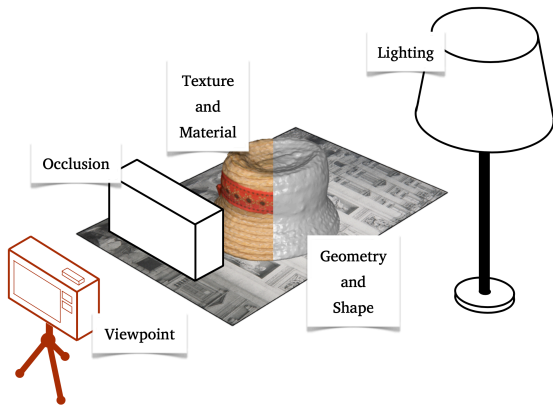
Motivation

Why do we need to care about Uncertainty in ML?



Why do we need to care about Uncertainty in ML?

- Ambiguity in the task
- Ambiguity in our models
- Downstream decision making
 - Safe
 - Robust
 - Transparent
- Improved performance
 - Data efficiency
 - Self-supervision
- **Evaluation and Model Selection!!**



Foundations in these two talks then Ivor will talk about some of the latest work in this field and some applications!

No Free Lunch

Overview...

Motivation

No Free Lunch

Whirlwind Introduction to Inverse Probabilities

Model Selection

Gaussian Processes (a Prior over Functions)

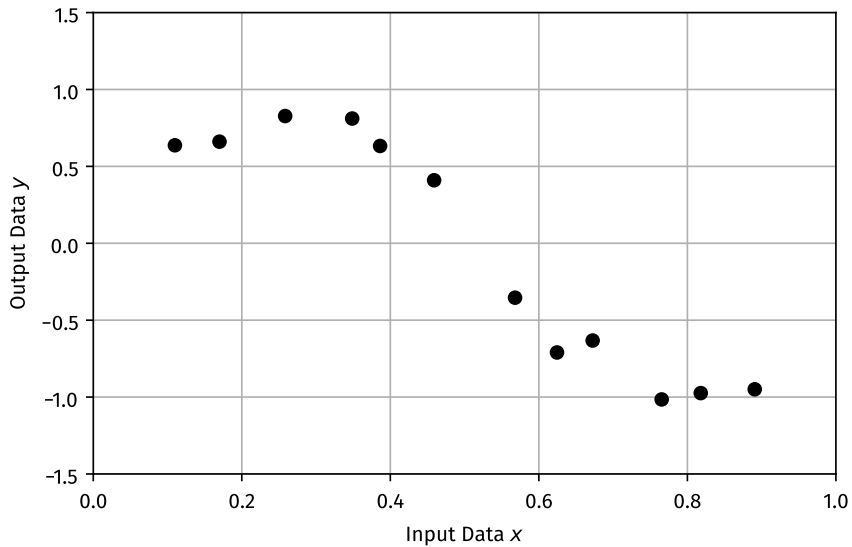
Application: Bayesian Optimisation

Evaluation

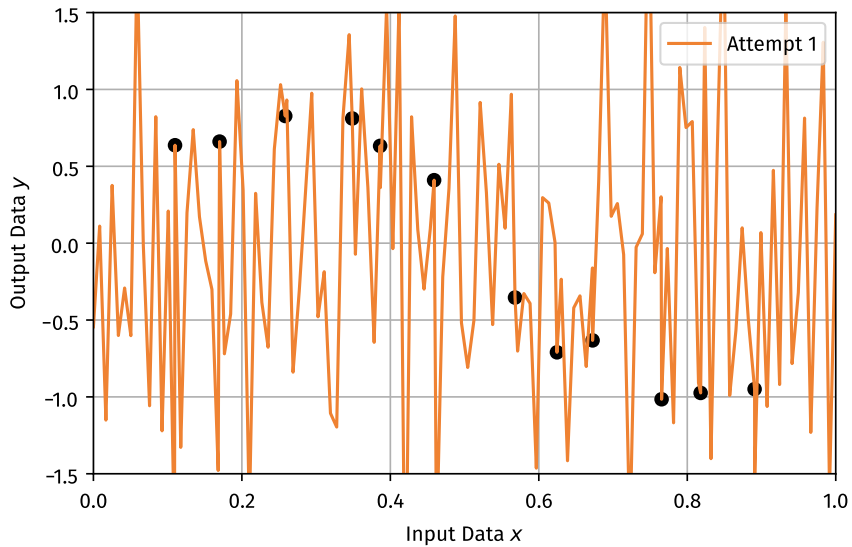
Conclusions

What happens between the dots?

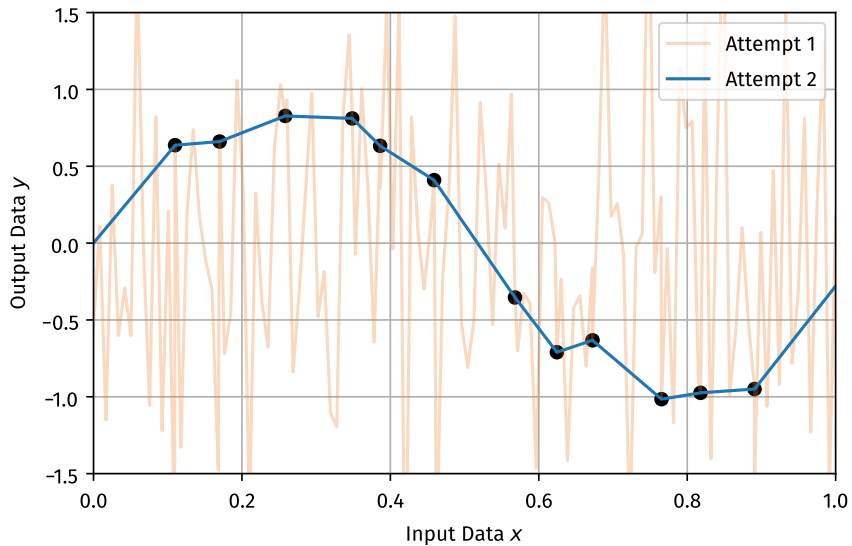
What happens between the dots?



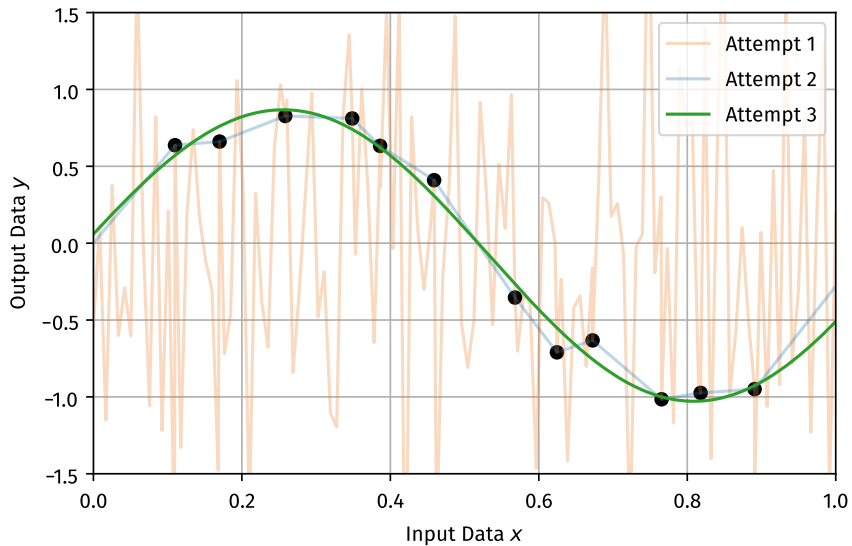
What happens between the dots?



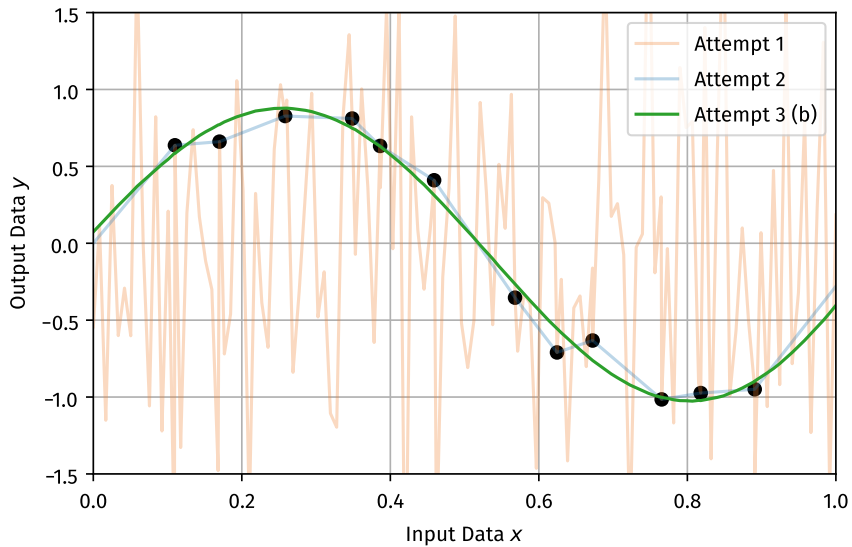
What happens between the dots?



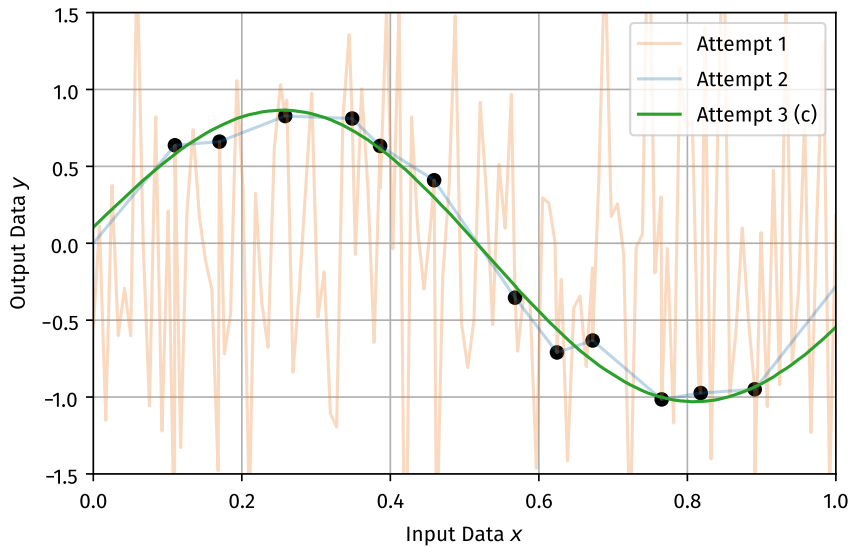
What happens between the dots?



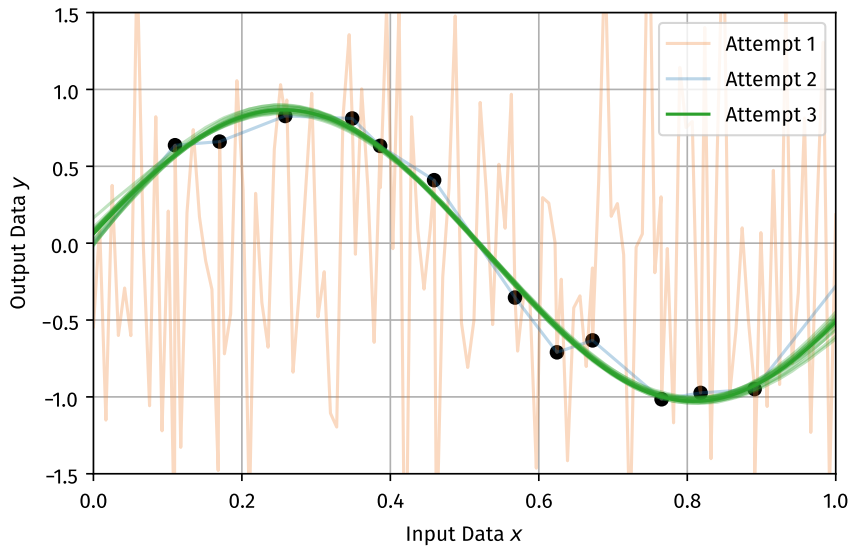
What happens between the dots?



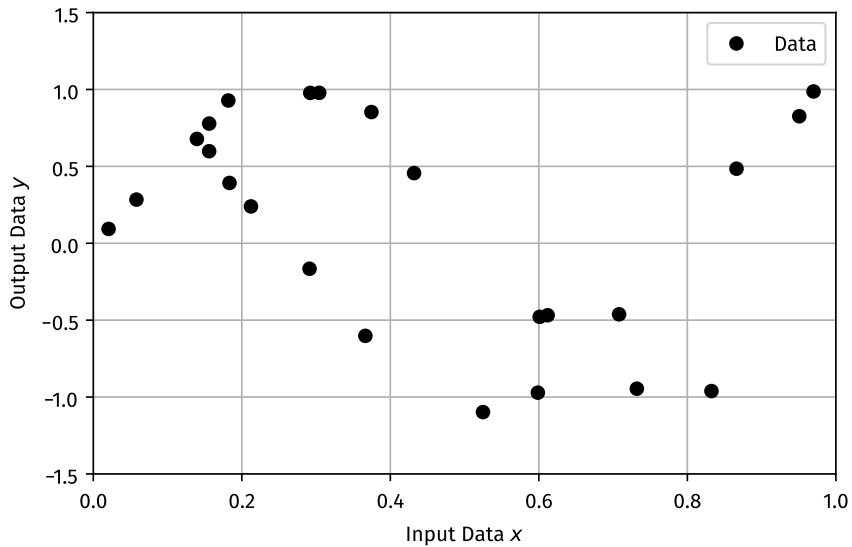
What happens between the dots?



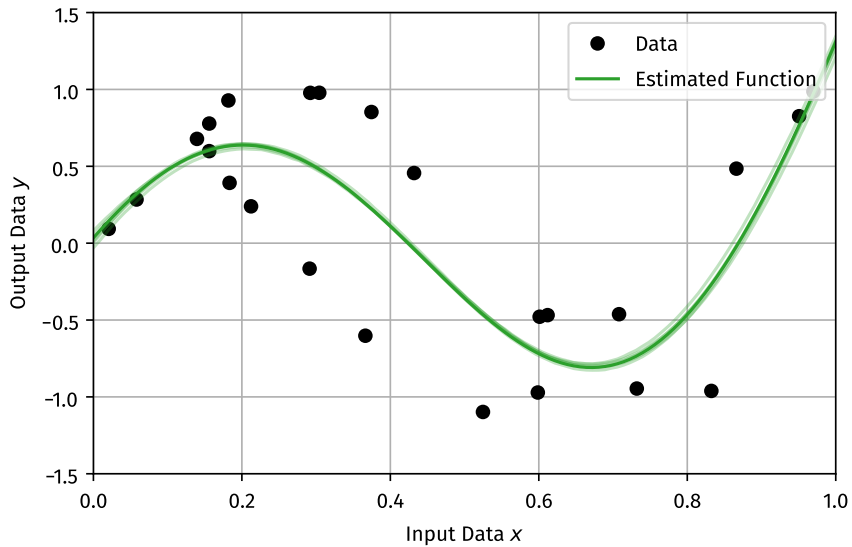
What happens between the dots?



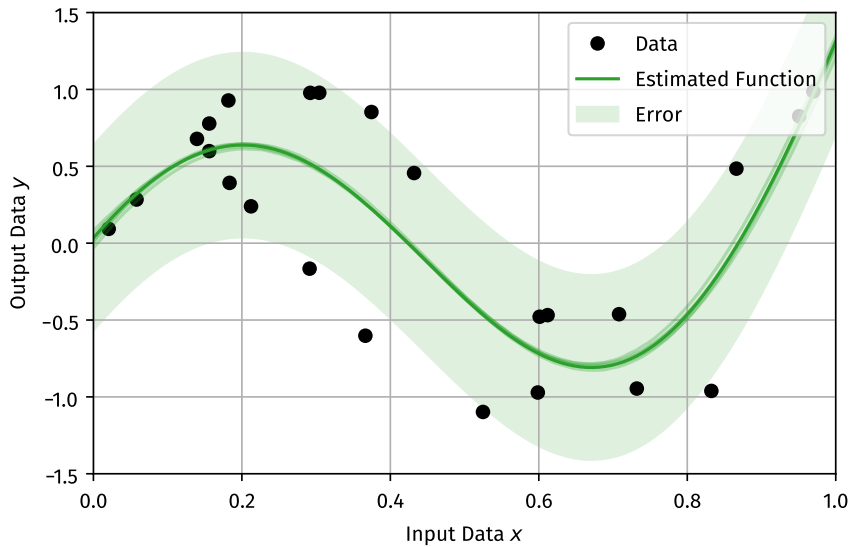
Ambiguity..



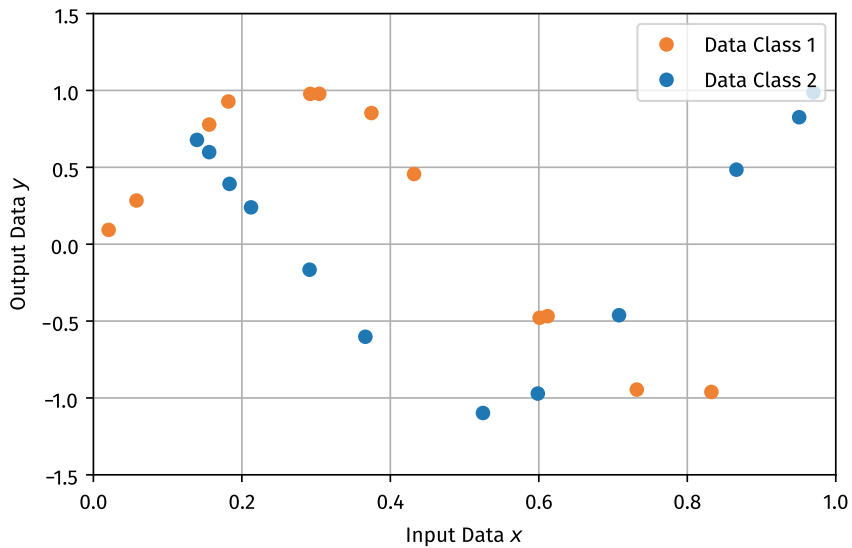
Ambiguity..



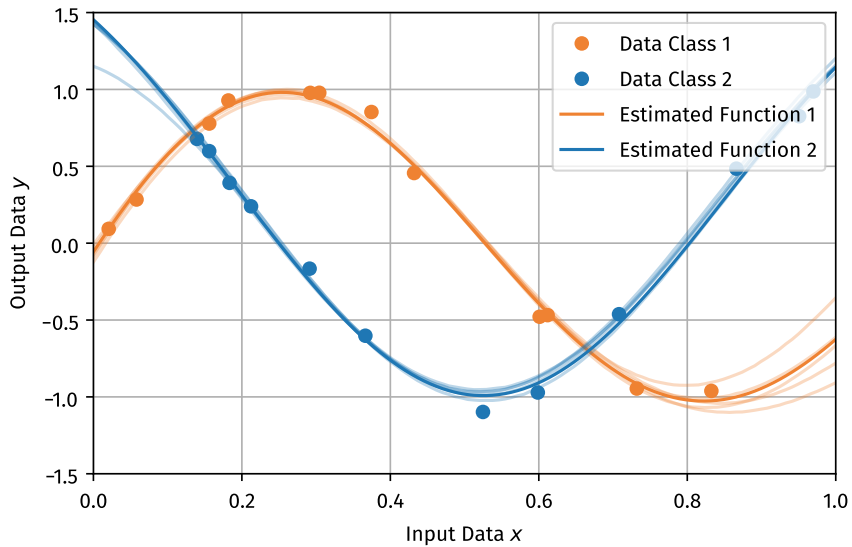
Ambiguity..



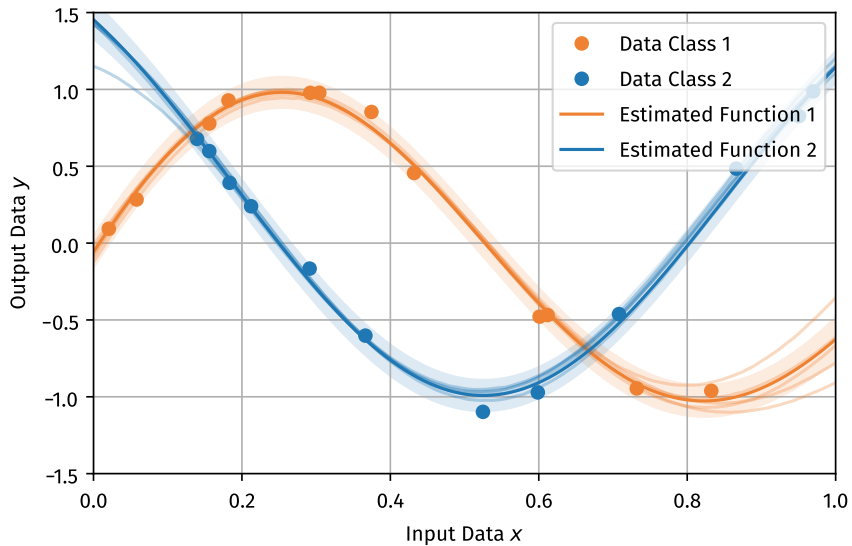
Ambiguity..



Ambiguity..

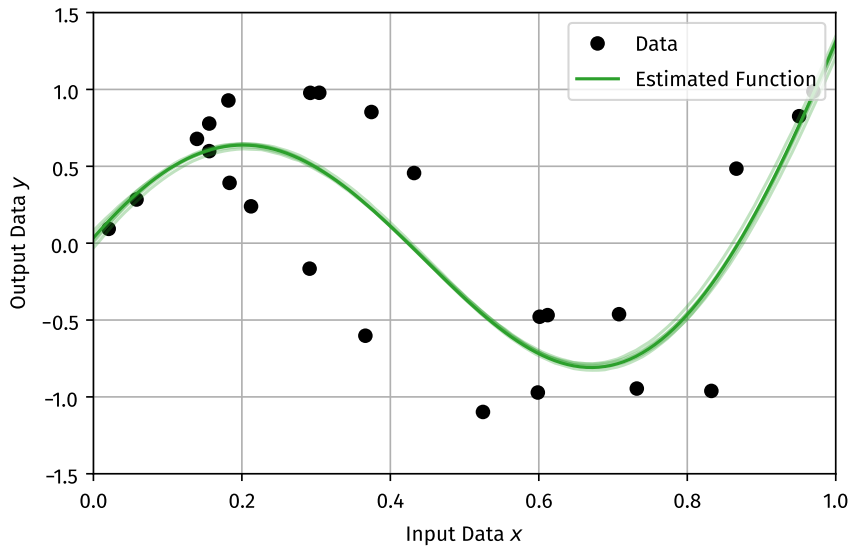


Ambiguity..

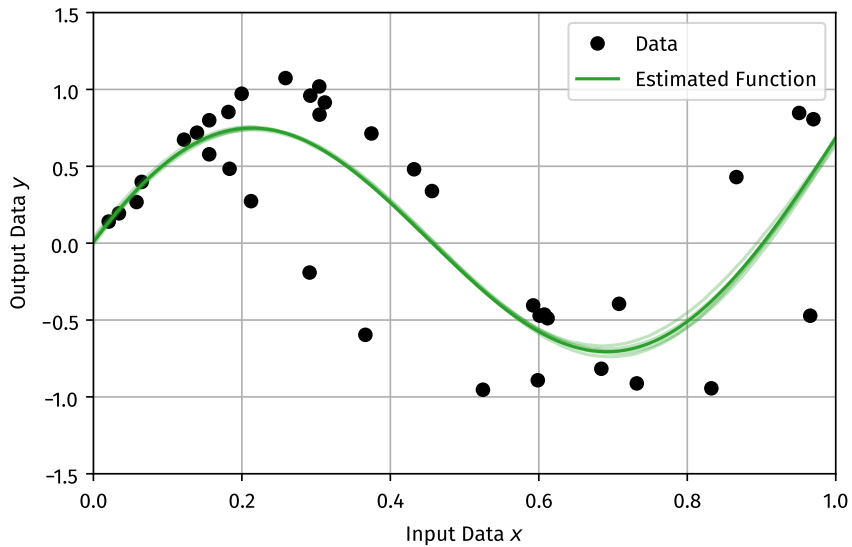


Average vs Worst Case..

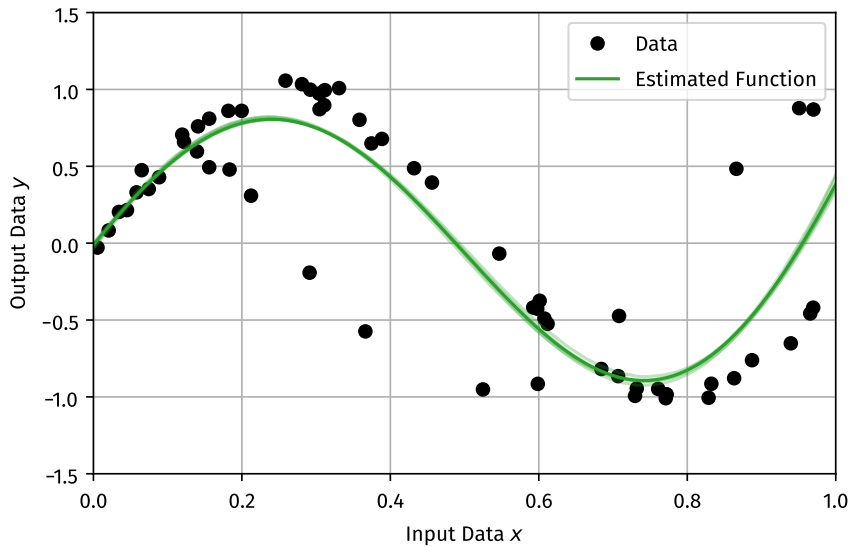
Average vs Worst Case..



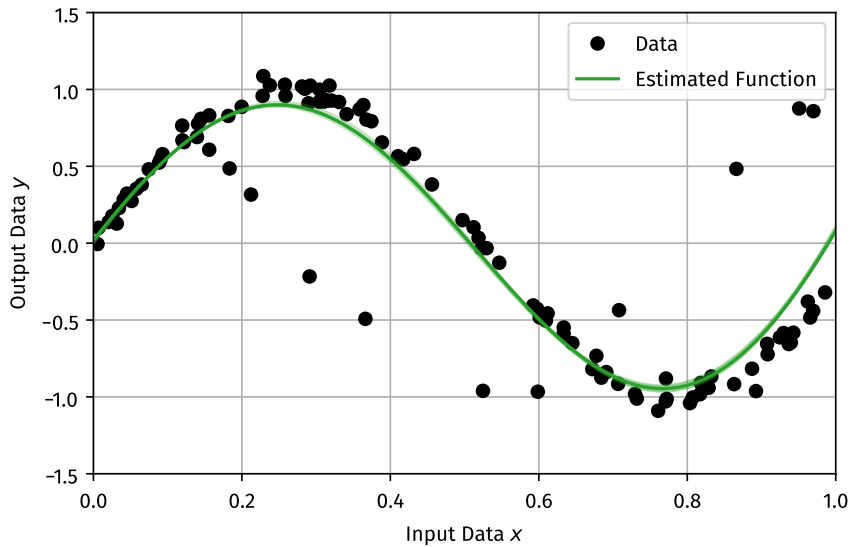
Average vs Worst Case..



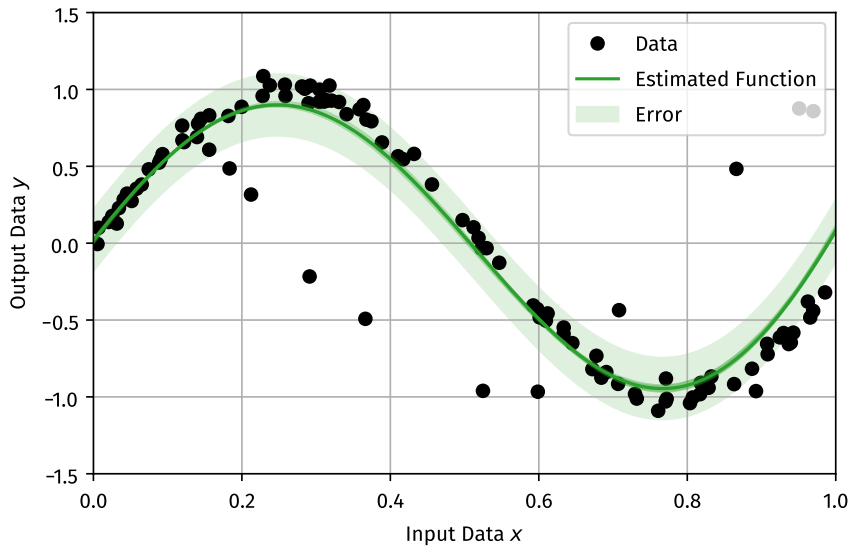
Average vs Worst Case..



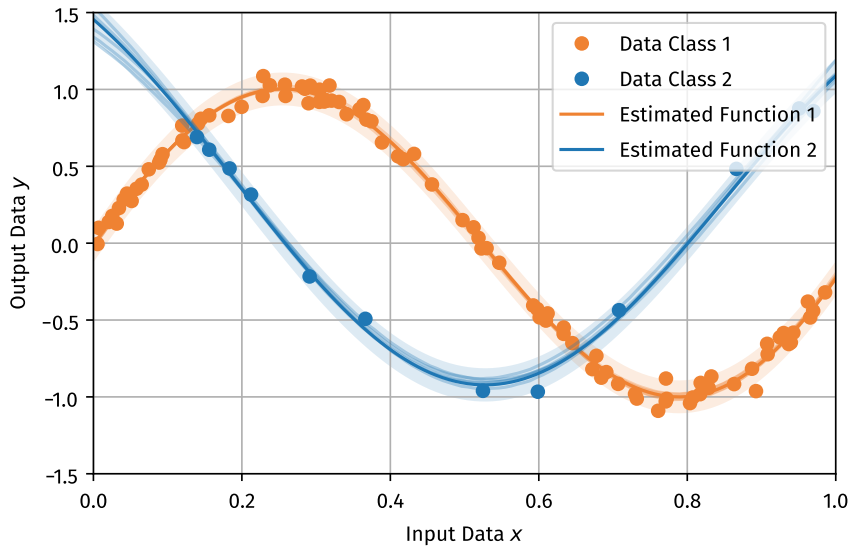
Average vs Worst Case..



Average vs Worst Case: Failure to model..

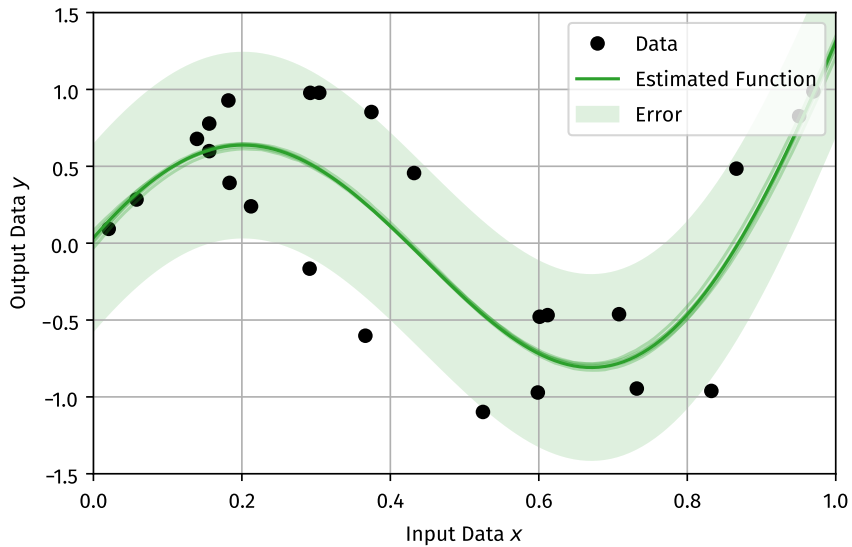


Average vs Worst Case: Explicitly accounting for imbalance..

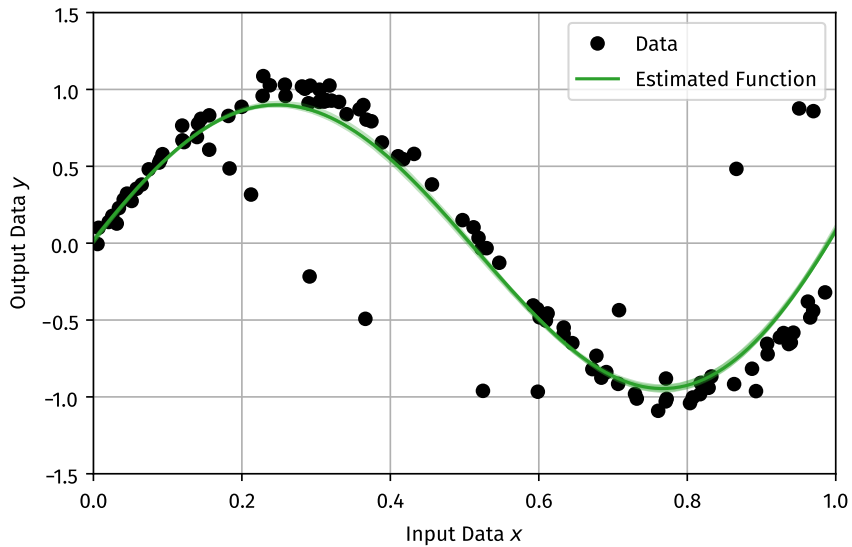


Story of Machine Learning (in three slides!)

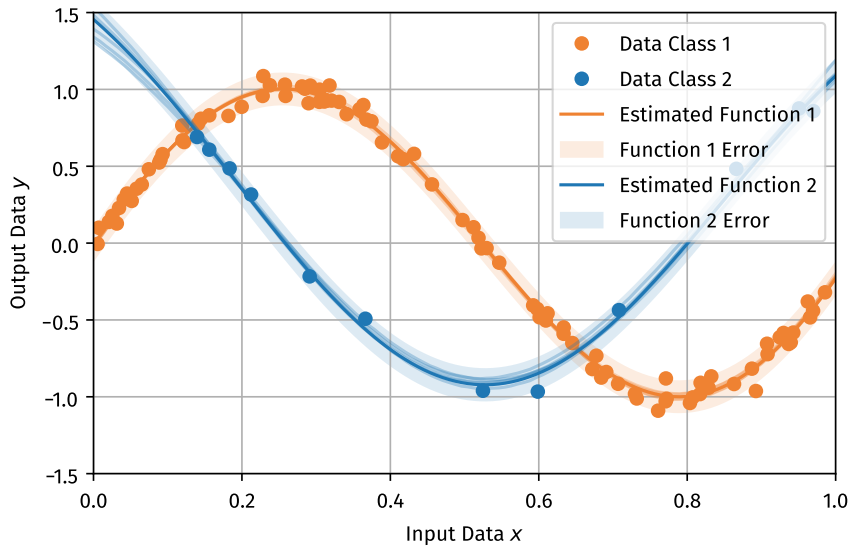
Story of ML: Past...



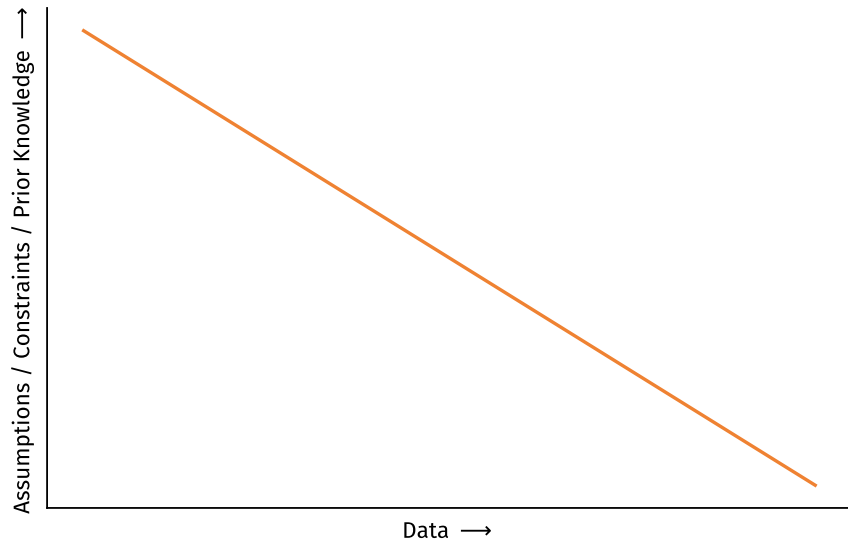
Story of ML: Present...



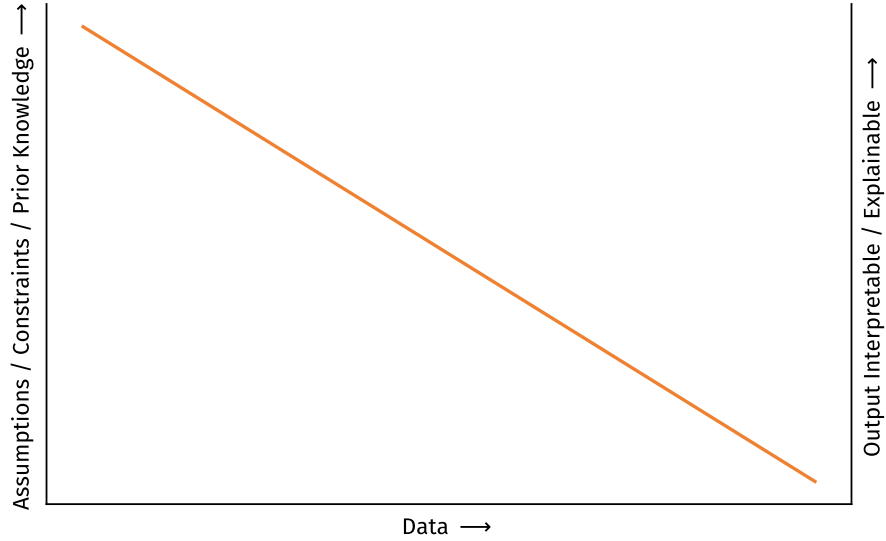
Story of ML: Future...



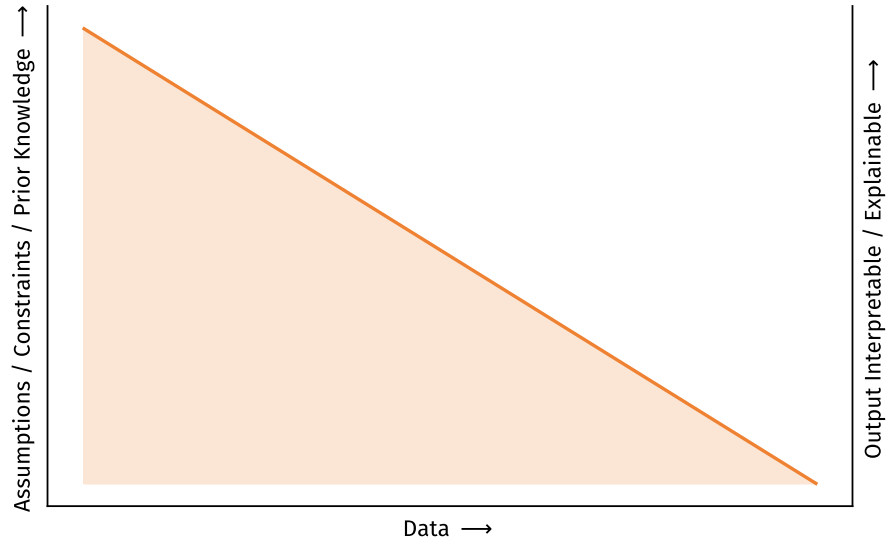
No free lunch



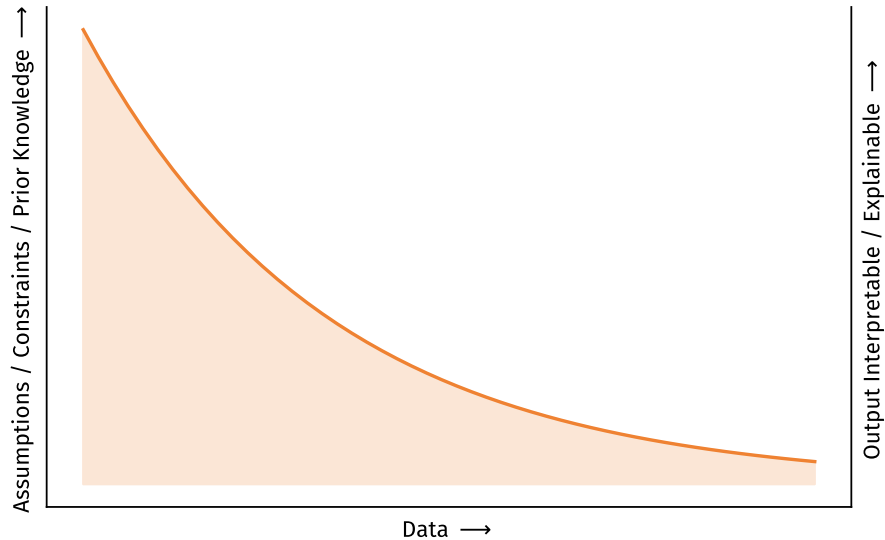
No free lunch



No free lunch



No free lunch (more realistic)



Why are we going to look at Bayesian methods?

“The Theory of probability is simply common sense reduced to calculus”

Pierre-Simon Laplace, 1749-1827

- We use Machine Learning to deal with the **unknown**
- Bayesian probability is the application of logic in the face of uncertainty

Why are we going to look at Bayesian methods?

“The Theory of probability is simply common sense reduced to calculus”
Pierre-Simon Laplace, 1749-1827

- We use Machine Learning to deal with the **unknown**
- Bayesian probability is the application of logic in the face of uncertainty
- Applications usually care about **Inverse Probability**..

Whirlwind Introduction to Inverse Probabilities

Overview...

Motivation

No Free Lunch

Whirlwind Introduction to Inverse Probabilities

Model Selection

Gaussian Processes (a Prior over Functions)

Application: Bayesian Optimisation

Evaluation

Conclusions

Monty Hall..

Monty Hall..



Monty Hall..



Monty Hall..



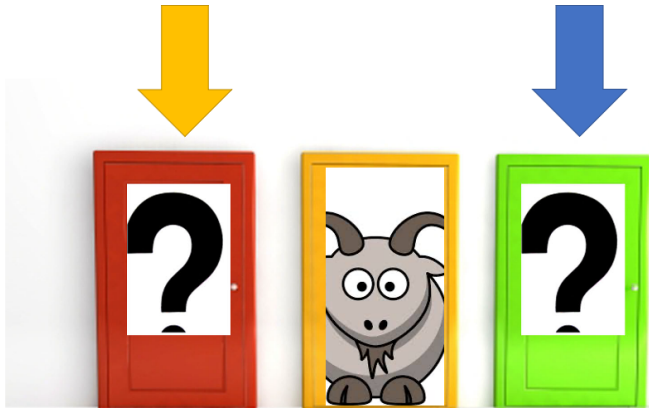
Monty Hall..



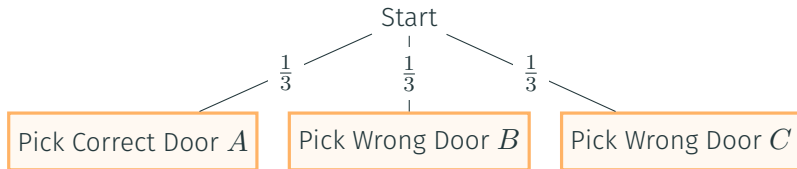
Monty Hall..



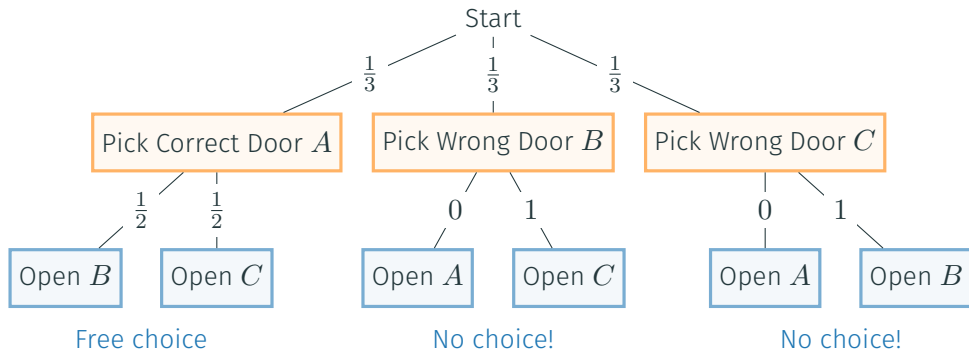
Monty Hall..



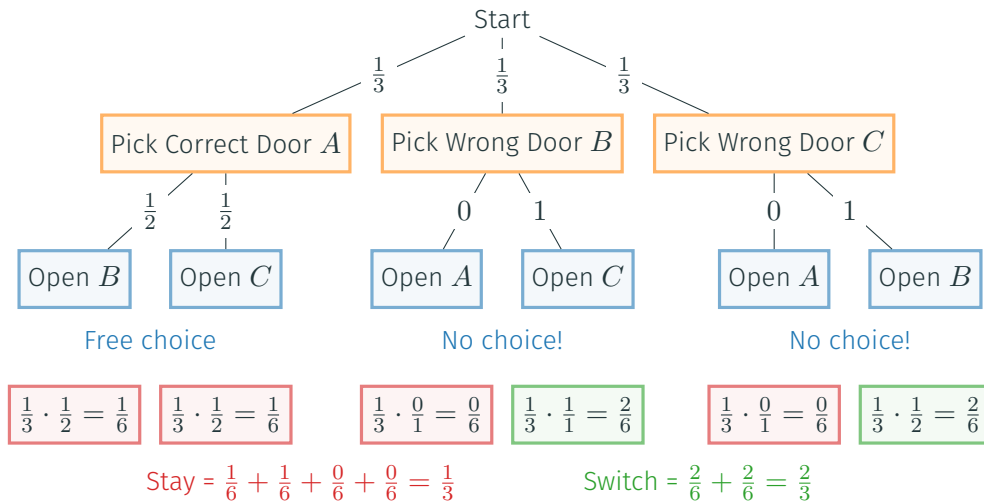
Monty Hall: Think of a tree diagram...



Monty Hall: Think of a tree diagram...



Monty Hall: Think of a tree diagram...



Monty Hall: How would we generate data (or simulate)?

Monty Hall: How would we generate data (or simulate)?

```
1 door_with_car = pick_random({1, 2, 3})
2 door_with_goat = {1, 2, 3} - door_with_car
3
4 door_picked = pick_random({1, 2, 3})
5
6 if door_picked == door_with_car:
7     door_to_open = pick_random(door_with_goat)
8 else:
9     door_to_open = door_with_goat - door_picked
```

Monty Hall: How would we generate data (or simulate)?

```
1 door_with_car = pick_random({1, 2, 3})           # 1/3 equal chance
2 door_with_goat = {1, 2, 3} - door_with_car
3
4 door_picked = pick_random({1, 2, 3})             # 1/3 equal chance
5
6 if door_picked == door_with_car:
7     door_to_open = pick_random(door_with_goat)    # 1 times in 3
8 else:
9     door_to_open = door_with_goat - door_picked  # 2 times in 3
```

Consider **Modelling** as a **Generative
Process**

Probability

- Notation $p(a) \equiv p(A = a)$ where a is a particular outcome chosen for a random variable A from the set of all possible outcomes $\mathbb{A}$

$p(A | B)$ means “probability of A being the case given that B occurs”

- Probabilities in the range $0 \rightarrow 1$
- $0 = \text{impossible}$
- $1 = \text{certain}$
- Sum over all possible outcomes must be 1
- *Important:* We only have $\sum_{a \in \mathbb{A}} p(a | b) = 1$ is only true over $a \in \mathbb{A}$ (not for b)

The Rules of Probability (there are only 2!)

The Sum Rule (Marginalisation)

$$p(A = a) = \sum_{b \in \mathbb{B}} p(A = a, B = b)$$

- If continuous, rather than discrete, use densities and

$$p(A = a) = \int_{\mathbb{R}} p(A = a, B = b) db$$

The Product Rule

$$p(A = a, B = b) = p(A = a \mid B = b) p(B = b) = p(B = b \mid A = a) p(A = a)$$

- Bayes' rule follows from these rules..
- Only consistent approach for probability as “degree of plausibility” (Cox)

Bayes' Rule (it's not really an additional rule!)

- From the product rule

$$p(A = a, B = b) = p(A = a \mid B = b) p(B = b) = p(B = b \mid A = a) p(A = a) \\ \Rightarrow p(A = a \mid B = b) = \frac{p(B = b \mid A = a) p(A = a)}{p(B = b)}$$

- From the sum rule, the denominator is

$$p(B = b) = \sum_{a \in \mathbb{A}} p(A = a, B = b) = \sum_{a \in \mathbb{A}} p(B = b \mid A = a) p(A = a)$$

- So we can confirm that the result must sum to one over $a \in \mathbb{A}$!

$$\Rightarrow p(A = a \mid B = b) = \frac{p(B = b \mid A = a) p(A = a)}{\sum_{a \in \mathbb{A}} p(B = b \mid A = a) p(A = a)}$$

Bayes' Rule

- We can also **condition** on other information $\mathcal{H}$

$$p(a \mid b, \mathcal{H}) = \frac{p(b \mid a, \mathcal{H}) p(a \mid \mathcal{H})}{p(b \mid \mathcal{H})}$$

- We give the parts of equation specific terms

$$\text{Posterior} = \frac{\text{Likelihood} \times \text{Prior}}{\text{Evidence}}$$

$$\text{Posterior Probability (after)} = \frac{\text{Likelihood (of event)} \times \text{Prior Probability (before)}}{\text{Evidence}}$$

Example of Bayes' Rule..

- Consider a legal trial..

$$\underbrace{p(\text{guilt} \mid \text{observations})}_{\text{Posterior}} = \frac{\overbrace{p(\text{observations} \mid \text{guilt})}^{\text{Likelihood}} \times \overbrace{p(\text{guilt})}^{\text{Prior}}}{\underbrace{p(\text{observations})}_{\text{Evidence}}}$$

Example of Bayes' Rule..

- Consider a legal trial..

$$\underbrace{p(\text{guilt} \mid \text{observations})}_{\text{Posterior}} = \frac{\overbrace{p(\text{observations} \mid \text{guilt})}^{\text{Likelihood}} \times \overbrace{p(\text{guilt})}^{\text{Prior}}}{\underbrace{\sum_{\text{guilt}} p(\text{observations} \mid \text{guilt}) p(\text{guilt})}_{\text{Evidence}}}$$

- The evidence is the sum of the top row over both the guilty ($\text{guilt} = 1$) and innocent ($\text{guilt} = 0$) cases.
- We might want to be careful about how we treat $p(\text{guilt})$.

Example of Bayes' Rule..

- Consider a legal trial..

$$\underbrace{p(\text{guilt} \mid \text{observations})}_{\text{Posterior}} = \frac{\overbrace{p(\text{observations} \mid \text{guilt})}^{\text{Likelihood}} \times \overbrace{p(\text{guilt})}^{\text{Prior}}}{\underbrace{\sum_{\text{guilt}} p(\text{observations} \mid \text{guilt}) p(\text{guilt})}_{\text{Evidence}}}$$

- The evidence is the sum of the top row over both the guilty ($\text{guilt} = 1$) and innocent ($\text{guilt} = 0$) cases.
- We might want to be careful about how we treat $p(\text{guilt})$.

Example of Bayes' Rule..

- Consider a legal trial..

$$\underbrace{p(\text{guilt} \mid \text{observations})}_{\text{Posterior}} = \frac{\overbrace{p(\text{observations} \mid \text{guilt})}^{\text{Likelihood}} \times \overbrace{p(\text{guilt})}^{\text{Prior}}}{\underbrace{\sum_{\text{guilt}} p(\text{observations} \mid \text{guilt}) p(\text{guilt})}_{\text{Evidence}}}$$

- The evidence is the sum of the top row over both the guilty ($\text{guilt} = 1$) and innocent ($\text{guilt} = 0$) cases.
- We might want to be careful about how we treat $p(\text{guilt})$.
- Consider that the jury has to return a verdict “*beyond all reasonable doubt*”..

Bayes' Rule with models and functions..

$$\underbrace{p(\text{functions} \mid \text{observed data})}_{\text{Posterior}} = \frac{\overbrace{p(\text{observed data} \mid \text{functions})}^{\text{Likelihood}} \times \overbrace{p(\text{functions})}^{\text{Prior}}}{\underbrace{p(\text{observed data})}_{\text{Evidence}}}$$

Bayes' Rule with models and functions..

$$\underbrace{p(\text{functions} \mid \text{observed data})}_{\text{Posterior}} = \frac{\overbrace{p(\text{observed data} \mid \text{functions})}^{\text{Likelihood}} \times \overbrace{p(\text{functions})}^{\text{Prior}}}{\underbrace{p(\text{observed data})}_{\text{Evidence}}}$$

$$\underbrace{p(f \mid \mathcal{D})}_{\text{Posterior}} = \frac{\overbrace{p(\mathcal{D} \mid f)}^{\text{Likelihood}} \times \overbrace{p(f)}^{\text{Prior}}}{\underbrace{p(\mathcal{D})}_{\text{Evidence}}} \quad p(\mathcal{D}) = \sum_f p(\mathcal{D} \mid f) p(f)$$

Data $\mathcal{D} = \{X, Y\}$, pairs of inputs $\{x_n\}$ and outputs $\{y_n\}$, and functions f

Bayes' Rule with models and functions..

$$\underbrace{p(\text{functions} \mid \text{observed data})}_{\text{Posterior}} = \frac{\overbrace{p(\text{observed data} \mid \text{functions})}^{\text{Likelihood}} \times \overbrace{p(\text{functions})}^{\text{Prior}}}{\underbrace{p(\text{observed data})}_{\text{Evidence}}}$$

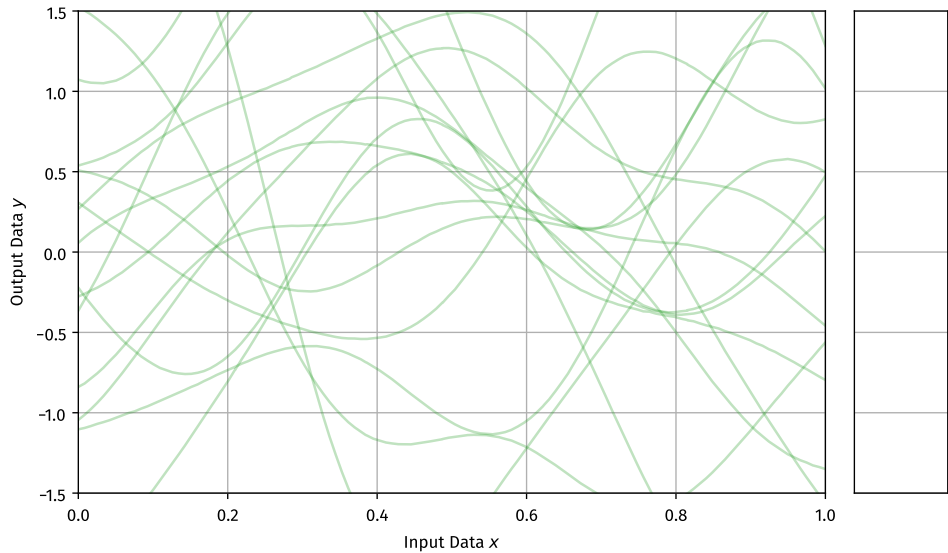
$$\underbrace{p(f \mid \mathcal{D})}_{\text{Posterior}} = \frac{\overbrace{p(\mathcal{D} \mid f)}^{\text{Likelihood}} \times \overbrace{p(f)}^{\text{Prior}}}{\underbrace{p(\mathcal{D})}_{\text{Evidence}}} \quad p(\mathcal{D}) = \sum_f p(\mathcal{D} \mid f) p(f)$$

Data $\mathcal{D} = \{X, Y\}$, pairs of inputs $\{x_n\}$ and outputs $\{y_n\}$, and functions f

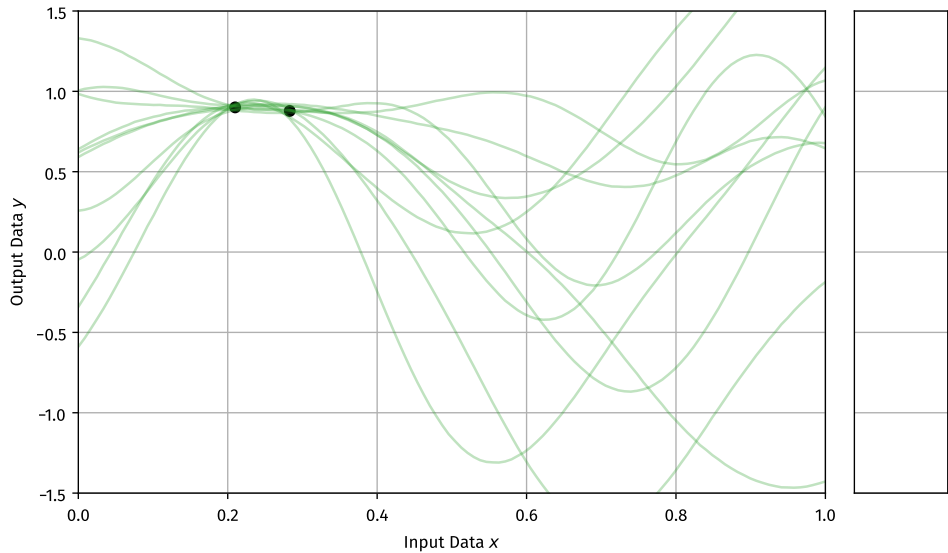
Average over functions to predict unknown output y^* for a new input x^* :

$$p(y^* \mid x^*, \mathcal{D}) = \sum_f p(y^* \mid x^*, f) p(f \mid \mathcal{D})$$

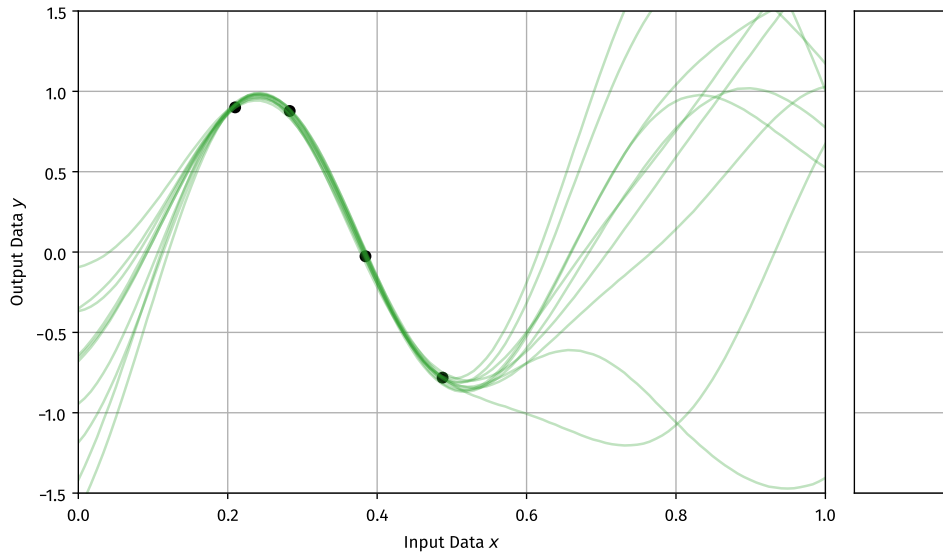
Prior over functions...



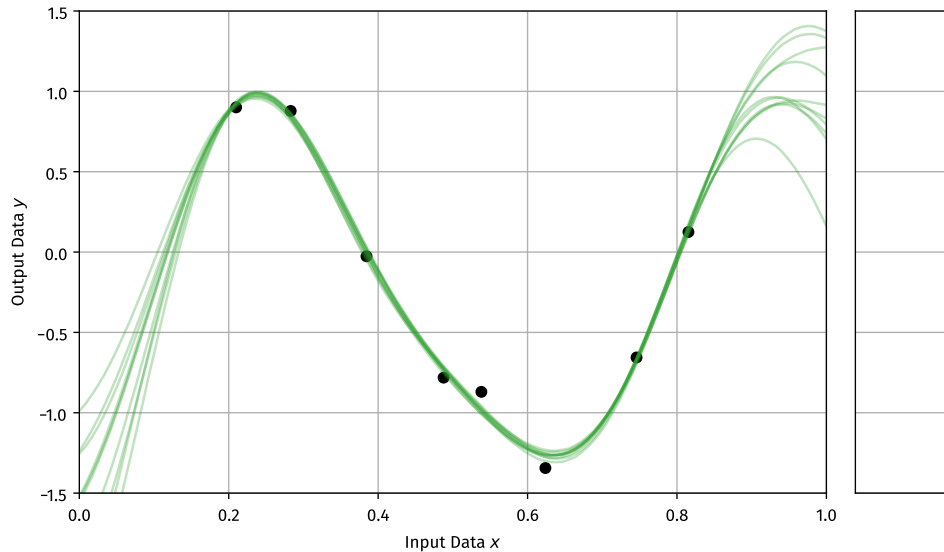
Combine prior with data...



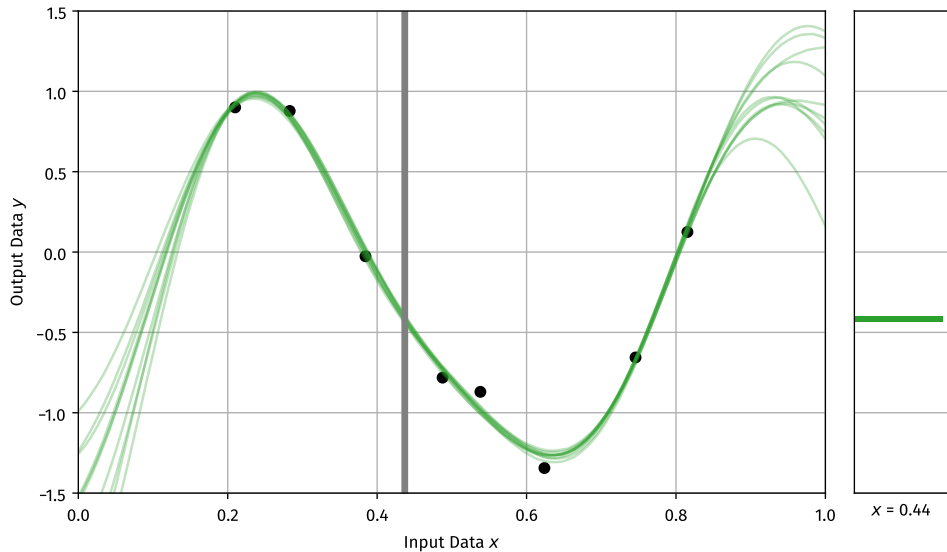
Combine prior with data...



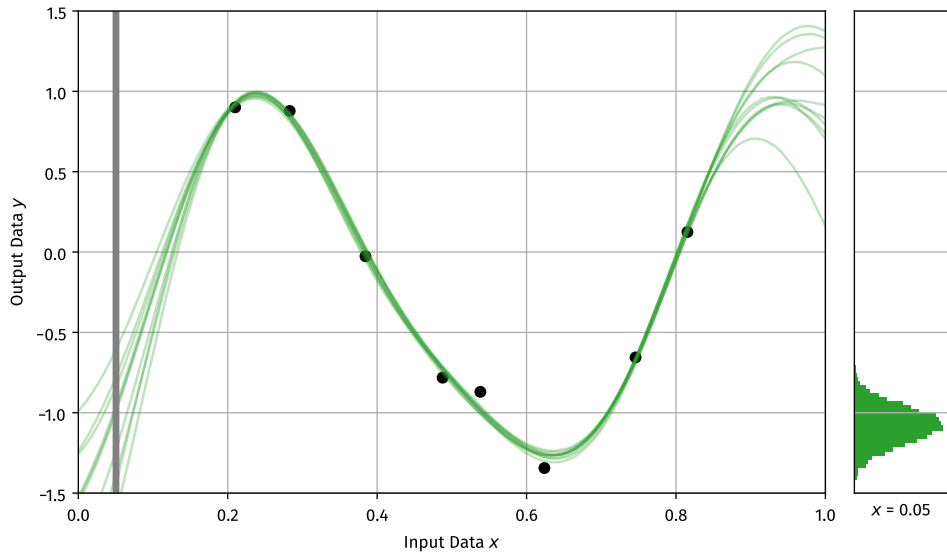
Combine prior with data...



Average over functions to predict...



Averaging over functions gives us (Epistemic) Uncertainty!



Bayes' Rule with models and parameters..

$$\underbrace{p(\text{params} \mid \text{obs. data})}_{\text{Posterior}} = \frac{\overbrace{p(\text{obs. data} \mid \text{params})}^{\text{Likelihood}} \times \overbrace{p(\text{params})}^{\text{Prior}}}{\underbrace{p(\text{obs. data})}_{\text{Evidence}}}$$

Bayes' Rule with models and parameters..

$$\underbrace{p(\text{params} \mid \text{obs. data})}_{\text{Posterior}} = \frac{\overbrace{p(\text{obs. data} \mid \text{params})}^{\text{Likelihood}} \times \overbrace{p(\text{params})}^{\text{Prior}}}{\underbrace{p(\text{obs. data})}_{\text{Evidence}}}$$

$$\underbrace{p(w \mid \mathcal{D})}_{\text{Posterior}} = \frac{\overbrace{p(\mathcal{D} \mid w)}^{\text{Likelihood}} \times \overbrace{p(w)}^{\text{Prior}}}{\underbrace{p(\mathcal{D})}_{\text{Evidence}}}$$

Data $\mathcal{D} = X, Y$, pairs of inputs x_n and outputs y_n

Bayes' Rule with models and parameters..

$$\underbrace{p(\text{params} \mid \text{obs. data})}_{\text{Posterior}} = \frac{\overbrace{p(\text{obs. data} \mid \text{params})}^{\text{Likelihood}} \times \overbrace{p(\text{params})}^{\text{Prior}}}{\underbrace{p(\text{obs. data})}_{\text{Evidence}}}$$

$$\underbrace{p(w \mid \mathcal{D})}_{\text{Posterior}} = \frac{\overbrace{p(\mathcal{D} \mid w)}^{\text{Likelihood}} \times \overbrace{p(w)}^{\text{Prior}}}{\underbrace{p(\mathcal{D})}_{\text{Evidence}}} \quad p(\mathcal{D}) = \sum_w p(\mathcal{D} \mid w) p(w)$$

Data $\mathcal{D} = X, Y$, pairs of inputs x_n and outputs y_n

Prediction of output y^* for a new input x^* :

$$p(y^* \mid x^*, \mathcal{D}) = \sum_w p(y^* \mid x^*, w) p(w \mid \mathcal{D})$$

New thought process; no longer

“Find the best parameters”,

now

“Find all the parameters that
agree with the data”..

Model Selection

Overview...

Motivation

No Free Lunch

Whirlwind Introduction to Inverse Probabilities

Model Selection

Gaussian Processes (a Prior over Functions)

Application: Bayesian Optimisation

Evaluation

Conclusions

Model selection (“Which model should I use?”)

- How much data do we need?

Model selection (“Which model should I use?”)

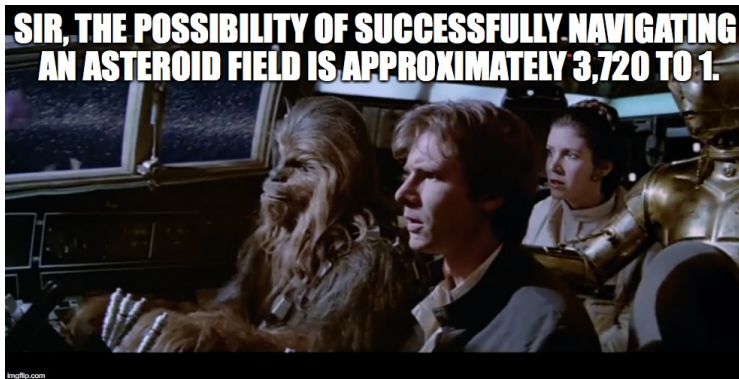
- How much data do we need?
- Might not be the right question..

Model selection (“Which model should I use?”)

- How much data do we need?
- Might not be the right question..
- What can we actually say?

Model selection (“Which model should I use?”)

- How much data do we need?
- Might not be the right question..
- What can we actually say? **The odds!**



Model selection (“Which model should I use?”)

- How much data do we need?
- Might not be the right question..
- What can we actually say? **The odds!**



Science (and Machine Learning) cannot
prove things to be true via data

Science (and Machine Learning) **cannot
prove things to be true via data**
we can only demonstrate that things
are **inconsistent with data**

Model selection illustration: Gravity!



Stable Diffusion: *“Drop cannonball and orange off the leaning tower of Pisa.”*

Model selection illustration: Gravity!

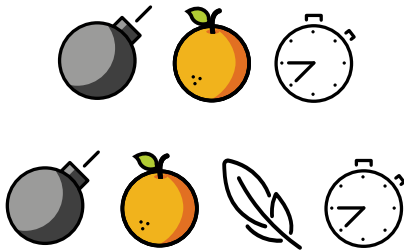


Stable Diffusion: *“Drop cannonball and orange off the leaning tower of Pisa.”*

Model selection illustration: Gravity!



Stable Diffusion: *“Drop cannonball and orange off the leaning tower of Pisa.”*



Model selection illustration: Gravity!



Stable Diffusion: *“Drop cannonball and orange off the leaning tower of Pisa.”*



Model selection illustration: Gravity!



Stable Diffusion: *“Drop cannonball and orange off the leaning tower of Pisa.”*



Apollo 15 Hammer-Feather Drop



NASASolarSystem
14.9K subscribers

Subscribe

4.8K



Share



576K views · 8 years ago

At the end of the last Apollo 15 moon walk, Commander David Scott (pictured above) performed a live demonstration for the television cameras. He held out a geologic hammer and a feather and dropped them at the same time. Because they were essentially in a vacuum, there w ...more

<https://www.youtube.com/watch?v=oYEgdZ3iEKA>

Bayes' Rule for model selection..

$$\underbrace{p(w \mid \mathcal{D})}_{\text{Posterior}} = \frac{\overbrace{p(\mathcal{D} \mid w)}^{\text{Likelihood}} \times \overbrace{p(w)}^{\text{Prior}}}{\underbrace{p(\mathcal{D})}_{\text{Evidence}}}$$

Data $\mathcal{D} = \{X, Y\}$, input/output pairs, and parameters w

Bayes' Rule for model selection..

$$\underbrace{p(w \mid \mathcal{D}, \mathcal{M} = m)}_{\text{Posterior under model}} = \frac{\overbrace{p(\mathcal{D} \mid w, \mathcal{M} = m)}^{\text{Likelihood under model}} \times \overbrace{p(w, \mathcal{M} = m)}^{\text{Prior}}}{\underbrace{p(\mathcal{D} \mid \mathcal{M} = m)}_{\text{Evidence for model}}}$$

Data $\mathcal{D} = \{X, Y\}$, input/output pairs, and parameters w for Model $\mathcal{M} = m$

Bayes' Rule for model selection..

$$\underbrace{p(w \mid \mathcal{D}, \mathcal{M} = m)}_{\text{Posterior under model}} = \frac{\overbrace{p(\mathcal{D} \mid w, \mathcal{M} = m)}^{\text{Likelihood under model}} \times \overbrace{p(w, \mathcal{M} = m)}^{\text{Prior}}}{\underbrace{p(\mathcal{D} \mid \mathcal{M} = m)}_{\text{Evidence for model}}}$$

Data $\mathcal{D} = \{X, Y\}$, input/output pairs, and parameters w for Model $\mathcal{M} = m$

$$\underbrace{p(\mathcal{M} = m \mid \mathcal{D})}_{\text{Posterior for model}} = \frac{\overbrace{p(\mathcal{D} \mid \mathcal{M} = m)}^{\text{Evidence for model}} \times \overbrace{p(\mathcal{M} = m)}^{\text{Prior for model}}}{\underbrace{p(\mathcal{D})}_{\text{Data}}}$$

Bayes' Rule for model selection..

$$\underbrace{p(w \mid \mathcal{D}, \mathcal{M} = m)}_{\text{Posterior under model}} = \frac{\overbrace{p(\mathcal{D} \mid w, \mathcal{M} = m)}^{\text{Likelihood under model}} \times \overbrace{p(w, \mathcal{M} = m)}^{\text{Prior}}}{\underbrace{p(\mathcal{D} \mid \mathcal{M} = m)}_{\text{Evidence for model}}}$$

Data $\mathcal{D} = \{X, Y\}$, input/output pairs, and parameters w for Model $\mathcal{M} = m$

$$\underbrace{p(\mathcal{M} = m \mid \mathcal{D})}_{\text{Posterior for model}} = \frac{\overbrace{p(\mathcal{D} \mid \mathcal{M} = m)}^{\text{Evidence for model}} \times \overbrace{p(\mathcal{M} = m)}^{\text{Prior for model}}}{\underbrace{p(\mathcal{D})}_{\text{Data}}}$$

If prior over models is equal, we compare via the **Evidence for the Model**: $p(\mathcal{D} \mid \mathcal{M} = m)$

Model selection example

Fitting polynomial models to data under Gaussian noise, $\varepsilon_n \sim \mathcal{N}(0, \sigma^2)$:

Model 1 : $y_n = a_0 + a_1x_n + \varepsilon_n$

Model 2 : $y_n = a_0 + a_1x_n + a_2x^2 + \varepsilon_n$

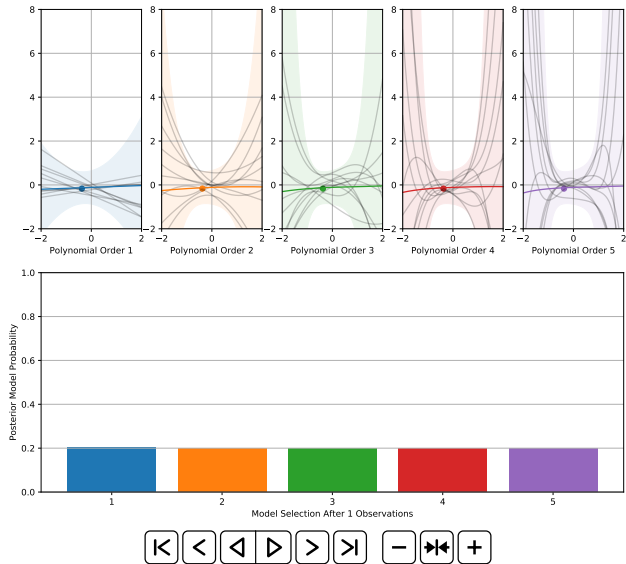
Model 3 : $y_n = a_0 + a_1x_n + a_2x^2 + a_3x^3 + \varepsilon_n$

Model 4 : $y_n = a_0 + a_1x_n + a_2x^2 + a_3x^3 + a_4x^4 + \varepsilon_n$

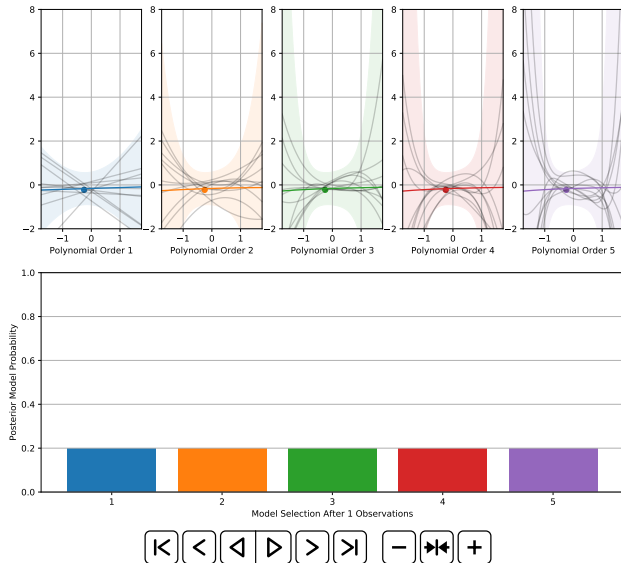
Model 5 : $y_n = a_0 + a_1x_n + a_2x^2 + a_3x^3 + a_4x^4 + a_5x^5 + \varepsilon_n$

Parameters $w_m = [a_0, \dots, a_m]$ for model m , where $m \in [1, \dots, 5]$.

Model selection example

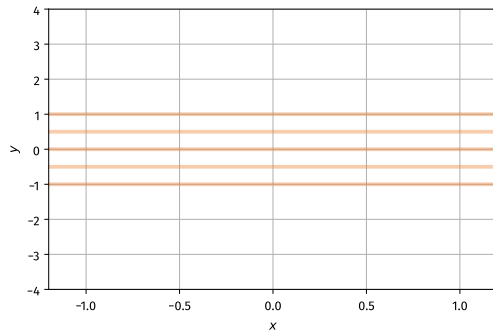
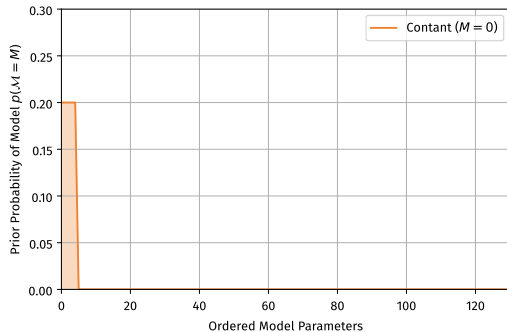


Model selection example (more noise)



Model Selection: Polynomial Example

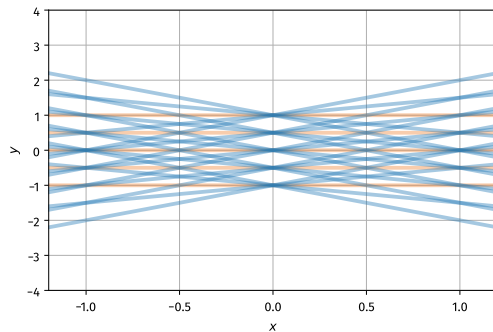
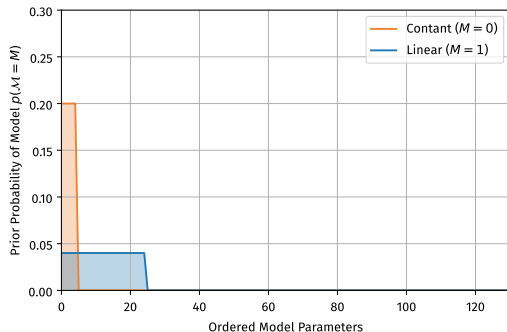
Simple (discrete) case: $y(x) = \sum_{m=0}^M a_m x^m$, $a_m \in \{-1, -1/2, 0, 1/2, 1\}$



$$\begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \begin{bmatrix} -1 \\ 0 \\ 0 \end{bmatrix} \begin{bmatrix} -1/2 \\ 0 \\ 0 \end{bmatrix} \begin{bmatrix} 1/2 \\ 0 \\ 0 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$$
$$w_0 = [a_0, 0, 0], N = 5$$

Model Selection: Polynomial Example

Simple (discrete) case: $y(x) = \sum_{m=0}^M a_m x^m$, $a_m \in \{-1, -1/2, 0, 1/2, 1\}$

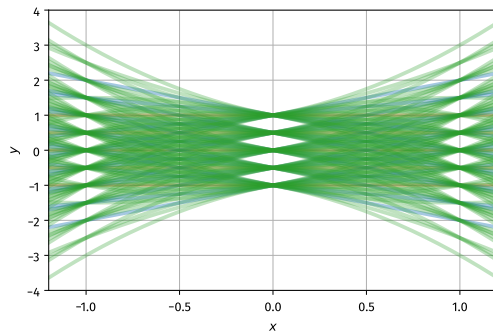
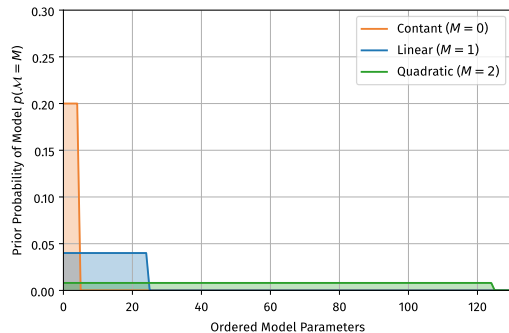


$$\begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \begin{bmatrix} -1 \\ 0 \\ 0 \end{bmatrix} \begin{bmatrix} -1/2 \\ 0 \\ 0 \end{bmatrix} \begin{bmatrix} 1/2 \\ 0 \\ 0 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$$
$$w_0 = [a_0, 0, 0], N = 5$$

$$\begin{bmatrix} -1 \\ -1 \\ 0 \end{bmatrix} \begin{bmatrix} -1 \\ -1/2 \\ 0 \end{bmatrix} \begin{bmatrix} -1 \\ 1/2 \\ 0 \end{bmatrix} \dots \begin{bmatrix} 1 \\ 1/2 \\ 0 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix}$$
$$w_1 = [a_0, a_1, 0], N = 25$$

Model Selection: Polynomial Example

Simple (discrete) case: $y(x) = \sum_{m=0}^M a_m x^m$, $a_m \in \{-1, -1/2, 0, 1/2, 1\}$



$$\begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \begin{bmatrix} -1 \\ 0 \\ 0 \end{bmatrix} \begin{bmatrix} -1/2 \\ 0 \\ 0 \end{bmatrix} \begin{bmatrix} 1/2 \\ 0 \\ 0 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$$

$$w_0 = [a_0, 0, 0], N = 5$$

$$\begin{bmatrix} -1 \\ -1 \\ 0 \end{bmatrix} \begin{bmatrix} -1 \\ -1/2 \\ 0 \end{bmatrix} \begin{bmatrix} -1 \\ 1/2 \\ 0 \end{bmatrix} \dots \begin{bmatrix} 1 \\ 1/2 \\ 0 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix}$$

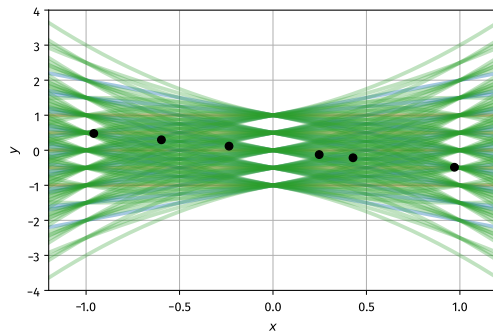
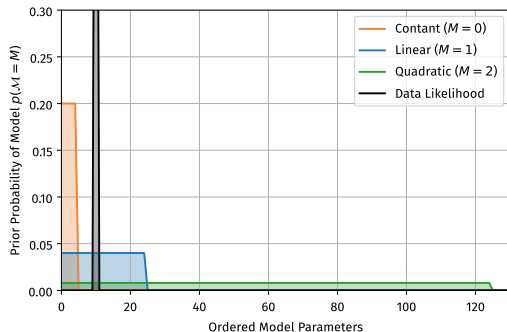
$$w_1 = [a_0, a_1, 0], N = 25$$

$$\begin{bmatrix} -1 \\ -1 \\ -1 \end{bmatrix} \begin{bmatrix} -1 \\ -1 \\ -1/2 \end{bmatrix} \begin{bmatrix} -1 \\ -1 \\ 1/2 \end{bmatrix} \dots \begin{bmatrix} 1 \\ 1 \\ 1/2 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

$$w_2 = [a_0, a_1, a_2], N = 125$$

Model Selection: Polynomial Example

Simple (discrete) case: $y(x) = \sum_{m=0}^M a_m x^m$, $a_m \in \{-1, -1/2, 0, 1/2, 1\}$



$$\begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \begin{bmatrix} -1 \\ 0 \\ 0 \end{bmatrix} \begin{bmatrix} -1/2 \\ 0 \\ 0 \end{bmatrix} \begin{bmatrix} 1/2 \\ 0 \\ 0 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$$

$$w_0 = [a_0, 0, 0], N = 5$$

$$\begin{bmatrix} -1 \\ -1 \\ 0 \end{bmatrix} \begin{bmatrix} -1 \\ -1/2 \\ 0 \end{bmatrix} \begin{bmatrix} -1 \\ 1/2 \\ 0 \end{bmatrix} \dots \begin{bmatrix} 1 \\ 1/2 \\ 0 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix}$$

$$w_1 = [a_0, a_1, 0], N = 25$$

$$\begin{bmatrix} -1 \\ -1 \\ -1 \end{bmatrix} \begin{bmatrix} -1 \\ -1 \\ -1/2 \end{bmatrix} \begin{bmatrix} -1 \\ -1 \\ 1/2 \end{bmatrix} \dots \begin{bmatrix} 1 \\ 1 \\ 1/2 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

$$w_2 = [a_0, a_1, a_2], N = 125$$

Gaussian Processes (a Prior over Functions)

Overview...

Motivation

No Free Lunch

Whirlwind Introduction to Inverse Probabilities

Model Selection

Gaussian Processes (a Prior over Functions)

Application: Bayesian Optimisation

Evaluation

Conclusions

Bayesian Regression Example

Consider a regression model with data $\{x_n, y_n\}$ and noise $\{\varepsilon_n\}$, $n = 1, \dots, N$:

$$y_n = f(x_n; \mathbf{w}) + \varepsilon_n .$$

We define the model with parameters $\mathbf{w}$ as

$$f(x; \mathbf{w}) = \sum_{m=1}^M w_m \phi_m(x) ;$$

a weighted linear sum of M *non-linear* bases $\phi_m(x)$ with

$$\phi_m(x) = \exp \left(-\frac{(x - c_m)^2}{2\ell^2} \right) .$$

This is often termed a **Radial Basis Function** (or Exponentiated Quadratic Function) with fixed centres c_m and lengthscale ℓ .

Bayesian Regression Example

If we assume an iid Gaussian noise model for $\varepsilon_n \sim \mathcal{N}(0, \sigma^2)$ then we have the likelihood

$$p(\mathbf{y} | \mathbf{x}, \mathbf{w}, \sigma^2) = \prod_{n=1}^N p(y_n | x_n, \mathbf{w}, \sigma^2),$$

with

$$p(y_n | x_n, \mathbf{w}, \sigma^2) = \mathcal{N}(f(x_n; \mathbf{w}), \sigma^2).$$

We put a zero mean Gaussian prior on the weights

$$p(\mathbf{w} | \alpha) = \prod_{m=1}^M \mathcal{N}(w_m | 0, \alpha^{-1})$$

with precision (inverse variance) α .

Bayesian Regression Example

Since Gaussians are *self-conjugate*, the posterior over the weights is also Gaussian:

$$p(\mathbf{w} | \mathbf{x}, \mathbf{y}, \alpha, \sigma^2) = \frac{p(\mathbf{y} | \mathbf{x}, \mathbf{w}, \sigma^2) p(\mathbf{w} | \alpha)}{p(\mathbf{y} | \mathbf{x}, \alpha, \sigma^2)} = \mathcal{N}(\mathbf{w} | \boldsymbol{\mu}, \boldsymbol{\Sigma}),$$

where

$$\boldsymbol{\mu} := [\boldsymbol{\Phi}^\top \boldsymbol{\Phi} + \sigma^2 \alpha \mathbf{I}]^{-1} \boldsymbol{\Phi}^\top \mathbf{x}$$

$$\boldsymbol{\Sigma} := \sigma^2 [\boldsymbol{\Phi}^\top \boldsymbol{\Phi} + \sigma^2 \alpha \mathbf{I}]^{-1}$$

$$[\boldsymbol{\Phi}]_{n,m} := \phi_m(x_n).$$

Therefore, the **predictive distribution** over y^* for a new input x^* is

$$\begin{aligned} p(y^* | x^*, \mathbf{x}, \mathbf{y}, \alpha, \sigma^2) &= \int p(y^* | x^*, \mathbf{w}) p(\mathbf{w} | \mathbf{x}, \mathbf{y}, \alpha, \sigma^2) d\mathbf{w} \\ &= \mathcal{N}(y^* | \boldsymbol{\phi}^{*\top} A^{-1} \boldsymbol{\Phi}^\top \mathbf{y}, \boldsymbol{\phi}^{*\top} A^{-1} \boldsymbol{\phi}^* + \sigma^2), \end{aligned}$$

with

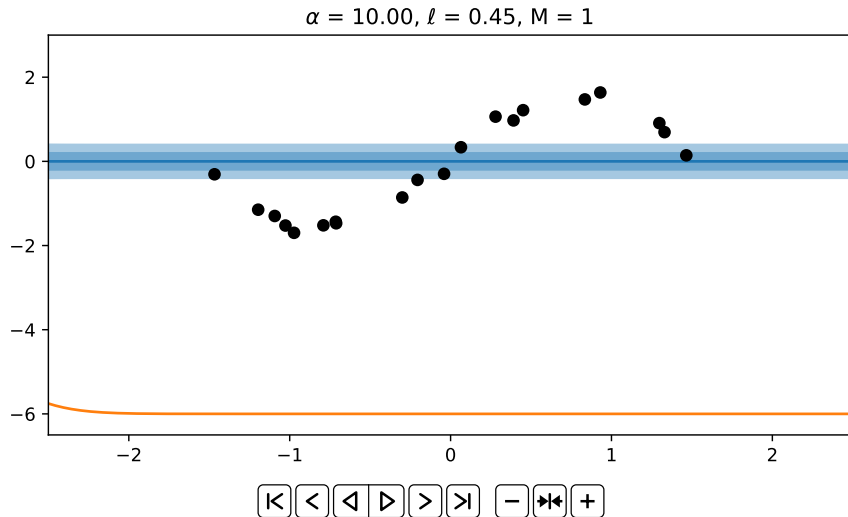
$$A := [\boldsymbol{\Phi}^\top \boldsymbol{\Phi} + \sigma^2 \alpha \mathbf{I}], [\boldsymbol{\phi}^*]_m := \phi_m(x^*).$$

This leaves us with a number of choices to make:

1. How many basis functions (M) should we use?
2. Where should they be located?
3. What about the lengthscale ℓ ?
4. How does the prior α affect things?

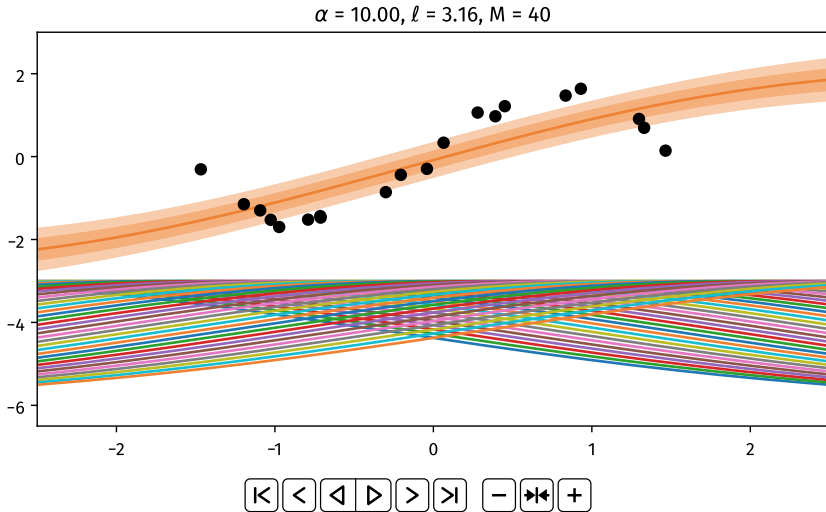
Let's look at these in more detail..

Bayesian Regression: Varying the Number of Bases (M)



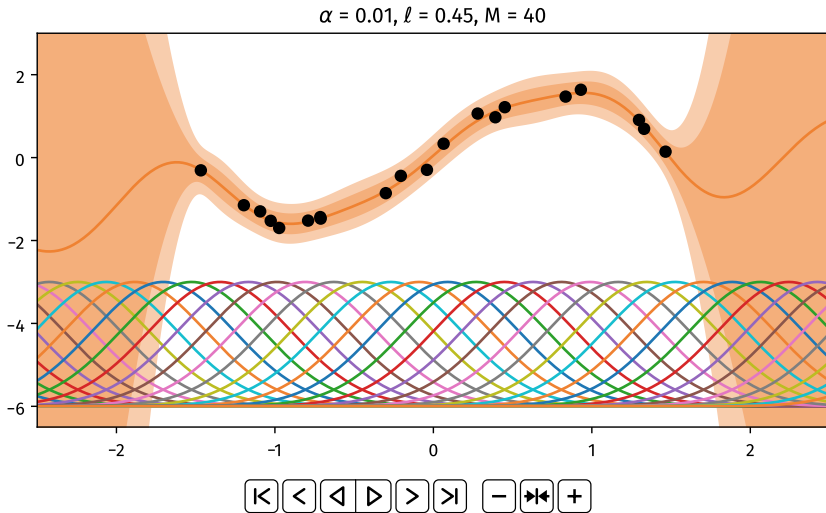
We can show that in the **infinite limit** ($M \rightarrow \infty$) we get a **Gaussian Process**!

Bayesian Regression: Varying the Lengthscale (ℓ)



We observe a transition from "under-fitting" to "over-fitting".

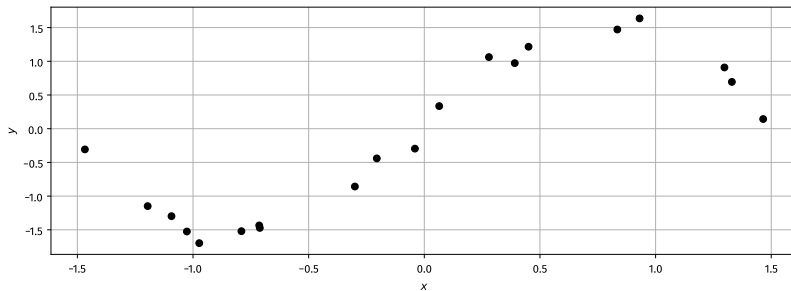
Bayesian Regression: Varying the Prior (α)



Varying the prior affects our "confidence" in the data.

Gaussian Process Regression

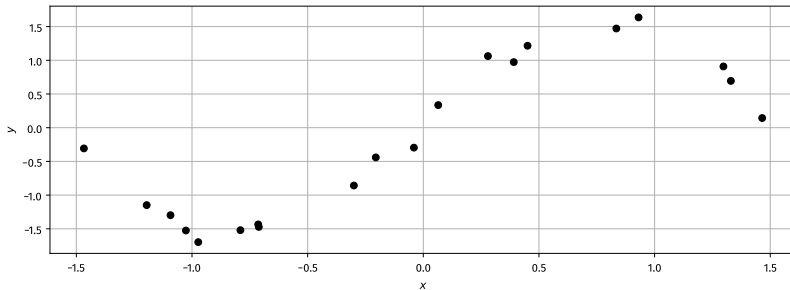
Let us take a step back and look at regression from the beginning...



We want to “fit” $y = f(x) + \varepsilon$, given our examples $\{x_n, y_n\}$; what else do we need?

Gaussian Process Regression

Let us take a step back and look at regression from the beginning...



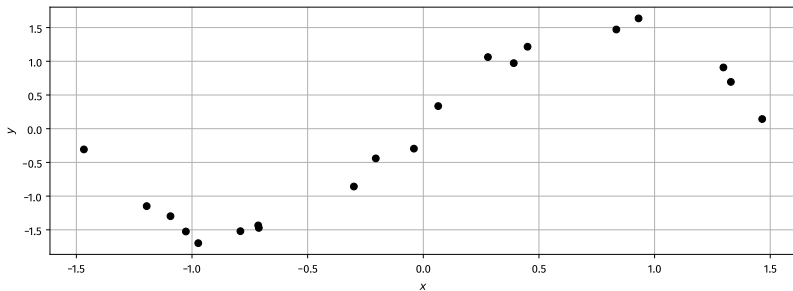
We want to “fit” $y = f(x) + \varepsilon$, given our examples $\{x_n, y_n\}$; what else do we need?

We are performing **learning**, therefore we need to make some **assumptions**...

- What are the minimal assumptions required?

Gaussian Process Regression

Let us take a step back and look at regression from the beginning...



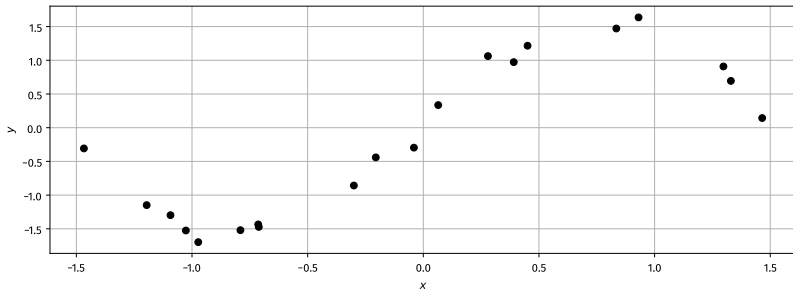
We want to “fit” $y = f(x) + \varepsilon$, given our examples $\{x_n, y_n\}$; what else do we need?

We are performing **learning**, therefore we need to make some **assumptions**...

- What are the minimal assumptions required? **Smoothness!**

Gaussian Process Regression

Let us take a step back and look at regression from the beginning...

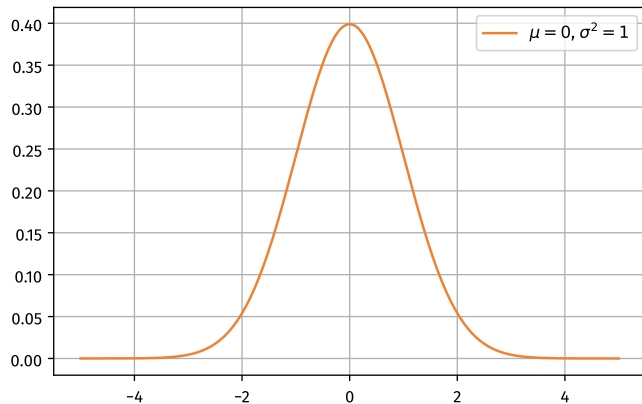


We want to “fit” $y = f(x) + \varepsilon$, given our examples $\{x_n, y_n\}$; what else do we need?

We are performing **learning**, therefore we need to make some **assumptions**...

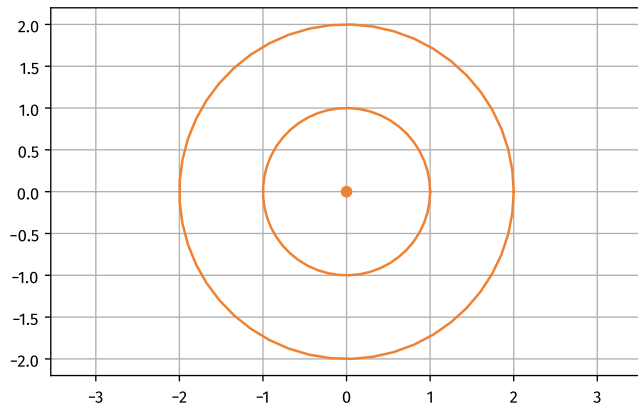
- What are the minimal assumptions required? **Smoothness!**
- We also want error bars! **To quantify our uncertainty.**

The Univariate Gaussian Distribution



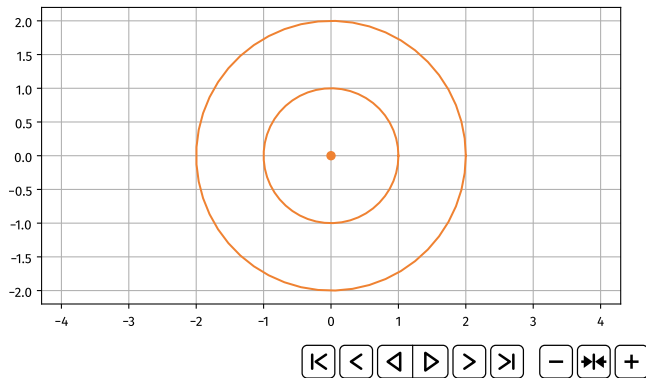
$$y = \mathcal{N}(x | \mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp \left[-\frac{(x-\mu)^2}{2\sigma^2} \right]$$

The Multivariate Gaussian Distribution



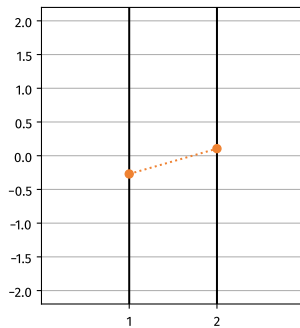
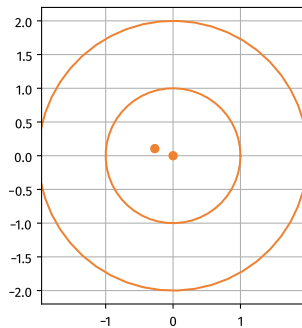
$$\mathbf{y} = \mathcal{N}(\mathbf{x} | \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{\sqrt{2\pi|\boldsymbol{\Sigma}|}} \exp\left[-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\mathbf{x} - \boldsymbol{\mu})\right], \boldsymbol{\mu} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \boldsymbol{\Sigma} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

Covariance of the Multivariate Gaussian



$$\Sigma = \begin{bmatrix} 1.00 & 0.00 \\ 0.00 & 1.00 \end{bmatrix}$$

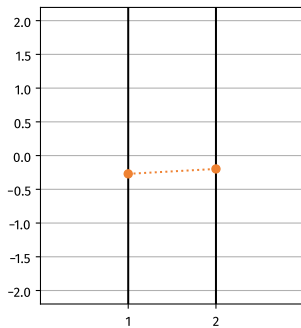
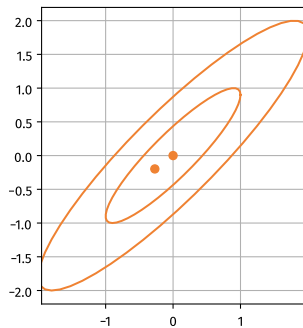
Sampling from a 2D Gaussian



$$\Sigma = \begin{bmatrix} 1.00 & 0.00 \\ 0.00 & 1.00 \end{bmatrix}$$



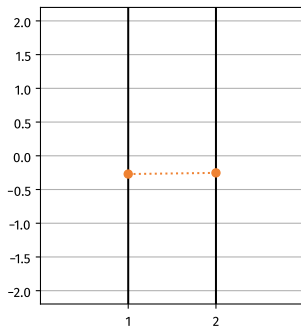
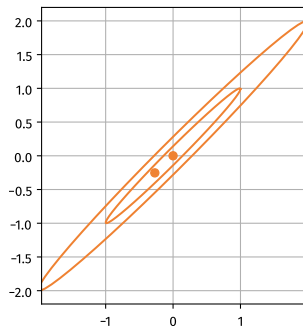
Sampling from a 2D Gaussian



$$\Sigma = \begin{bmatrix} 1.00 & 0.90 \\ 0.90 & 1.00 \end{bmatrix}$$



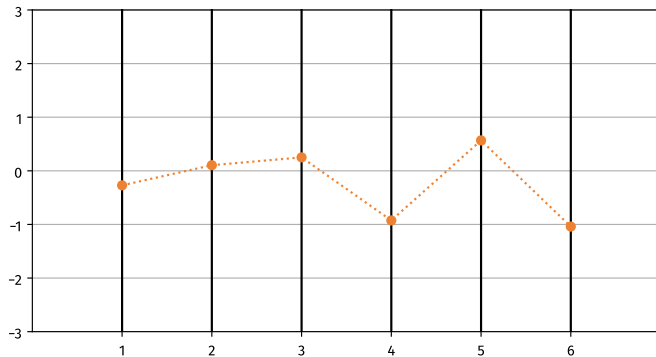
Sampling from a 2D Gaussian



$$\Sigma = \begin{bmatrix} 1.00 & 0.99 \\ 0.99 & 1.00 \end{bmatrix}$$



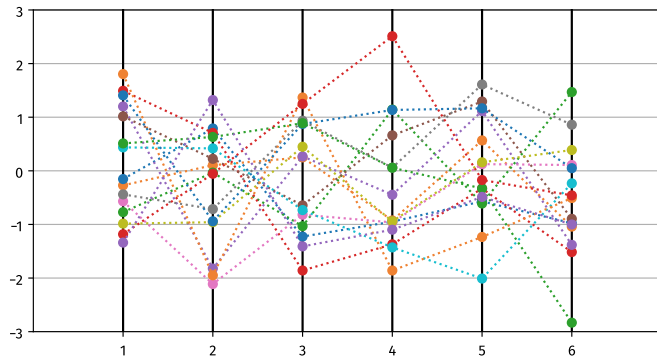
Sampling from a 6D Gaussian



$$\Sigma = \begin{bmatrix} 1.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\ 0.00 & 1.00 & 0.00 & 0.00 & 0.00 & 0.00 \\ 0.00 & 0.00 & 1.00 & 0.00 & 0.00 & 0.00 \\ 0.00 & 0.00 & 0.00 & 1.00 & 0.00 & 0.00 \\ 0.00 & 0.00 & 0.00 & 0.00 & 1.00 & 0.00 \\ 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 1.00 \end{bmatrix}$$



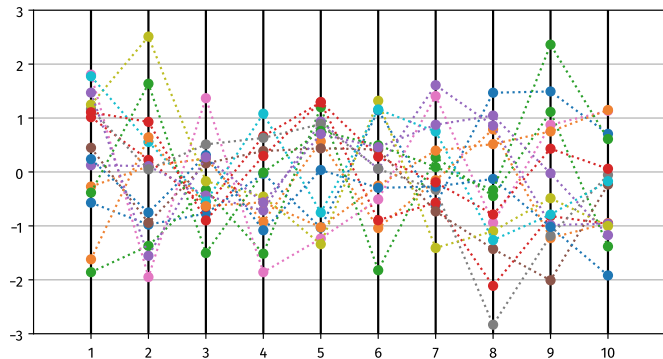
Sampling from a 6D Gaussian (varying covariance)



$$\Sigma = \begin{bmatrix} 1.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\ 0.00 & 1.00 & 0.00 & 0.00 & 0.00 & 0.00 \\ 0.00 & 0.00 & 1.00 & 0.00 & 0.00 & 0.00 \\ 0.00 & 0.00 & 0.00 & 1.00 & 0.00 & 0.00 \\ 0.00 & 0.00 & 0.00 & 0.00 & 1.00 & 0.00 \\ 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 1.00 \end{bmatrix}$$



Sampling from a 10D Gaussian (varying covariance)



$$\Sigma = \begin{bmatrix} 1.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\ 0.00 & 1.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\ 0.00 & 0.00 & 1.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\ 0.00 & 0.00 & 0.00 & 1.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\ 0.00 & 0.00 & 0.00 & 0.00 & 1.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\ 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 1.00 & 0.00 & 0.00 & 0.00 & 0.00 \\ 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 1.00 & 0.00 & 0.00 & 0.00 \\ 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 1.00 & 0.00 & 0.00 \\ 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 1.00 & 0.00 \\ 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 1.00 \end{bmatrix}$$



Observations

- As we change the covariance structure we change the nature of the samples

Observations

- As we change the covariance structure we change the nature of the samples
- Increasing the covariance → smoother samples

Observations

- As we change the covariance structure we change the nature of the samples
- Increasing the covariance → smoother samples
- BUT: how did we know where to place the x -axis values?

Observations

- As we change the covariance structure we change the nature of the samples
- Increasing the covariance → smoother samples
- BUT: how did we know where to place the x -axis values?
- What is the relationship between the different dimensions in the covariance Σ ?

Observations

- As we change the covariance structure we change the nature of the samples
- Increasing the covariance → smoother samples
- BUT: how did we know where to place the x -axis values?
- What is the relationship between the different dimensions in the covariance Σ ?

Observations

- As we change the covariance structure we change the nature of the samples
- Increasing the covariance → smoother samples
- BUT: how did we know where to place the x -axis values?
- What is the relationship between the different dimensions in the covariance Σ ?

$$\Sigma = \begin{bmatrix} \sigma_{11} & \sigma_{12} \\ \sigma_{21} & \sigma_{22} \end{bmatrix}$$

Observations

- As we change the covariance structure we change the nature of the samples
- Increasing the covariance → smoother samples
- BUT: how did we know where to place the x -axis values?
- What is the relationship between the different dimensions in the covariance Σ ?

$$\bullet \sigma_{12} = \sigma_{21}$$

$$\Sigma = \begin{bmatrix} \sigma_{11} & \sigma_{12} \\ \sigma_{21} & \sigma_{22} \end{bmatrix}$$

Observations

- As we change the covariance structure we change the nature of the samples
- Increasing the covariance → smoother samples
- BUT: how did we know where to place the x -axis values?
- What is the relationship between the different dimensions in the covariance Σ ?

$$\Sigma = \begin{bmatrix} \sigma_{11} & \sigma_{12} \\ \sigma_{21} & \sigma_{22} \end{bmatrix}$$

- $\sigma_{12} = \sigma_{21}$
- High values of $\sigma_{12} \Rightarrow x_1$ and x_2 correlated

Observations

- As we change the covariance structure we change the nature of the samples
- Increasing the covariance → smoother samples
- BUT: how did we know where to place the x -axis values?
- What is the relationship between the different dimensions in the covariance Σ ?

$$\Sigma = \begin{bmatrix} \sigma_{11} & \sigma_{12} \\ \sigma_{21} & \sigma_{22} \end{bmatrix}$$

- $\sigma_{12} = \sigma_{21}$
- High values of $\sigma_{12} \Rightarrow x_1$ and x_2 correlated
- Low values of $\sigma_{12} \Rightarrow x_1$ and x_2 uncorrelated

Observations

- As we change the covariance structure we change the nature of the samples
- Increasing the covariance → smoother samples
- BUT: how did we know where to place the x -axis values?
- What is the relationship between the different dimensions in the covariance Σ ?

$$\Sigma = \begin{bmatrix} \sigma_{11} & \sigma_{12} \\ \sigma_{21} & \sigma_{22} \end{bmatrix}$$

- $\sigma_{12} = \sigma_{21}$
- High values of $\sigma_{12} \Rightarrow x_1$ and x_2 correlated
- Low values of $\sigma_{12} \Rightarrow x_1$ and x_2 uncorrelated
- High correlation produces smooth output

Observations

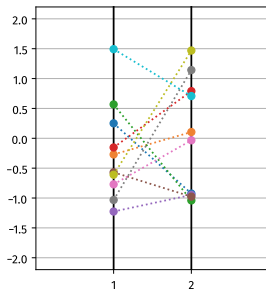
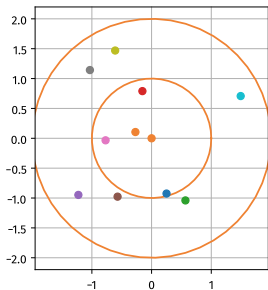
- As we change the covariance structure we change the nature of the samples
- **Increasing the covariance** → **smoother samples**
- **BUT:** how did we know where to place the x -axis values?
- **What is the relationship between the different dimensions in the covariance Σ ?**

$$\Sigma = \begin{bmatrix} \sigma_{11} & \sigma_{12} \\ \sigma_{21} & \sigma_{22} \end{bmatrix}$$

- $\sigma_{12} = \sigma_{21}$
- High values of $\sigma_{12} \Rightarrow x_1$ and x_2 correlated
- Low values of $\sigma_{12} \Rightarrow x_1$ and x_2 uncorrelated
- High correlation produces smooth output
- **Smoothness prior** $\Rightarrow$ nearby x_1 and x_2 should be correlated!

Consider a Covariance Function...

$$\Sigma(x_1, x_2) = \begin{bmatrix} \exp(-\|x_1 - x_1\|^2) \equiv 1.0 & \exp(-\|x_1 - x_2\|^2) \\ \exp(-\|x_2 - x_1\|^2) & \exp(-\|x_2 - x_2\|^2) \equiv 1.0 \end{bmatrix}$$

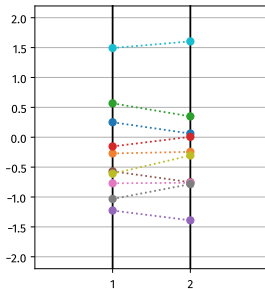
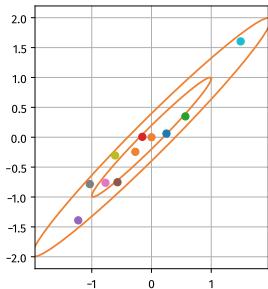


$$\Sigma = \begin{bmatrix} 1.00 & 0.00 \\ 0.00 & 1.00 \end{bmatrix}$$

$$\Sigma = \begin{bmatrix} 1.00 & 0.00 \\ 0.00 & 1.00 \end{bmatrix} \Rightarrow \exp(-\|x_1 - x_2\|^2) \text{ is small} \Rightarrow \|x_1 - x_2\|^2 \text{ is large}$$

Consider a Covariance Function...

$$\Sigma(x_1, x_2) = \begin{bmatrix} \exp(-\|x_1 - x_1\|^2) \equiv 1.0 & \exp(-\|x_1 - x_2\|^2) \\ \exp(-\|x_2 - x_1\|^2) & \exp(-\|x_2 - x_2\|^2) \equiv 1.0 \end{bmatrix}$$



$$\Sigma = \begin{bmatrix} 1.00 & 0.98 \\ 0.98 & 1.00 \end{bmatrix}$$

$$\Sigma = \begin{bmatrix} 1.00 & 0.98 \\ 0.98 & 1.00 \end{bmatrix} \Rightarrow \exp(-\|x_1 - x_2\|^2) \text{ is large} \Rightarrow \|x_1 - x_2\|^2 \text{ is small}$$

The Covariance Function $\Leftrightarrow$ Smoothness!!

- Excitement :-D
- We can now encode a prior on continuous functions!

¹but RBF can be a more general term

The Covariance Function $\Leftrightarrow$ Smoothness!!

- Excitement :-D
- We can now encode a prior on continuous functions!
- Despite the fact that $x \in \mathbb{R}$ is an **infinite** object!

¹but RBF can be a more general term

The Covariance Function $\Leftrightarrow$ Smoothness!!

- Excitement :-D
- We can now encode a prior on continuous functions!
- Despite the fact that $x \in \mathbb{R}$ is an **infinite** object!
- For any finite set $\{x_1, x_2, \dots, x_N\}$ we know how the function correlates:

$$\Sigma_{i,j} = \kappa(x_i, x_j) = \exp(-\|x_i - x_j\|^2)$$

- This is the **exponentiated quadratic** covariance function (or *kernel*)
 - (sometimes referred to as a Radial-Basis Function (RBF) kernel)¹

¹but RBF can be a more general term

Exponentiated Quadratic Covariance Function / Kernel

$$\Sigma_{i,j} = \kappa_{\text{EQ}}(x_i, x_j) = \sigma_f^2 \exp\left(-\frac{1}{2\ell^2} \|x_i - x_j\|^2\right)$$

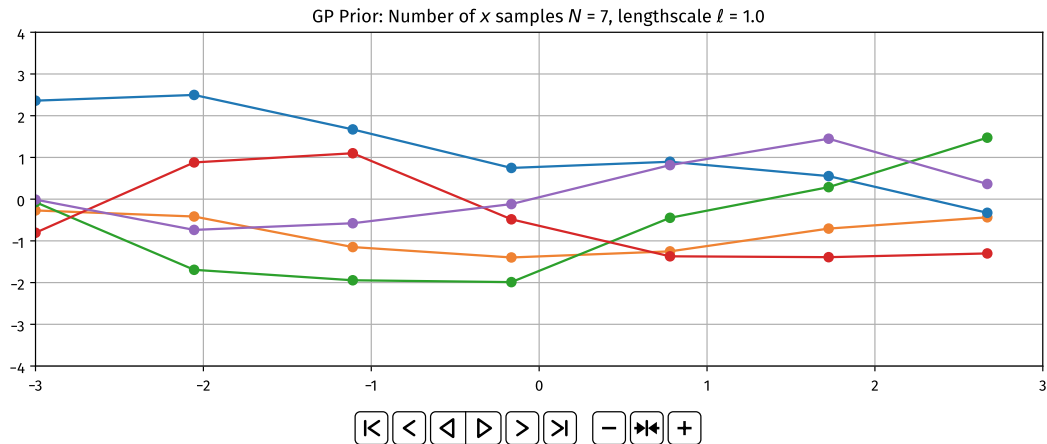
- Hyperparameters of the kernel:
 - σ_f^2 is the **variance** of the kernel
 - ℓ is the **lengthscale** of the kernel
- The length-scale modulates the smoothness vs. distance between the x locations

Exponentiated Quadratic Covariance Function / Kernel

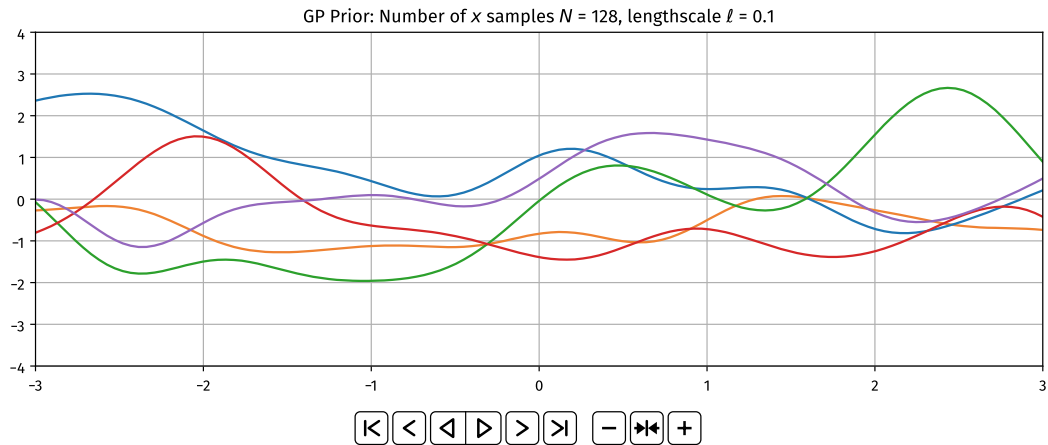
$$\Sigma_{i,j} = \kappa_{\text{EQ}}(x_i, x_j) = \sigma_f^2 \exp\left(-\frac{1}{2\ell^2} \|x_i - x_j\|^2\right)$$

- Hyperparameters of the kernel:
 - σ_f^2 is the **variance** of the kernel
 - ℓ is the **lengthscale** of the kernel
- The length-scale modulates the smoothness vs. distance between the x locations
- From this definition, we can **draw samples** from our Gaussian Process prior..

Samples from the GP Prior...



Samples from the GP Prior...



Full Definition of a GP

- More formally: a Gaussian Process is a **Random Process** or **Stochastic Process**
- In contrast to a *random variable* that adopts a distribution (e.g. $x \sim \mathcal{N}(\mu, \sigma^2)$), a stochastic process defines a **family of random variables**
- This family is defined over some *index* set $x \in \mathbb{X}$ (e.g. our input space $x \in \mathbb{R}$)
- A GP is a random process where any finite collection from the index set, e.g. $(x_1, x_2, \dots, x_N)$, define a set of jointly Gaussian distributed random variables
- A Gaussian *distribution* is defined by the mean and (co)variance
- A Gaussian *process* is defined by a **mean function** $\mu(\cdot)$ and **covariance function** $\kappa(\cdot, \cdot)$

$$f \sim \mathcal{GP}(\mu(x), \kappa(x, x'))$$

- **Note:** We usually assume the mean function to be zero (e.g. subtract it from the data before we start)

Aside on Covariance Kernels

- The use of kernels to replace feature spaces has a long history and is the source of a number of interesting machine learning algorithms
- It is **important** to note that covariance kernels are **not completely arbitrary functions!**
- They must satisfy the properties of an inner product on some feature space (although this feature space can have an infinite dimension as is the case for an exponentiated quadratic kernel)
 - They must form a symmetric matrix with the same data for both inputs
 - This matrix must be positive semi-definite (form a valid covariance matrix)
 - Many kernels are stationary, that is $\kappa(x_i, x_j) = g(|x_i - x_j|)$ (e.g. the exponentiated quadratic), however there are valid non-stationary kernels as well

What about the Data?

- We can think of the Gaussian Process as a **prior over the function**

What about the Data?

- We can think of the Gaussian Process as a **prior over the function**
- Once we have a prior we can sample from then we know we can combine it with a likelihood to perform Bayesian inference!

What about the Data?

- We can think of the Gaussian Process as a **prior over the function**
- Once we have a prior we can sample from then we know we can combine it with a likelihood to perform Bayesian inference!
- We can make this explicit by considering the unknown function $f(\cdot)$ given the noisy observations y for known inputs x

$$p(y|x) \propto p(y|f) p(f|x)$$

What about the Data?

- We can think of the Gaussian Process as a **prior over the function**
- Once we have a prior we can sample from then we know we can combine it with a likelihood to perform Bayesian inference!
- We can make this explicit by considering the unknown function $f(\cdot)$ given the noisy observations y for known inputs x

$$p(y|x) \propto p(y|f) p(f|x)$$

- Now we have the GP as the prior over $f(\cdot)$ so we can write

$$f \sim p(f|x) = \mathcal{GP}(0, \kappa_{\text{EQ}}(x, x'))$$

What about the Data?

- We can think of the Gaussian Process as a **prior over the function**
- Once we have a prior we can sample from then we know we can combine it with a likelihood to perform Bayesian inference!
- We can make this explicit by considering the unknown function $f(\cdot)$ given the noisy observations y for known inputs x

$$p(y|x) \propto p(y|f) p(f|x)$$

- Now we have the GP as the prior over $f(\cdot)$ so we can write

$$f \sim p(f|x) = \mathcal{GP}(0, \kappa_{\text{EQ}}(x, x'))$$

- Equally, we have our standard Gaussian likelihood for the observation noise

$$p(y|f) = \mathcal{N}(y|f, \sigma_n^2)$$

What about the Data?

- We can think of the Gaussian Process as a **prior over the function**
- Once we have a prior we can sample from then we know we can combine it with a likelihood to perform Bayesian inference!
- We can make this explicit by considering the unknown function $f(\cdot)$ given the noisy observations y for known inputs x

$$p(y|x) \propto p(y|f) p(f|x)$$

- Now we have the GP as the prior over $f(\cdot)$ so we can write

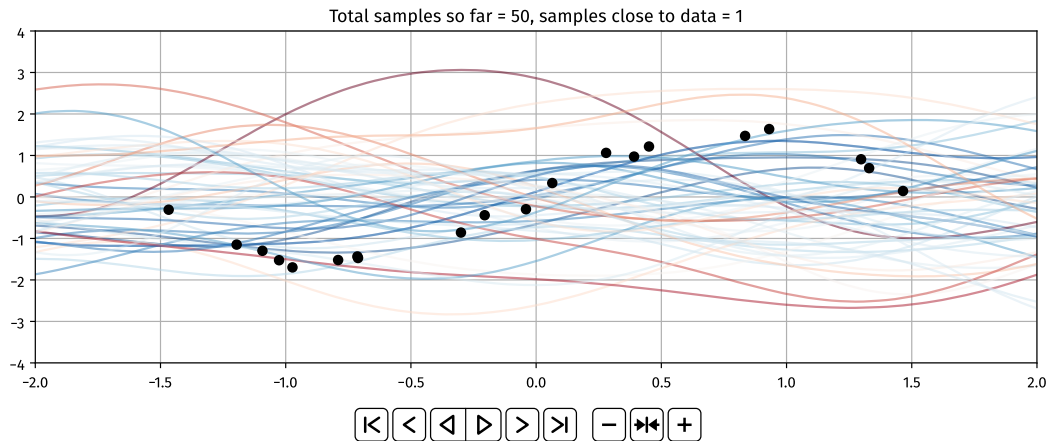
$$f \sim p(f|x) = \mathcal{GP}(0, \kappa_{\text{EQ}}(x, x'))$$

- Equally, we have our standard Gaussian likelihood for the observation noise

$$p(y|f) = \mathcal{N}(y|f, \sigma_n^2)$$

- So, we sample from the GP and **weigh** those samples according to the likelihood...

What about the Data?



Can we do Better than Sampling?

- **Excitement!:** GP's are a rare case with a closed form of Bayes' Rule!
- Caveat: This is when we have a Gaussian likelihood model for the noise
- No surprises - this comes from the product of Gaussians $\rightarrow$ Gaussian

$$\begin{aligned} p(\mathbf{y} | \mathbf{x}) &= \int p(\mathbf{y} | f) p(f | \mathbf{x}) df \\ &= \int \mathcal{N}(\mathbf{y} | f, \sigma_n^2 I) \mathcal{GP}(\mathbf{0}, \kappa(\mathbf{x}, \mathbf{x})) df \\ &= \int \mathcal{N}(\mathbf{y} | f, \sigma_n^2 I) \mathcal{N}(f | \mathbf{0}, K_{\mathbf{x}, \mathbf{x}}) df \\ &= \mathcal{N}(\mathbf{y} | \mathbf{0}, K_{\mathbf{x}, \mathbf{x}} + \sigma_n^2 I) \end{aligned}$$

- Here construct the covariance matrix $K_{\mathbf{x}, \mathbf{x}}$ via $[K_{\mathbf{x}, \mathbf{x}}]_{i,j} := \kappa(x_i, x_j)$

What about a new data point?

- Construction that lead to $p(\mathbf{y} | \mathbf{x}) = \mathcal{N}(\mathbf{y} | \mathbf{0}, K_{\mathbf{x}, \mathbf{x}} + \sigma_n^2 I)$ true for arbitrary $\{\mathbf{x}, \mathbf{y}\}$
- Now let us add an extra **test** input and output (x^*, y^*)

$$p(\mathbf{y}, y^* | \mathbf{x}, x^*) = \mathcal{N}(\mathbf{y} | \mathbf{0}, K_{\mathbf{x}, \mathbf{x}} + \sigma_n^2 I)$$

- We can write this joint distribution as

$$\begin{bmatrix} \mathbf{y} \\ y^* \end{bmatrix} \sim \mathcal{N} \left(\mathbf{0}, \begin{bmatrix} \kappa(\mathbf{x}, \mathbf{x}) & \kappa(\mathbf{x}, x^*) \\ \kappa(x^*, \mathbf{x}) & \kappa(x^*, x^*) \end{bmatrix} \right)$$

What about a new data point?

- Construction that lead to $p(\mathbf{y} | \mathbf{x}) = \mathcal{N}(\mathbf{y} | \mathbf{0}, K_{\mathbf{x}, \mathbf{x}} + \sigma_n^2 I)$ true for arbitrary $\{\mathbf{x}, \mathbf{y}\}$
- Now let us add an extra **test** input and output (x^*, y^*)

$$p(\mathbf{y}, y^* | \mathbf{x}, x^*) = \mathcal{N}(\mathbf{y} | \mathbf{0}, K_{\mathbf{x}, \mathbf{x}} + \sigma_n^2 I)$$

- We can write this joint distribution as

$$\begin{bmatrix} \mathbf{y} \\ y^* \end{bmatrix} \sim \mathcal{N} \left(\mathbf{0}, \begin{bmatrix} \kappa(\mathbf{x}, \mathbf{x}) & \kappa(\mathbf{x}, x^*) \\ \kappa(x^*, \mathbf{x}) & \kappa(x^*, x^*) \end{bmatrix} \right)$$

- Use the properties of Gaussians to find $p(y^* | \mathbf{y}, \mathbf{x}, x^*)$ in **closed form**!

What about a new data point?

- Construction that lead to $p(\mathbf{y} | \mathbf{x}) = \mathcal{N}(\mathbf{y} | \mathbf{0}, K_{\mathbf{x}, \mathbf{x}} + \sigma_n^2 I)$ true for arbitrary $\{\mathbf{x}, \mathbf{y}\}$
- Now let us add an extra **test** input and output (x^*, y^*)

$$p(\mathbf{y}, y^* | \mathbf{x}, x^*) = \mathcal{N}(\mathbf{y} | \mathbf{0}, K_{\mathbf{x}, \mathbf{x}} + \sigma_n^2 I)$$

- We can write this joint distribution as

$$\begin{bmatrix} \mathbf{y} \\ y^* \end{bmatrix} \sim \mathcal{N} \left(\mathbf{0}, \begin{bmatrix} \kappa(\mathbf{x}, \mathbf{x}) & \kappa(\mathbf{x}, x^*) \\ \kappa(x^*, \mathbf{x}) & \kappa(x^*, x^*) \end{bmatrix} \right)$$

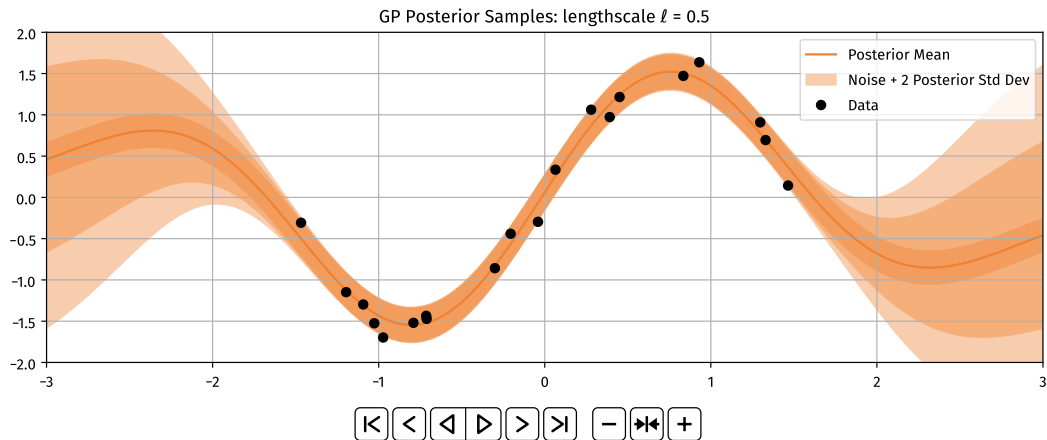
- Use the properties of Gaussians to find $p(y^* | \mathbf{y}, \mathbf{x}, x^*)$ in **closed form**!

$$p(y^* | x^*, \mathbf{x}, \mathbf{y}) = \mathcal{N}(\mu^*, \Sigma^*)$$

$$\mu^* = \kappa(x^*, \mathbf{x}) \kappa(\mathbf{x}, \mathbf{x})^{-1} \mathbf{y}$$

$$\Sigma^* = \kappa(x^*, x^*) - \kappa(x^*, \mathbf{x}) \kappa(\mathbf{x}, \mathbf{x})^{-1} \kappa(\mathbf{x}, x^*)$$

Samples from the posterior (closed form!):



Learning the Hyper-Parameters

$$\Sigma_{i,j} = \kappa(x_i, x_j) = \sigma_f^2 \exp\left(-\frac{1}{2\ell^2} \|x_i - x_j\|^2\right)$$

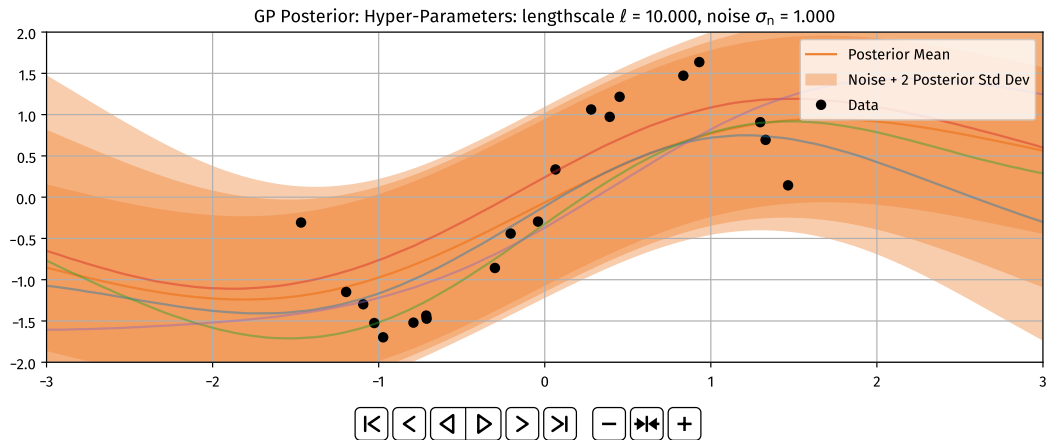
- We have to **learn the values of the hyper-parameters** from our training data
- In particular, for the exponentiated quadratic covariance we have the **lengthscale** ℓ and the **signal variance** σ_f^2

Learning the Hyper-Parameters

$$\Sigma_{i,j} = \kappa(x_i, x_j) = \sigma_f^2 \exp\left(-\frac{1}{2\ell^2} \|x_i - x_j\|^2\right)$$

- We have to **learn the values of the hyper-parameters** from our training data
- In particular, for the exponentiated quadratic covariance we have the **lengthscale** ℓ and the **signal variance** σ_f^2
- We also need to estimate the **noise variance** σ_n^2
- *(We do this in a similar manner to the hyperparameters but machine learners differ over whether or not the noise variance is technically a hyper-parameter or a model parameter)*

Learning the GP Hyper-Parameters



How do we Find the Balance?

- Use the **marginal log-likelihood** to obtain the parameters²

$$\begin{aligned}\log p(\mathbf{y} \mid \mathbf{x}, \ell, \sigma_n^2) &= \log \mathcal{N}(\mathbf{y} \mid \mathbf{0}, \kappa(\mathbf{x}, \mathbf{x} \mid \ell) + \sigma_n^2 \mathbf{I}) \\ &= -\frac{N}{2} \log(2\pi) - \frac{1}{2} \log |\mathbf{K}| - \frac{1}{2} \text{Tr} [\mathbf{K}^{-1} \mathbf{y} \mathbf{y}^\top]\end{aligned}$$

with $\mathbf{K} := \kappa(\mathbf{x}, \mathbf{x} \mid \ell) + \sigma_n^2 \mathbf{I}$

²Type-II Maximum Likelihood

How do we Find the Balance?

$$\log p(\mathbf{y}|\mathbf{x}, \ell, \sigma_n^2) \propto -\frac{1}{2} \log |\mathbf{K}| - \frac{1}{2} \text{Tr} [\mathbf{K}^{-1} \mathbf{y} \mathbf{y}^\top]$$

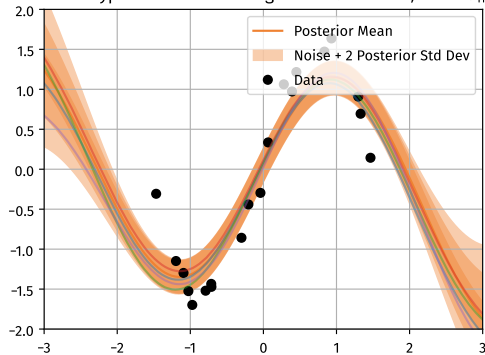
$$\text{with } \mathbf{K} = \kappa(\mathbf{x}, \mathbf{x}|\ell) + \sigma_n^2 \mathbf{I}$$

The **Marginal Log Likelihood** is a trade-off between two terms:

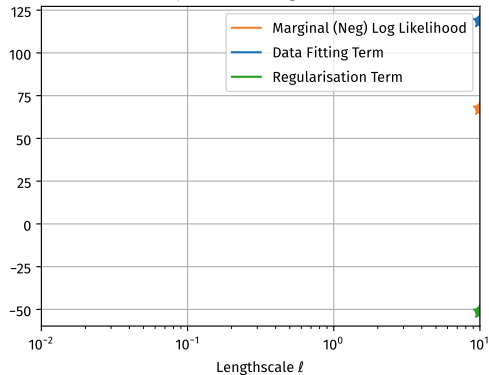
- Data-fit term $\text{Tr} [\mathbf{K}^{-1} \mathbf{y} \mathbf{y}^\top]$ is optimal when the covariance function explains the training data perfectly (that is $\mathbf{K} = \mathbf{y} \mathbf{y}^\top$)
- Bayesian regularisation term $\log |\mathbf{K}|$ penalises complexity in the kernel and favours simpler solutions (maximum penalty for $\mathbf{K} = \mathbf{I}$)

GP Optimising Hyper-Parameters

GP Posterior: Hyper-Parameters: lengthscale $\ell = 10.000$, noise $\sigma_n = 0.200$



Objectives vs lengthscale ℓ



How has the GP Helped?

Looking back at the Bayesian Regression choices:

How has the GP Helped?

Looking back at the Bayesian Regression choices:

1. How many basis functions (M) should we use?
2. Where should they be located?
 - The GP takes has infinite capacity ($M \rightarrow \infty$) so we effectively have basis functions located everywhere (through the kernel trick)

How has the GP Helped?

Looking back at the Bayesian Regression choices:

1. How many basis functions (M) should we use?
2. Where should they be located?
 - The GP takes has infinite capacity ($M \rightarrow \infty$) so we effectively have basis functions located everywhere (through the kernel trick)
3. What about the lengthscale ℓ ?
 - We can fit the **hyperparameters** of the covariance kernels (for example the lengthscale) by maximising the marginal log-likelihood

How has the GP Helped?

Looking back at the Bayesian Regression choices:

1. How many basis functions (M) should we use?
2. Where should they be located?
 - The GP takes has infinite capacity ($M \rightarrow \infty$) so we effectively have basis functions located everywhere (through the kernel trick)
3. What about the lengthscale ℓ ?
 - We can fit the **hyperparameters** of the covariance kernels (for example the lengthscale) by maximising the marginal log-likelihood
4. How does the prior α affect things?
 - We can fold this term into the covariance kernel (usually as the signal variance hyperparameter) and estimate it in the same manner as the lengthscale via the marginal log-likelihood

Summary

- We have motivated the use of Gaussian Process priors for regression and shown that they provide a **non-parametric** framework for Bayesian regression that automatically take care of many of the (arbitrary) decisions from **parametric** Bayesian regression

Are we all done?

- The models so far have some limitations:
 1. Computational scaling with data
 2. Only work for a Gaussian likelihood
 3. Point estimates for hyperparameters
 4. Only shown models for regression (**supervised** learning)
 5. Can extend ideas into **unsupervised** learning

Application: Bayesian Optimisation

Overview...

Motivation

No Free Lunch

Whirlwind Introduction to Inverse Probabilities

Model Selection

Gaussian Processes (a Prior over Functions)

Application: Bayesian Optimisation

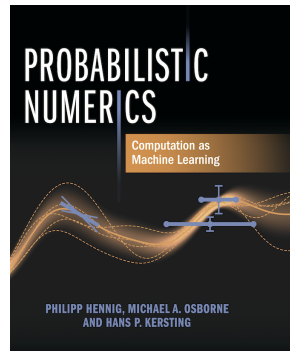
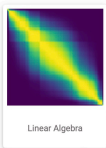
Evaluation

Conclusions

Probabilistic Numerics!

Probabilistic Numerics: Computation as Machine Learning

- Numerical problem $\rightarrow$ statistical inference
- Encode structural knowledge as a **prior**
- Uncertainty from all sources:
 - Stochastic input
 - **Numerical Computation** itself!
- Example applications:



<https://www.probabilistic->

[numerics.org/assets/ProbabilisticNumerics.pdf](https://www.probabilistic-numerics.org/assets/ProbabilisticNumerics.pdf)

<https://www.probabilistic-numerics.org>

Bayesian Optimisation Applications

AI-Assisted ML

Deep learning has seen great success across a wide variety of domains. The best neural architectures are often carefully constructed by seasoned deep learning experts in each domain. For example, years of experimentation have shown how to arrange bidirectional transformers to work well for language tasks and dilated separable convolutions for image tasks. A relatively new sub-field of deep-learning deals with automated machine learning, or as we prefer to call it: AI-assisted machine learning. The fundamental idea is that AI will create a first pass of the deep-learning model given a use-case or a dataset. Developers/data scientists can then either use that model directly or fine-tune. We are conducting cutting-edge research in the main pillars of AI-Assisted ML: hyperparameter optimization (HPO) and neural architecture search (NAS).



Hyperparameter optimization

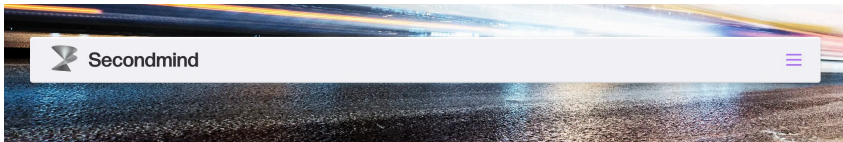
When developing a deep learning model, there are many knobs and dials to tune that depend on the specific task and dataset at hand. For example, setting the learning rate too high can prevent the algorithm from converging. Setting the learning rate too low can cause the algorithm to get stuck at a local minimum. There are countless other hyperparameters such as the number of epochs, batch size, momentum, regularization, shape, and size of the neural network. These hyperparameters are all dependent on each other and interact in intricate ways, so finding the best hyperparameters for a given dataset is an extremely difficult and highly nonconvex optimization problem.

Randomly testing different sets of hyperparameters may eventually find a decent solution but could take years of computation time. Efficiently tuning deep learning hyperparameters is an active area of research. Five years ago, the best algorithms weren't much better than random search. Now algorithms are capable of orders of magnitude speedups. At Abacus.AI, we use state-of-the-art HPO while training all our models.



Neural Architecture Search

Neural architecture search (NAS) is a rapidly developing area of research in which the process of choosing the best architecture is automated.



Secondmind expands strategic partnership with Mazda

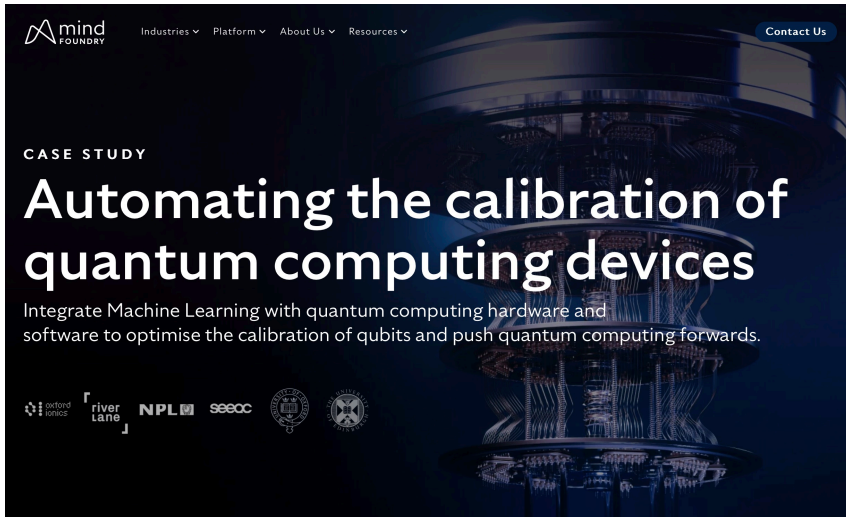
Date November 28, 2023

Author Kate Matthews

Secondmind expands strategic partnership with Mazda to accelerate vehicle design and development through advanced AI

Mazda joins existing Secondmind investors in new \$16 million funding round

Secondmind announced today the expansion of its long-standing strategic



mind
FOUNDRY

Industries ▾ Platform ▾ About Us ▾ Resources ▾

[Contact Us](#)

CASE STUDY

Automating the calibration of quantum computing devices

Integrate Machine Learning with quantum computing hardware and software to optimise the calibration of qubits and push quantum computing forwards.

[Home](#)[Dates](#)[Calls](#) ▾[Competitions](#) ▾[Venue](#)[Organizers](#)[Blog](#)[Ethics](#)[Program](#) ▾[Attend](#) ▾[Previous edition](#)

AutoML Conference 2023

📍 Potsdam/Berlin, Germany

📅 September 12th – 15th 2023

[Photos](#)[Accepted papers](#)[YT Channel](#)

Goals

The international conference on automated machine learning (AutoML) is the premier gathering of professionals focussed on the progressive automation of machine learning (ML), aiming to develop automated methods for making ML methods more efficient, robust, trustworthy, and available to everyone. A special focus of the AutoML conference lies on openness: via sharing of code, we hope to facilitate a culture of open collaborations across academic and industrial partners.

Optimisation problem

- We want to solve

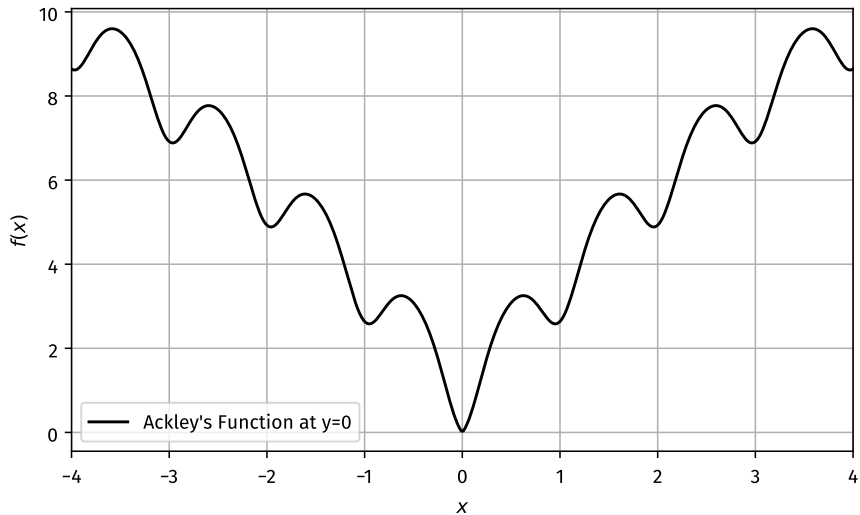
$$\mathbf{x}^* = \arg \min_{\mathbf{x} \in \mathcal{X}} f(\mathbf{x})$$

- We have a method to evaluate f but it is explicitly unknown (and non-convex!)
- Evaluations of f are expensive (in cost or time)
- Evaluations of f may not be precise (noisy)
- We assume it is too expensive to calculate gradients >- More formally:

$f : \mathcal{X} \rightarrow \mathbb{R}$ is 'well behaved' within a bounded domain $\mathcal{X} \subseteq \mathbb{R}^D$

e.g. the function is *Lipschitz Continuous* such that $|f(\mathbf{x}_1) - f(\mathbf{x}_2)| \leq L \|\mathbf{x}_1 - \mathbf{x}_2\|$ for some distances $|\cdot|$ and $\|\cdot\|$, and some **known constant** $L \geq 0$

Optimisation problem illustration



Important (and expensive) problems!

- Thinking outside the box: Many tasks can be posed as optimisation problems..
 - Science - e.g. function evaluation might be an experiment to conduct
 - Engineering - e.g. optimisation of designs might require simulations
 - Machine Learning - e.g. hyperparameter search, problems without gradients
 - Robotics/RL - e.g. function evaluation could be a full roll-out
 - Scheduling - e.g. complex interactions remove convexity

Important (and expensive) problems!

- Thinking outside the box: Many tasks can be posed as optimisation problems..
 - Science - e.g. function evaluation might be an experiment to conduct
 - Engineering - e.g. optimisation of designs might require simulations
 - Machine Learning - e.g. hyperparameter search, problems without gradients
 - Robotics/RL - e.g. function evaluation could be a full roll-out
 - Scheduling - e.g. complex interactions remove convexity
- Can run over non-vector spaces (e.g. molecule design or drug discovery)
- Can optimise graphs (e.g. causal discovery)
- Bayesian experimental design $\Rightarrow$ automate science!

Important (and expensive) problems!

- Thinking outside the box: Many tasks can be posed as optimisation problems..
 - Science - e.g. function evaluation might be an experiment to conduct
 - Engineering - e.g. optimisation of designs might require simulations
 - Machine Learning - e.g. hyperparameter search, problems without gradients
 - Robotics/RL - e.g. function evaluation could be a full roll-out
 - Scheduling - e.g. complex interactions remove convexity
- Can run over non-vector spaces (e.g. molecule design or drug discovery)
- Can optimise graphs (e.g. causal discovery)
- Bayesian experimental design $\Rightarrow$ automate science!
- Leads to the concept of **black-box optimisation**!

How to optimise a black-box function?

1. Use previous knowledge?

“Grad student descent” - tweak by hand using experience and hope for the best..

How to optimise a black-box function?

1. Use previous knowledge?

“Grad student descent” - tweak by hand using experience and hope for the best..

2. Search over a grid of values?

If L-Lipschitz such that $|f(\mathbf{x}_1) - f(\mathbf{x}_2)| \leq L\|\mathbf{x}_1 - \mathbf{x}_2\|$ (and noise free), then to guarantee a precision of ε we would need $(L/\varepsilon)^D$ evaluations! (e.g. $(1/0.01)^5 = 10$ billion)

How to optimise a black-box function?

1. Use previous knowledge?

“Grad student descent” - tweak by hand using experience and hope for the best..

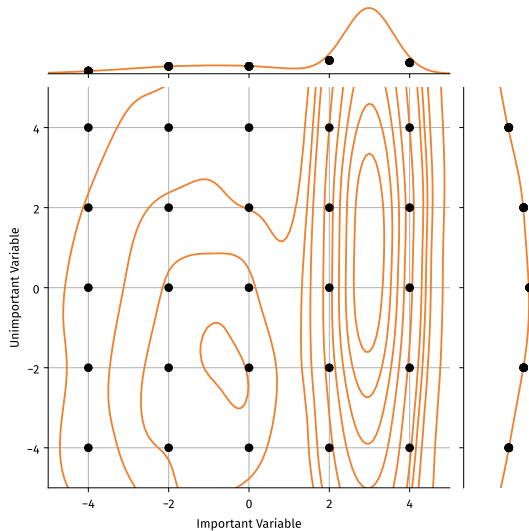
2. Search over a grid of values?

If L-Lipschitz such that $|f(\mathbf{x}_1) - f(\mathbf{x}_2)| \leq L\|\mathbf{x}_1 - \mathbf{x}_2\|$ (and noise free), then to guarantee a precision of ε we would need $(L/\varepsilon)^D$ evaluations! (e.g. $(1/0.01)^5 = 10$ billion)

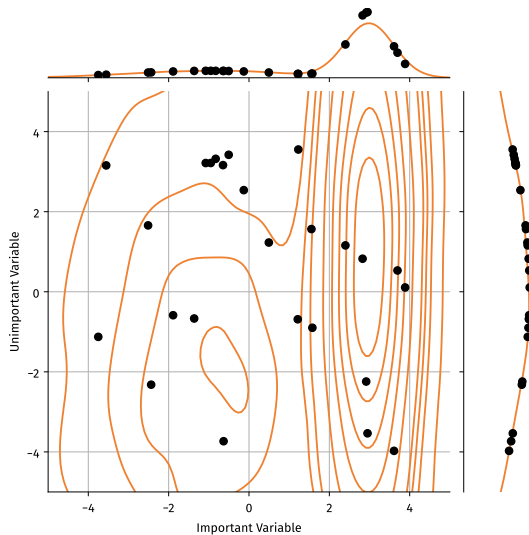
3. Random search?

Might well be better than grid search (e.g. if not all variables are equally important) but still expensive for good coverage

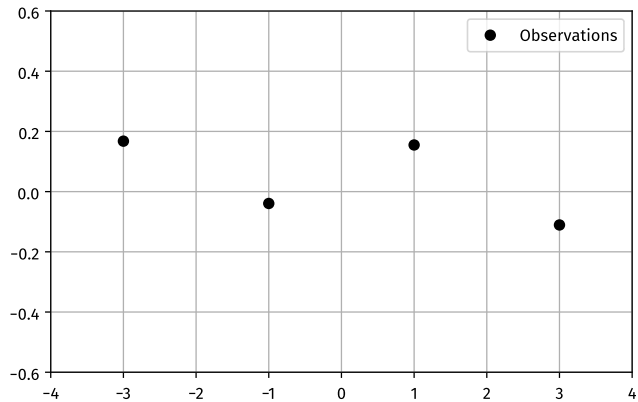
Grid search with an unimportant dimension..



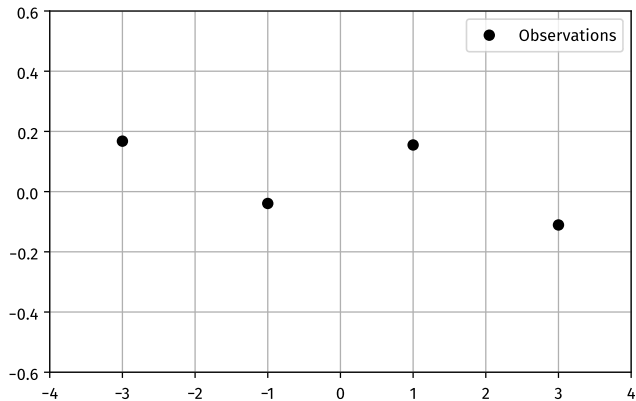
Random sampling with an unimportant dimension..



Let's have a go..

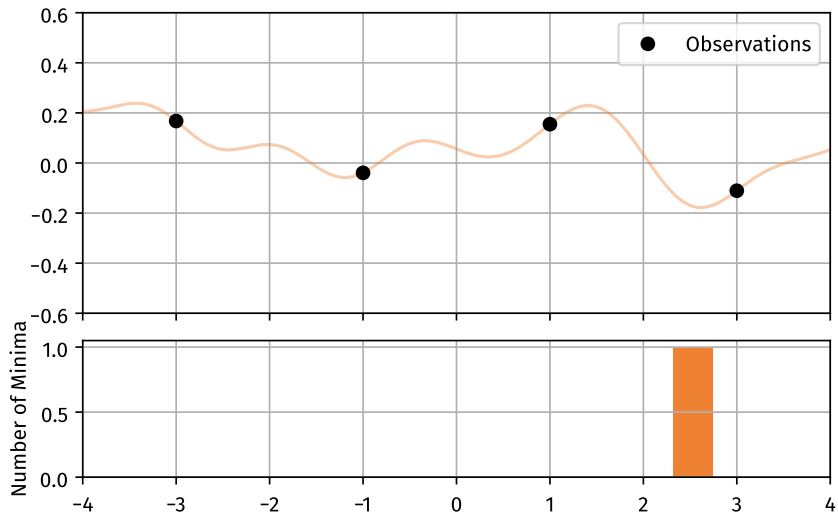


Let's have a go..

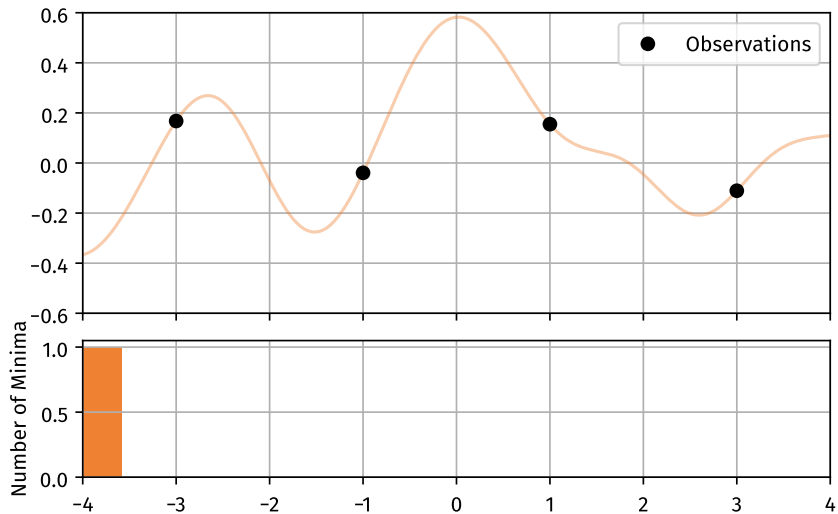


Where's the minimum?

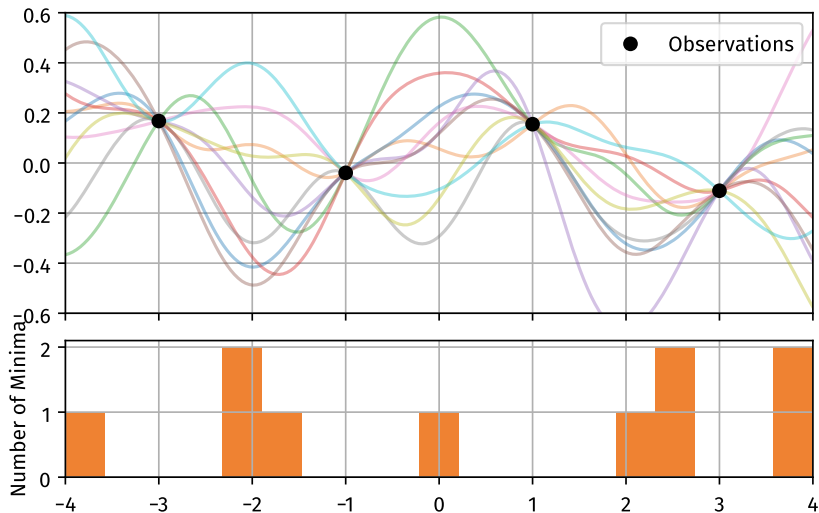
First attempt..



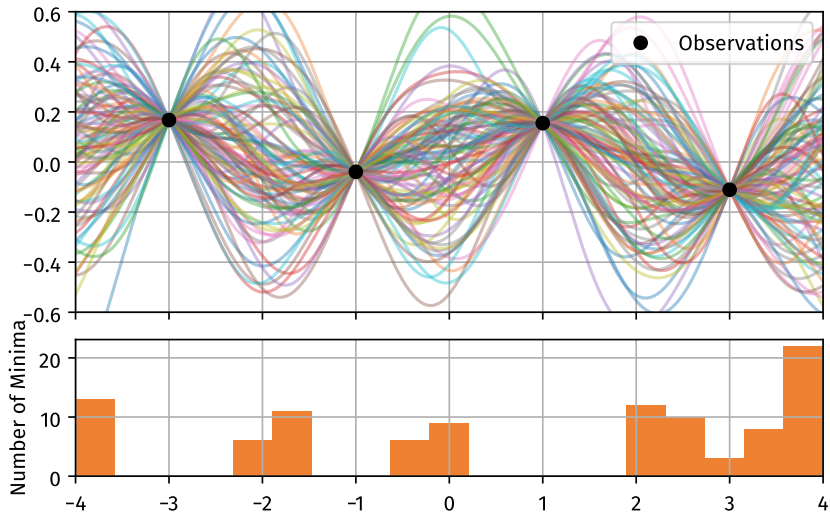
Second attempt..



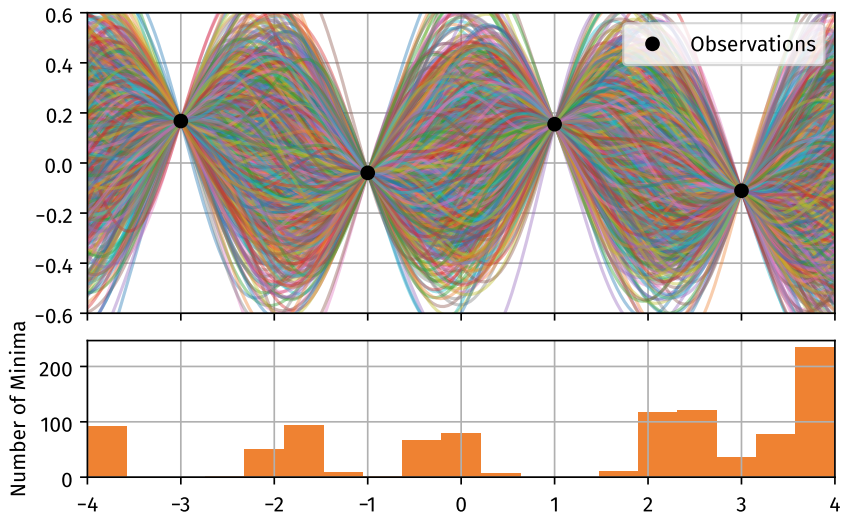
Many attempts..



Many many attempts..

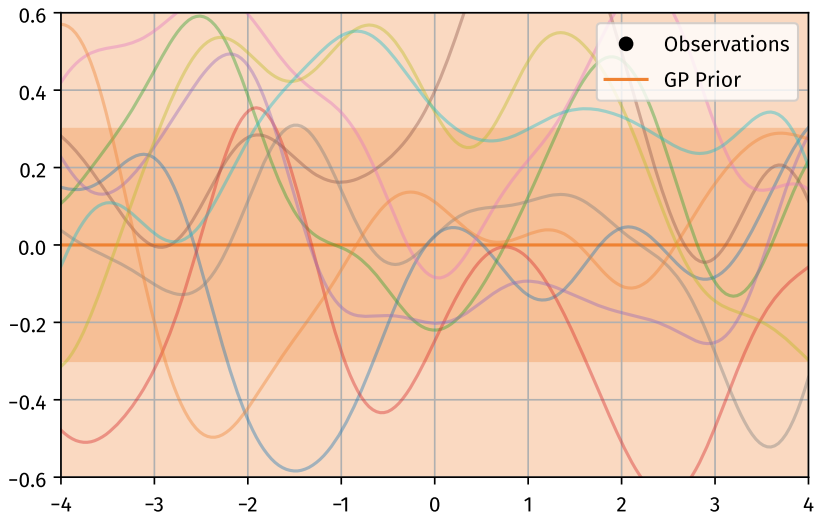


Many many attempts..

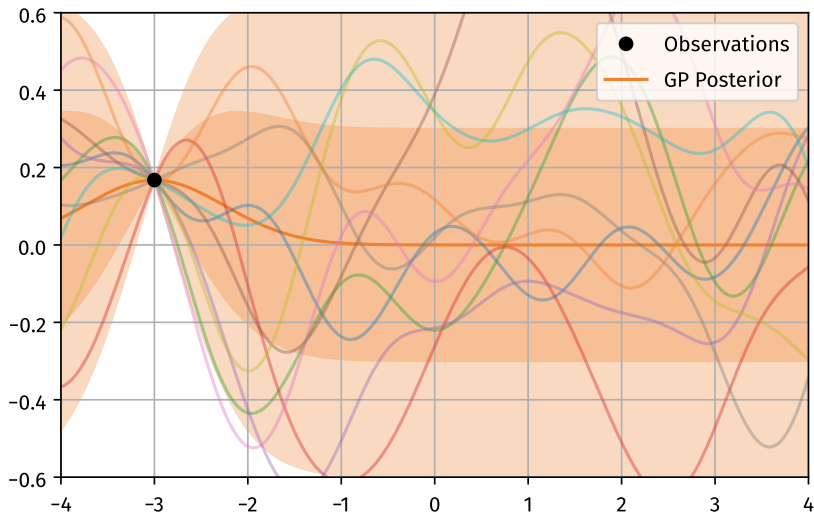


But remember the Gaussian Process as a posterior over functions..

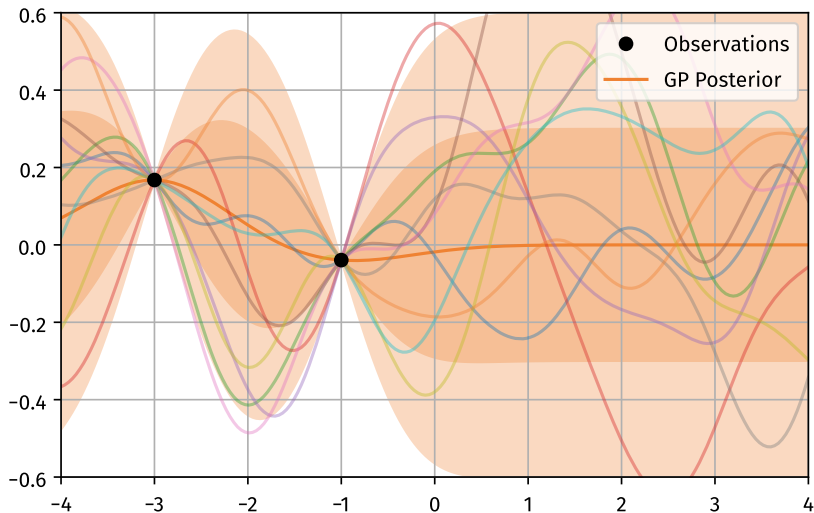
But remember the Gaussian Process as a posterior over functions..



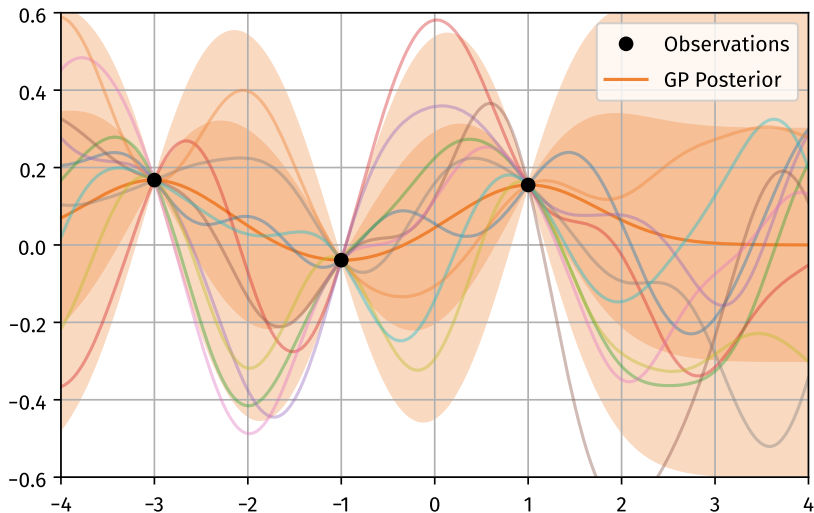
But remember the Gaussian Process as a posterior over functions..



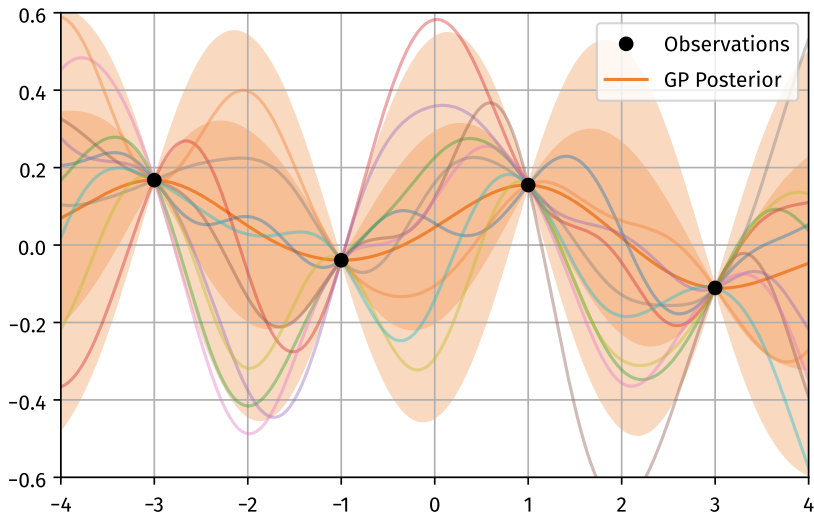
But remember the Gaussian Process as a posterior over functions..



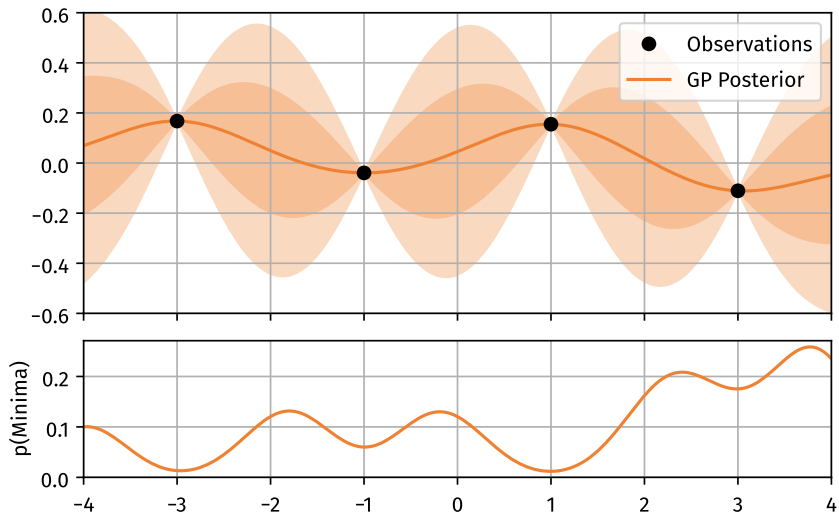
But remember the Gaussian Process as a posterior over functions..



But remember the Gaussian Process as a posterior over functions..



Ideal solution..



Use of “surrogate modelling” for optimisation

- IDEA: Replace complex objective f with a surrogate model

Use of “surrogate modelling” for optimisation

- IDEA: Replace complex objective f with a surrogate model
- This surrogate model is trained from **observations**

Use of “surrogate modelling” for optimisation

- IDEA: Replace complex objective f with a surrogate model
- This surrogate model is trained from **observations**
- QUESTION: *How should we collect new observations to get the best surrogate?*

Use of “surrogate modelling” for optimisation

- IDEA: Replace complex objective f with a surrogate model
- This surrogate model is trained from **observations**
- QUESTION: *How should we collect new observations to get the best surrogate?*

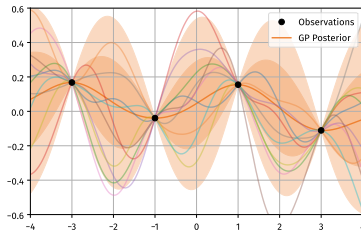
Use of “surrogate modelling” for optimisation

- IDEA: Replace complex objective f with a **surrogate model**
 - This surrogate model is trained from **observations**
 - QUESTION: *How should we collect new observations to get the best surrogate?*
1. Define some utility (or loss) function to collect new data
 - These should satisfy some optimal criteria
 - **Optimisation as a decision!**

Use of “surrogate modelling” for optimisation

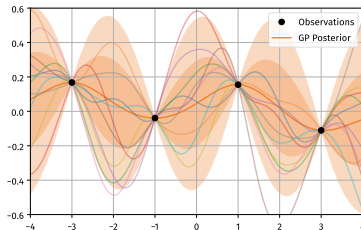
- IDEA: Replace complex objective f with a **surrogate model**
 - This surrogate model is trained from **observations**
 - QUESTION: *How should we collect new observations to get the best surrogate?*
1. Define some utility (or loss) function to collect new data
 - These should satisfy some optimal criteria
 - **Optimisation as a decision!**
 2. Each sequential decision problem posed as *inference* using the surrogate model
 - Trade off exploration/exploitation for new data points
 - Need calibrated uncertainty (epistemic and aleatoric)

Surrogate modelling using GP emulator..



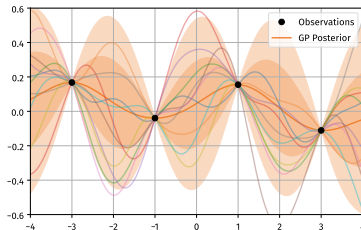
- Often use a Gaussian Process “emulator” due to good uncertainty calibration

Surrogate modelling using GP emulator..



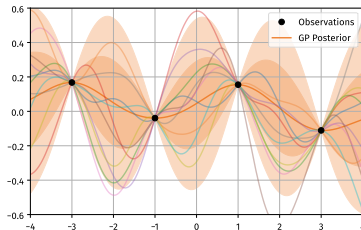
- Often use a Gaussian Process “emulator” due to good uncertainty calibration
- Don’t have to! Could use any Bayesian model (to provide uncertainty)

Surrogate modelling using GP emulator..



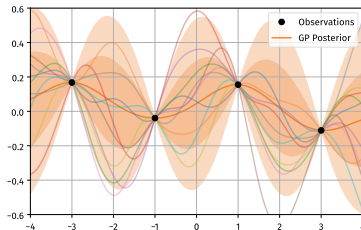
- Often use a Gaussian Process “emulator” due to good uncertainty calibration
- Don’t have to! Could use any Bayesian model (to provide uncertainty)
- Helpful to have posterior computed efficiently in closed form

Surrogate modelling using GP emulator..



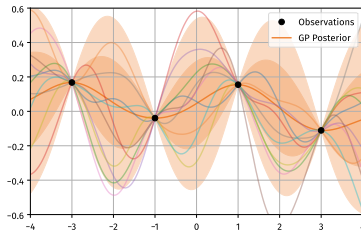
- Often use a Gaussian Process “emulator” due to good uncertainty calibration
- Don’t have to! Could use any Bayesian model (to provide uncertainty)
- Helpful to have posterior computed efficiently in closed form
 - *Surrogate needs to be cheaper than the original objective!*

Surrogate modelling using GP emulator..



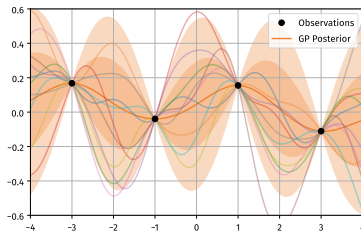
- Often use a Gaussian Process “emulator” due to good uncertainty calibration
- Don’t have to! Could use any Bayesian model (to provide uncertainty)
- Helpful to have posterior computed efficiently in closed form
 - *Surrogate needs to be cheaper than the original objective!*
- Other models could include:

Surrogate modelling using GP emulator..



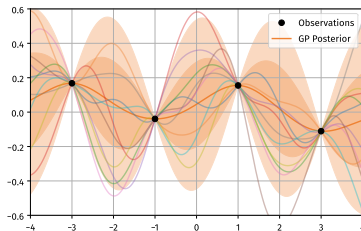
- Often use a Gaussian Process “emulator” due to good uncertainty calibration
- Don’t have to! Could use any Bayesian model (to provide uncertainty)
- Helpful to have posterior computed efficiently in closed form
 - *Surrogate needs to be cheaper than the original objective!*
- Other models could include:
 - *Student-T processes (robust), Random Trees and Forests, Bayesian Neural Networks, etc...*

Surrogate modelling using GP emulator..



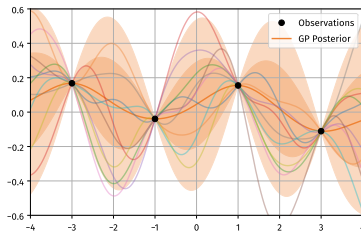
- Original problem: $\mathbf{x}^* = \arg \min_{\mathbf{x} \in \mathcal{X}} f(\mathbf{x})$
- Surrogate: $s(\mathbf{x}) \approx f(\mathbf{x})$

Surrogate modelling using GP emulator..



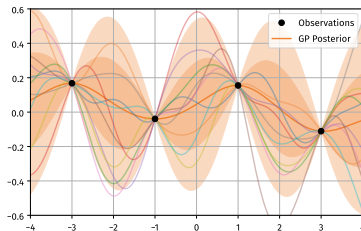
- Original problem: $\mathbf{x}^* = \arg \min_{\mathbf{x} \in \mathcal{X}} f(\mathbf{x})$
- Surrogate: $s(\mathbf{x}) \approx f(\mathbf{x})$
- Use GP Prior: $s(\mathbf{x}) \sim \mathcal{GP}(\mu_{\theta}(\mathbf{x}), K_{\theta}(\mathbf{x}, \mathbf{x}'))$
- Observations so far: $\mathcal{D} := \{X, Y\}$
- GP Posterior: $s(\mathbf{x} \mid \mathcal{D}) := \mathcal{N}(\mu_{\theta}(\mathbf{x} \mid \mathcal{D}), \sigma_{\theta}^2(\mathbf{x} \mid \mathcal{D}))$

Surrogate modelling using GP emulator..



- Original problem: $\mathbf{x}^* = \arg \min_{\mathbf{x} \in \mathcal{X}} f(\mathbf{x})$
- Surrogate: $s(\mathbf{x}) \approx f(\mathbf{x})$
- Use GP Prior: $s(\mathbf{x}) \sim \mathcal{GP}(\mu_\theta(\mathbf{x}), K_\theta(\mathbf{x}, \mathbf{x}'))$
- Observations so far: $\mathcal{D} := \{X, Y\}$
- GP Posterior: $s(\mathbf{x} \mid \mathcal{D}) := \mathcal{N}(\mu_\theta(\mathbf{x} \mid \mathcal{D}), \sigma_\theta^2(\mathbf{x} \mid \mathcal{D}))$
- Exploration and exploitation can be traded-off via $\mu_\theta(\mathbf{x} \mid \mathcal{D})$ and $\sigma_\theta(\mathbf{x} \mid \mathcal{D})$

Surrogate modelling using GP emulator..



- Original problem: $\mathbf{x}^* = \arg \min_{\mathbf{x} \in \mathcal{X}} f(\mathbf{x})$
- Surrogate: $s(\mathbf{x}) \approx f(\mathbf{x})$
- Use GP Prior: $s(\mathbf{x}) \sim \mathcal{GP}(\mu_\theta(\mathbf{x}), K_\theta(\mathbf{x}, \mathbf{x}'))$
- Observations so far: $\mathcal{D} := \{X, Y\}$
- GP Posterior: $s(\mathbf{x} \mid \mathcal{D}) := \mathcal{N}(\mu_\theta(\mathbf{x} \mid \mathcal{D}), \sigma_\theta^2(\mathbf{x} \mid \mathcal{D}))$
- Exploration and exploitation can be traded-off via $\mu_\theta(\mathbf{x} \mid \mathcal{D})$ and $\sigma_\theta(\mathbf{x} \mid \mathcal{D}) \Rightarrow \text{Yay :-)}$

The acquisition function

- Recall: Optimisation as a sequential decision process...

The acquisition function

- Recall: Optimisation as a sequential decision process...
- We need to decide where to “sample” the objective to get the next data point

The acquisition function

- Recall: Optimisation as a sequential decision process...
- We need to decide where to “sample” the objective to get the next data point
- **Acquisition function!** A proxy measure of where to sample

The acquisition function

- Recall: Optimisation as a sequential decision process...
- We need to decide where to “sample” the objective to get the next data point
- **Acquisition function!** A proxy measure of where to sample
- Optimising the acquisition function should yield a good trade-off between exploration and exploitation

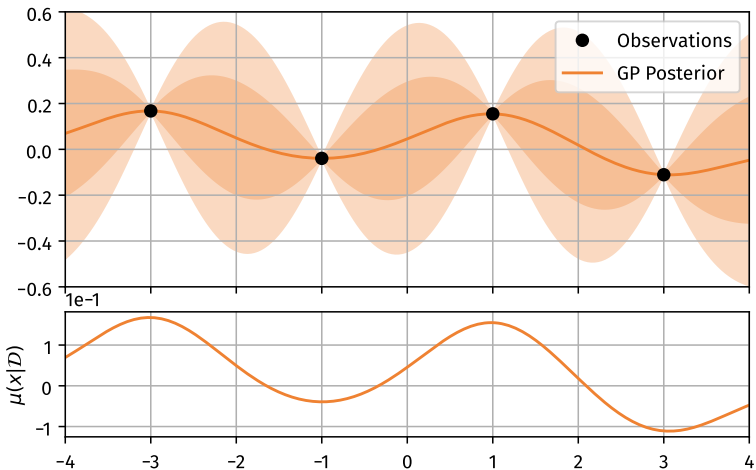
Does the surrogate need to be a good model for the objective everywhere?

Does the surrogate need to be a good model for the objective everywhere?

- Maybe only needs to be “good” around the minimum and “good enough” elsewhere...

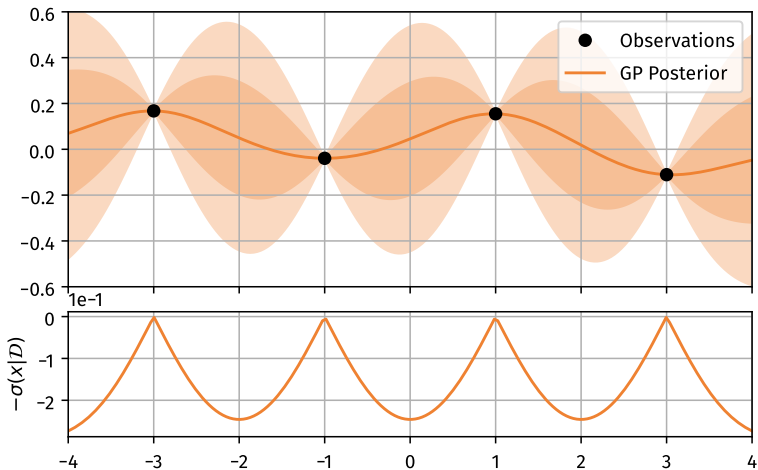
The acquisition function

Mean function $\leftrightarrow$ exploitation (best guess given current data)

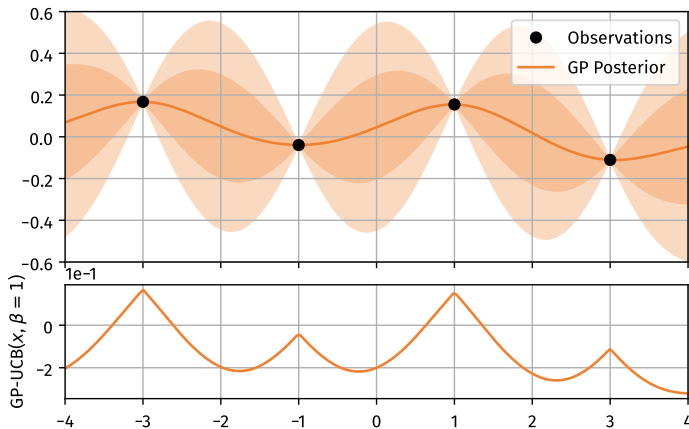


The acquisition function

Std deviation function $\leftrightarrow$ exploration (away from data)

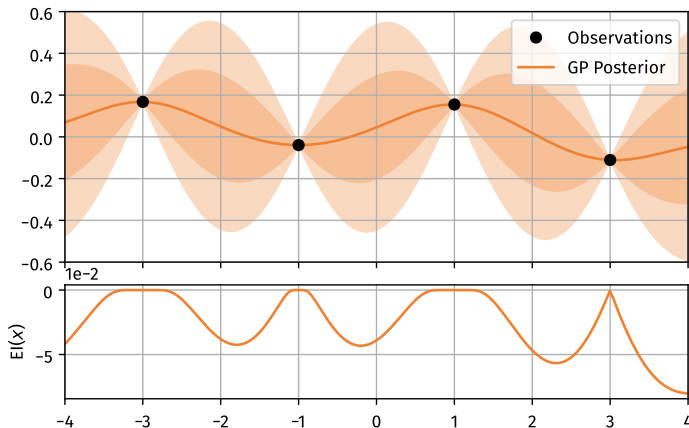


Acquisition function: GP Upper Confidence Bound



- GP-UCB balances explore/exploit: $\alpha_{\text{UCB}}(x|\beta, \mathcal{D}) = \mu(x|\mathcal{D}) - \beta \sigma(x|\mathcal{D})$
- Theoretical guarantees available! Optimal choices for β “regularisation parameter”

Acquisition function: Expected Improvement



- EI perhaps most common: $\alpha_{\text{EI}}(x|\mathcal{D}) = \int \min\{0, s - s_{\min}\} p(s | x, \mathcal{D}) ds(x)$
- Closed form for Gaussian posteriors but can be too “greedy” in some problems

Acquisition function: Expected Improvement (Derivation for interest)

$$s^* := \min_{n \in [1, N]} s(x_n) \text{ (i.e. the smallest value seen so far in } \mathcal{D} \text{)}$$

$$\text{EI}(x) := \mathbb{E}[\min\{s^*, s(x)\}]$$

$$= s^* + \int_{-\infty}^{\infty} \min\{0, s(x) - s^*\} p(s(x) \mid \mathcal{D}) ds(x)$$

$$= s^* + \int_{s^*}^{\infty} 0 \cdot p(s(x) \mid \mathcal{D}) ds(x) + \int_{-\infty}^{s^*} (s(x) - s^*) p(s(x) \mid \mathcal{D}) ds(x)$$

$$= s^* + \int_{-\infty}^{s^*} (s(x) - s^*) p(s(x) \mid \mathcal{D}) ds(x)$$

$$= -\sigma^2(x) \mathcal{N}(s^* \mid \mu(x), \sigma^2(x)) + (\mu(x) - s^*) \Phi(s^* \mid \mu(x), \sigma^2(x))$$

- $\Phi(\cdot \mid \mu, \sigma^2)$ is the Normal cumulative distribution function
- Dropped dependency on $\mathcal{D}$ in $s(\cdot)$, $\mu(\cdot)$ and $\sigma(\cdot)$ for clarity

Other acquisition functions

- There are many different acquisition functions from different motivations, e.g.
- (Predictive) Entropy Search:

$$\alpha_{\text{ES}}(x|\mathcal{D}) = \mathbb{H}[p(x_{\min} | \mathcal{D})] - \mathbb{E}_{p(s|x,\mathcal{D})} \left[\mathbb{H}[p(x_{\min} | \mathcal{D} \cup \{x, s\})] \right]$$

- Information theoretic approach to reduce the entropy of $p(x_{\min})$
- Note: Approximating $p(x_{\min})$ is not trivial
- Thompson Sampling:
$$\alpha_{\text{ES}}(x|\mathcal{D}) = \hat{s}(x), \text{ where } \hat{s}(x) \xleftarrow{\text{sample}} s(\mathbf{x} | \mathcal{D}) := \mathcal{N}(\mu_{\theta}(\mathbf{x} | \mathcal{D}), \sigma_{\theta}^2(\mathbf{x} | \mathcal{D}))$$
- Stochastic acquisition function (sample from GP posterior)
- Used in PES to compute $p(x_{\min})$

The Bayesian optimisation algorithm

- Choose a **prior** for the objective $f(\cdot)$ and collect some initial **data** (also need the domain range!)

The Bayesian optimisation algorithm

- Choose a **prior** for the objective $f(\cdot)$ and collect some initial **data** (also need the domain range!)
- While you still have time/budget:

The Bayesian optimisation algorithm

- Choose a **prior** for the objective $f(\cdot)$ and collect some initial **data** (also need the domain range!)
- While you still have time/budget:
 1. Combine the prior and available data to find the **posterior**

The Bayesian optimisation algorithm

- Choose a **prior** for the objective $f(\cdot)$ and collect some initial **data** (also need the domain range!)
- While you still have time/budget:
 1. Combine the prior and available data to find the **posterior**
 2. Use the posterior to construct the **acquisition function**

The Bayesian optimisation algorithm

- Choose a **prior** for the objective $f(\cdot)$ and collect some initial **data** (also need the domain range!)
- While you still have time/budget:
 1. Combine the prior and available data to find the **posterior**
 2. Use the posterior to construct the **acquisition function**
 3. Find the best **next query** from the acquisition function

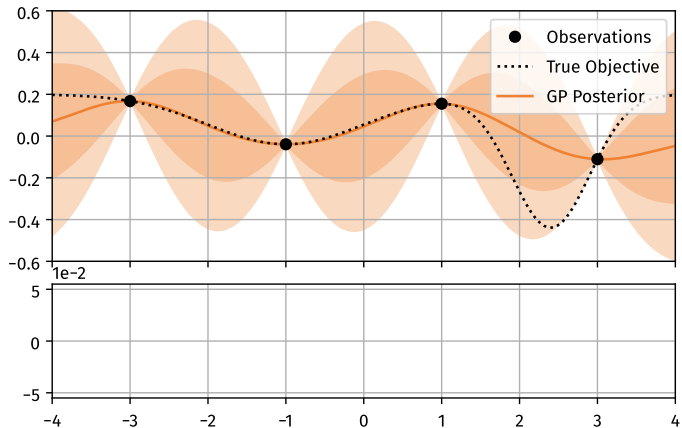
The Bayesian optimisation algorithm

- Choose a **prior** for the objective $f(\cdot)$ and collect some initial **data** (also need the domain range!)
- While you still have time/budget:
 1. Combine the prior and available data to find the **posterior**
 2. Use the posterior to construct the **acquisition function**
 3. Find the best **next query** from the acquisition function
 4. Evaluate the objective at the query and **extend the dataset**

The Bayesian optimisation algorithm

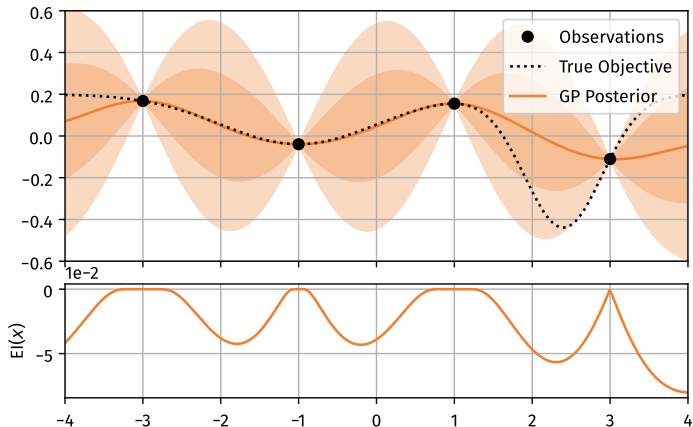
- Choose a **prior** for the objective $f(\cdot)$ and collect some initial **data** (also need the domain range!)
- While you still have time/budget:
 1. Combine the prior and available data to find the **posterior**
 2. Use the posterior to construct the **acquisition function**
 3. Find the best **next query** from the acquisition function
 4. Evaluate the objective at the query and **extend the dataset**
- Report the best objective sample found!

Bayesian optimisation in action!



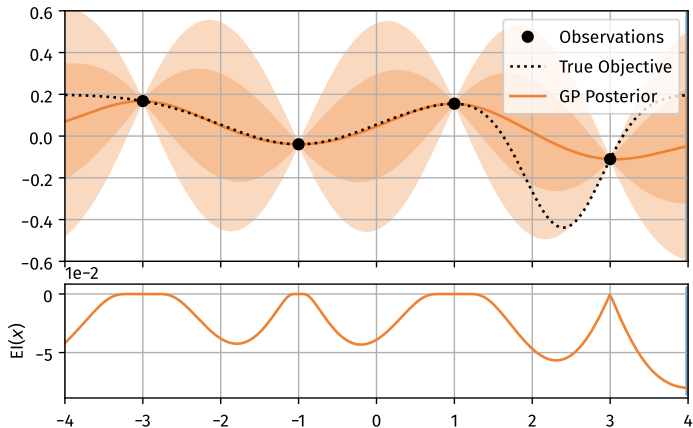
1. Combine the prior and available data to find the **posterior**

Bayesian optimisation in action!



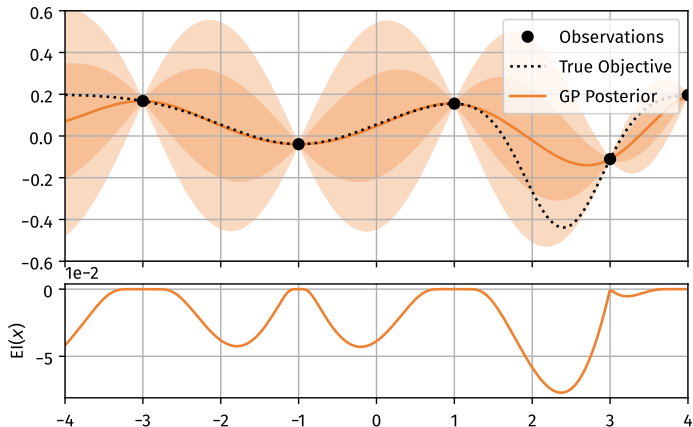
2. Use the posterior to construct the **acquisition function**

Bayesian optimisation in action!



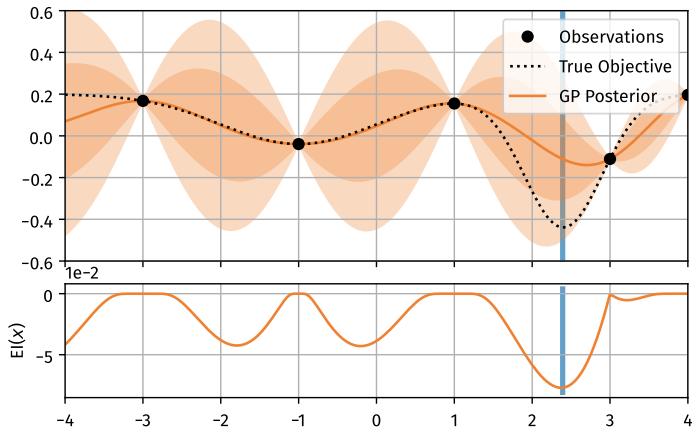
3. Find the best **next query** from the acquisition function

Bayesian optimisation in action!



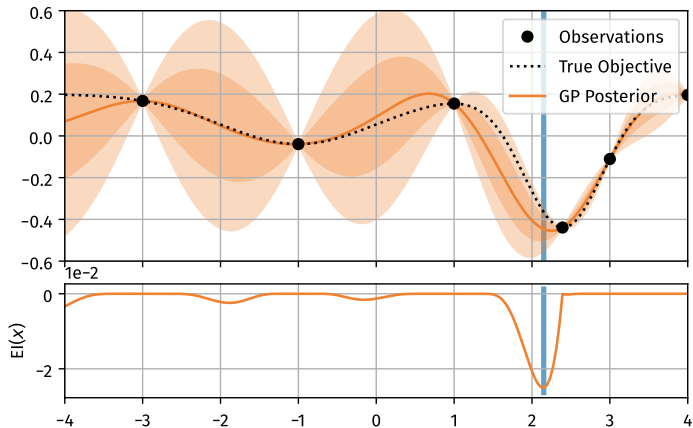
4. Evaluate the objective at the query and **extend the dataset**

Bayesian optimisation in action!



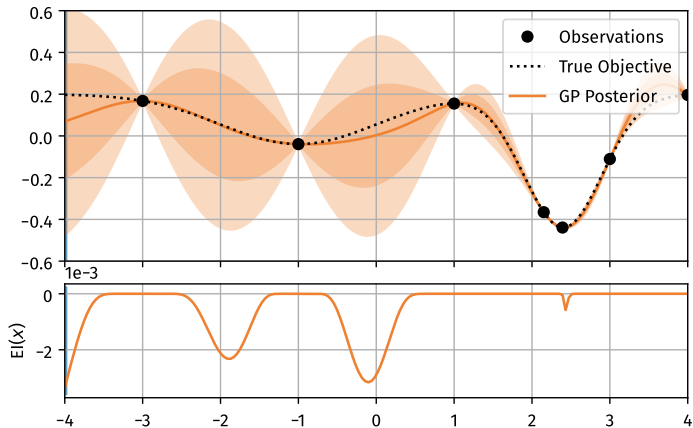
- Repeat!!

Bayesian optimisation in action!



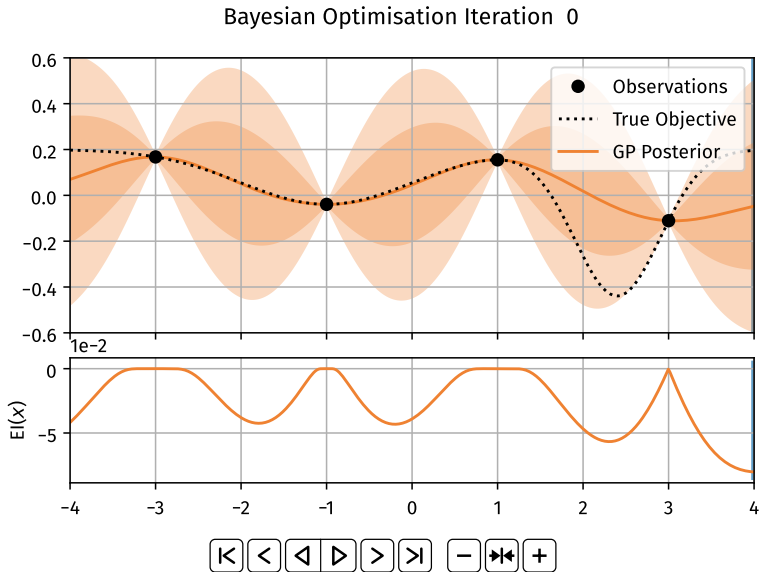
- Repeat!!

Bayesian optimisation in action!

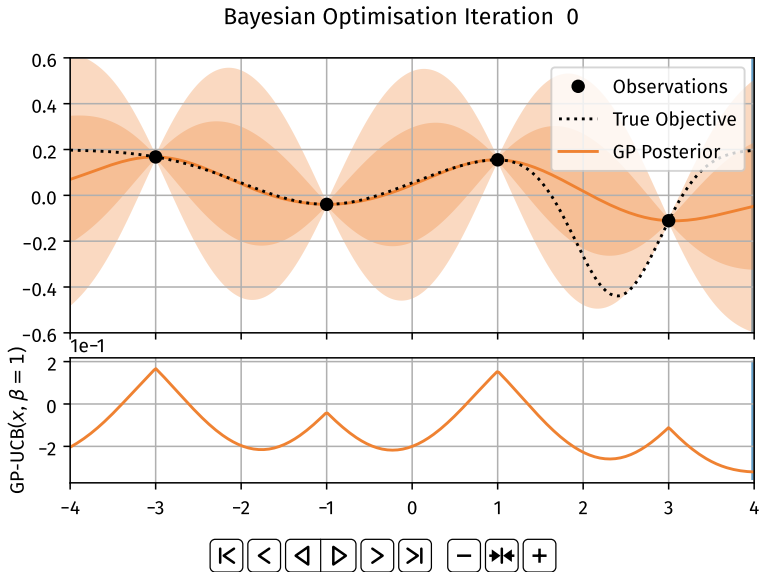


- Repeat!!

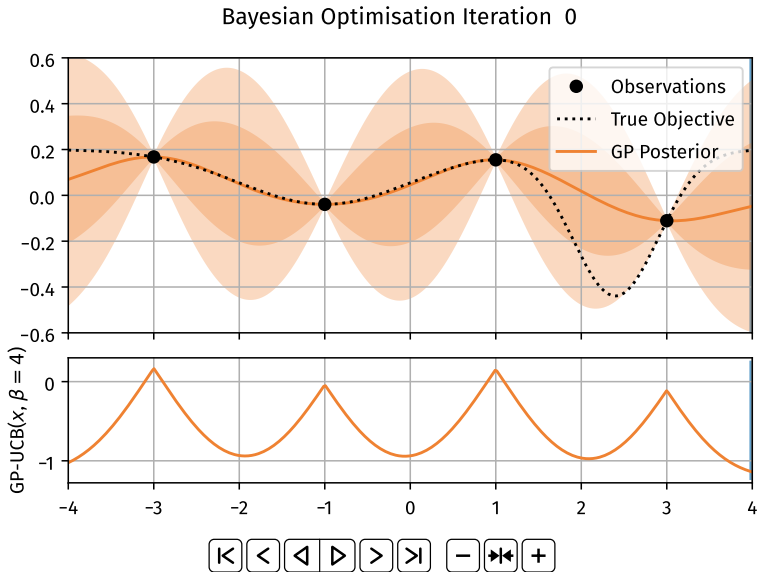
Bayesian optimisation! Acquisition function: EI



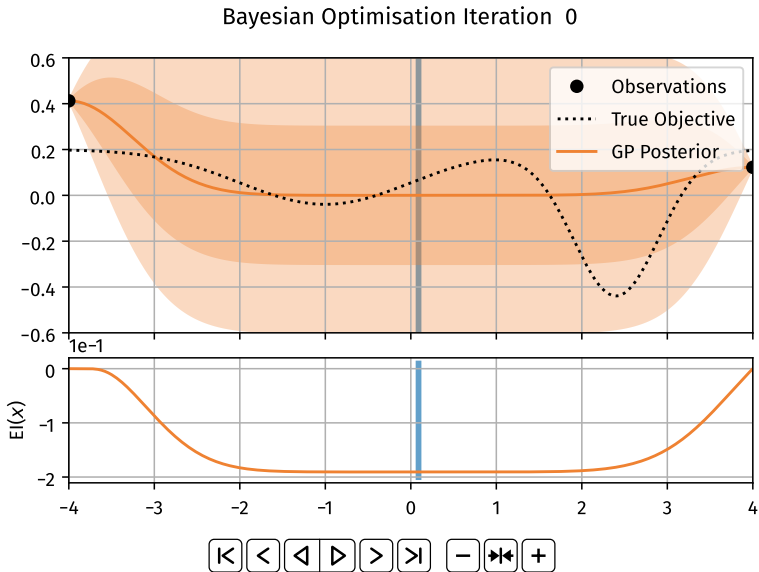
Bayesian optimisation! Acquisition function: UCB



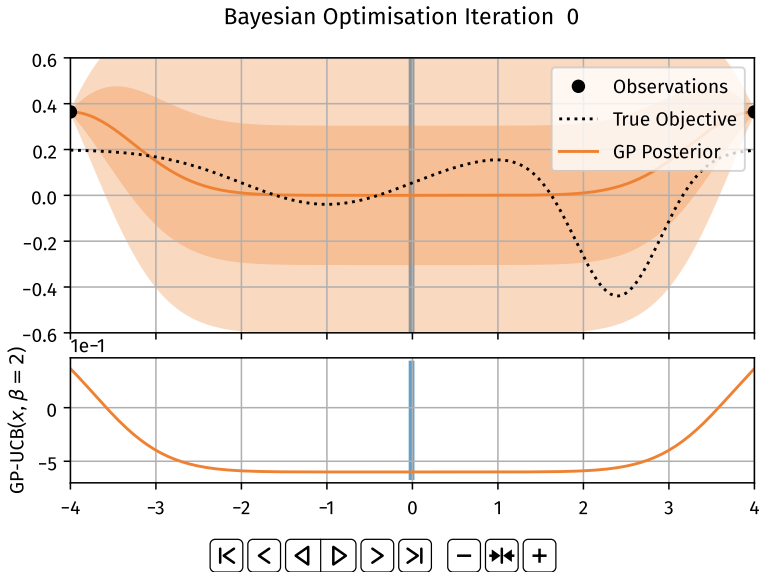
Bayesian optimisation! Acquisition function: UCB (greater exploration, $\beta \uparrow$)



Bayesian optimisation! Acquisition function: EI (Ackley's Function)



Bayesian optimisation! Acquisition function: UCB (Ackley's Function)



Bayesian Optimisation Summary

Transformed the original **unsolvable** problem:

$$\mathbf{x}^* = \arg \min_{\mathbf{x} \in \mathcal{X}} f(\mathbf{x})$$

Bayesian Optimisation Summary

Transformed the original **unsolvable** problem:

$$\mathbf{x}^* = \arg \min_{\mathbf{x} \in \mathcal{X}} f(\mathbf{x})$$

into a series of **solvable** problems:

$$x_{n+1} = \arg \min_{\mathbf{x} \in \mathcal{X}} \alpha(\mathbf{x} \mid \mathcal{D}_n, s(x \mid \mathcal{D}_n))$$

Bayesian Optimisation Summary

Transformed the original **unsolvable** problem:

$$\mathbf{x}^* = \arg \min_{\mathbf{x} \in \mathcal{X}} f(\mathbf{x})$$

into a series of **solvable** problems:

$$x_{n+1} = \arg \min_{\mathbf{x} \in \mathcal{X}} \alpha(\mathbf{x} \mid \mathcal{D}_n, s(x \mid \mathcal{D}_n))$$

where:

- $\alpha(\cdot)$ is inexpensive to evaluate
- Gradients of $\alpha(\cdot)$ are available (faster optimisation methods)
- Issue is now finding sensible modelling and heuristics for the surrogate and $\alpha(\cdot)$

- Make sure your priors are sensible!
 - Is the covariance function an appropriate model?
 - Amplitude scale is right
 - Lengthscale is appropriate
 - Are the observations noisy? (might need noise model)
- Should I reestimate my hyperparams at each iteration?
- Data normalised
- Can I guarantee to optimise the acquisition function appropriately
- Computational complexity of GP $\mathcal{O}(N^3)$ (usually not an issue vs objective!)

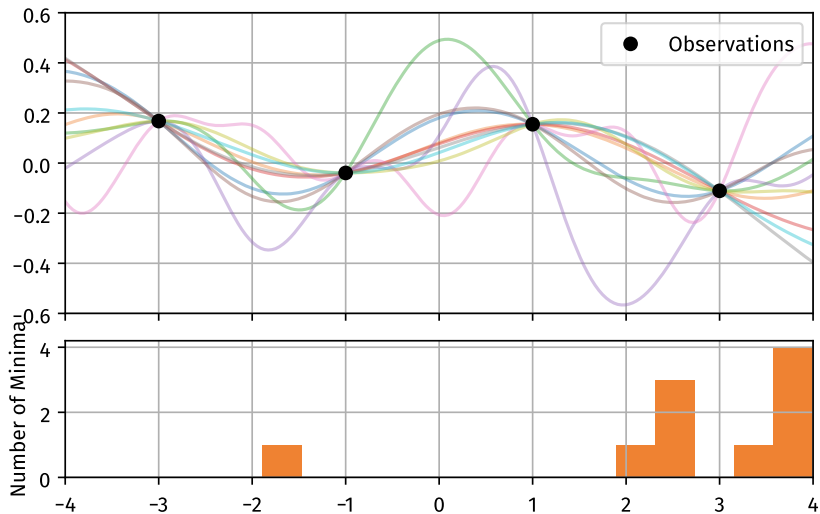
Extensions..

- **Batched Bayes Opt:** Ask for multiple new observations in parallel at each stage
- **Multi-Step Look-Ahead:** (Related) try to find a group of queries that are optimal rather than a single optimal query
- Clearly these ideas are tied into sequential decision making processes and hence **Reinforcement Learning..**

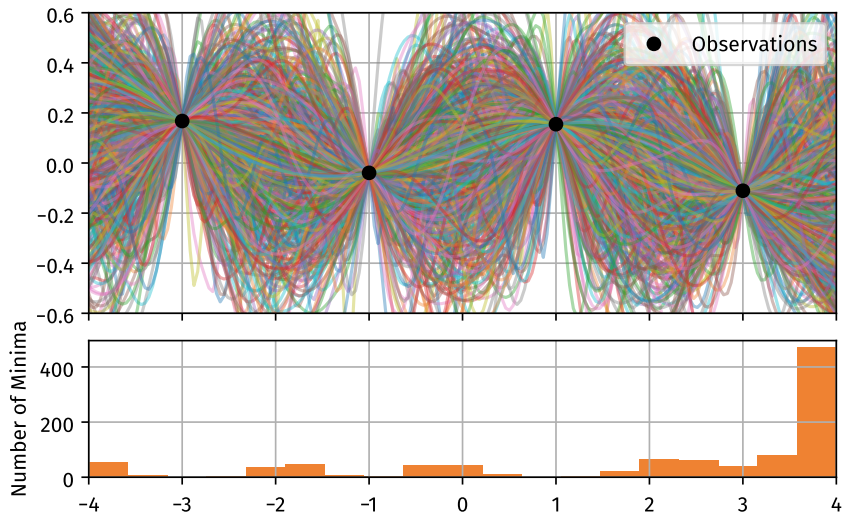
Further reading:

- Gaussian Process Summer School has video lectures and practicals on “Bayes Opt”:
<https://gpss.cc>
- Probabilistic Numerics Book from the start of the lecture has comprehensive overview of Bayesian optimisation

Varying the lengthscale as well..



Varying the lengthscale as well..



Evaluation

Overview...

Motivation

No Free Lunch

Whirlwind Introduction to Inverse Probabilities

Model Selection

Gaussian Processes (a Prior over Functions)

Application: Bayesian Optimisation

Evaluation

Conclusions

Evaluation in Machine Learning

We have made a cool new model with great advances..

- How do we evaluate empirical performance?
- What do we care about?

Evaluation in Machine Learning

We have made a cool new model with great advances..

- How do we evaluate empirical performance?
- What do we care about?

We probably want:

- Fair comparisons (comparing methods)
- Useful comparisons (what can we learn)
- Understand limitations (how confident should we be)

Evaluation in Machine Learning

We have made a cool new model with great advances..

- How do we evaluate empirical performance?
- What do we care about?

We probably want:

- Fair comparisons (comparing methods)
- Useful comparisons (what can we learn)
- Understand limitations (how confident should we be)
- **How can we make it better if we don't know where it fails?!**

Illustrative example: Which method is best?

Model name	RMSE Mean ↓
MD2 Boot+Log	3.850
MD2 Boot+Self	3.795
Diagonal	4.000
SUPN Boot+Log	4.071
SUPN Boot+Self	4.091

Illustrative example: Which method is best?

Model name	RMSE Mean ↓
MD2 Boot+Log	3.850
MD2 Boot+Self	3.795
Diagonal	4.000
SUPN Boot+Log	4.071
SUPN Boot+Self	4.091

- We are showing the Root of the Mean Squared Error
- We want lower scores so the MD2 model is the best?

Illustrative example: Which method is best?

Model name	RMSE Mean ↓
MD2 Boot+Log	3.850 (1.370)
MD2 Boot+Self	3.795 (1.397)
Diagonal	4.000 (1.457)
SUPN Boot+Log	4.071 (1.489)
SUPN Boot+Self	4.091 (1.442)

Illustrative example: Which method is best?

Model name	RMSE Mean ↓
MD2 Boot+Log	3.850 (1.370)
MD2 Boot+Self	3.795 (1.397)
Diagonal	4.000 (1.457)
SUPN Boot+Log	4.071 (1.489)
SUPN Boot+Self	4.091 (1.442)

- We now have standard errors (i.e. the standard deviation)
- Does this change things?

Illustrative example: Which method is best?

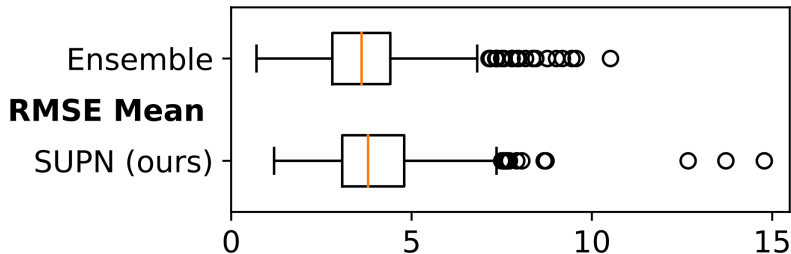


Figure 2. Box plot illustrating the strong distribution overlap between the original ensemble and the trained SUPN model for Boot+Log RMSE mean.

Illustrative example: Which method is best?

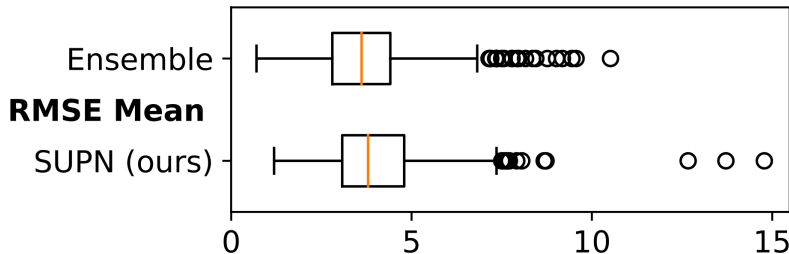


Figure 2. Box plot illustrating the strong distribution overlap between the original ensemble and the trained SUPN model for Boot+Log RMSE mean.

- Looking at the **distributions** reveals very similar performance
- The SUPN mean is skewed by only 3 outliers!

Important slide acknowledgements!

Important slide acknowledgements!

Illustrations taken from the excellent new text book from Simon Prince:

Understanding Deep Learning, Simon J.D. Prince, MIT Press

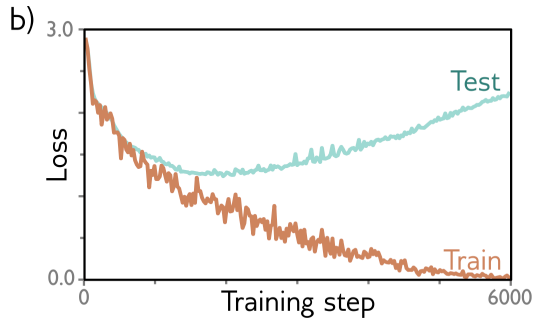
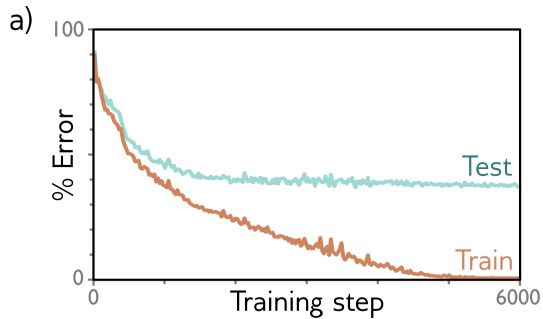
Final draft available on the website:

<https://udlbook.github.io/udlbook/>

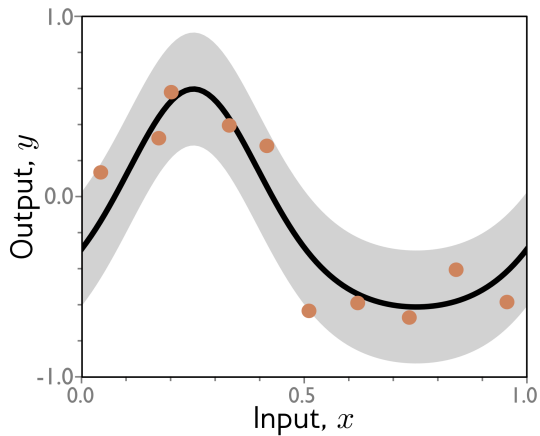
I strongly recommend you take a look!

Evaluation: Training vs Test

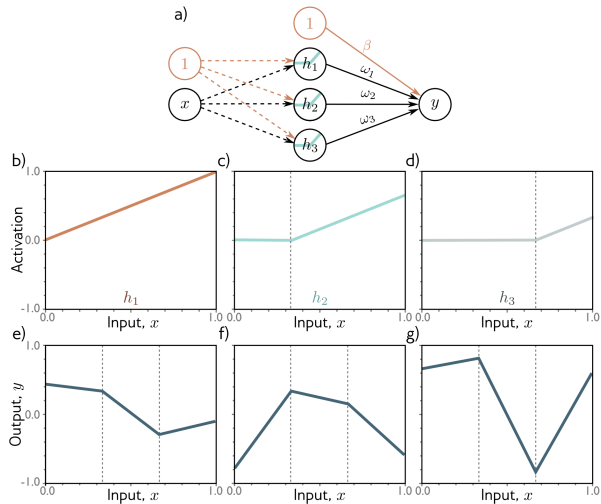
Evaluation: Training vs Test



Evaluation: Regression example



Evaluation: Toy regression model



Evaluation: Noise, Bias and Variance

$$y_n = f(x_n; \phi) + \varepsilon_n, \quad n = 1 \dots N, \quad \varepsilon_n \sim \mathcal{N}(0, \sigma^2)$$

Evaluation: Noise, Bias and Variance

$$y_n = f(x_n; \phi) + \varepsilon_n, \quad n = 1 \dots N, \quad \varepsilon_n \sim \mathcal{N}(0, \sigma^2)$$

$$y(x) = f(x; \phi) + \varepsilon$$

$$\mu(x) = \mathbb{E}_y[y(x)] = \int y(x) p(y \mid x) dy$$

Evaluation: Noise, Bias and Variance

$$y_n = f(x_n; \phi) + \varepsilon_n, \quad n = 1 \dots N, \quad \varepsilon_n \sim \mathcal{N}(0, \sigma^2)$$

$$y(x) = f(x; \phi) + \varepsilon$$

$$\mu(x) = \mathbb{E}_y[y(x)] = \int y(x) p(y | x) dy$$

Now consider a least squares (L2) loss function:

$$\begin{aligned} \mathcal{L}(x, \phi) &= (f(x; \phi) - y(x))^2 \\ &= ((f(x; \phi) - \mu(x)) + (\mu(x) - y(x)))^2 \\ &= (f(x; \phi) - \mu(x))^2 + 2(f(x; \phi) - \mu(x)) (\mu(x) - y(x)) + (\mu(x) - y(x))^2 \end{aligned}$$

Evaluation: Noise, Bias and Variance

$$y_n = f(x_n; \phi) + \varepsilon_n, \quad n = 1 \dots N, \quad \varepsilon_n \sim \mathcal{N}(0, \sigma^2)$$

$$y(x) = f(x; \phi) + \varepsilon$$

$$\mu(x) = \mathbb{E}_y[y(x)] = \int y(x) p(y | x) dy$$

Now consider a least squares (L2) loss function:

$$\begin{aligned} \mathcal{L}(x, \phi) &= (f(x; \phi) - y(x))^2 \\ &= ((f(x; \phi) - \mu(x)) + (\mu(x) - y(x)))^2 \\ &= (f(x; \phi) - \mu(x))^2 + 2(f(x; \phi) - \mu(x)) (\mu(x) - y(x)) + (\mu(x) - y(x))^2 \\ &\Rightarrow \mathbb{E}_y[\mathcal{L}(x, \phi)] = (f(x; \phi) - \mu(x))^2 + \sigma^2 \end{aligned}$$

Evaluation: Noise, Bias and Variance

$$\mathbb{E}_y[\mathcal{L}(x, \phi)] = (f(x; \phi) - \mu(x))^2 + \underbrace{\sigma^2}_{\text{noise}}$$

We have partitioned the expected loss into two terms, the second is some **irreducible noise** that comes with the observations (e.g. sensor noise).

Evaluation: Noise, Bias and Variance

$$\mathbb{E}_y[\mathcal{L}(x, \phi)] = (f(x; \phi) - \mu(x))^2 + \underbrace{\sigma^2}_{\text{noise}}$$

We have partitioned the expected loss into two terms, the second is some **irreducible noise** that comes with the observations (e.g. sensor noise).

So far we have ignored the fact that we actually estimate parameters from a **sampled dataset** $\mathcal{D} = \{x_n, y_n\}_{n=1}^N$

$$\begin{aligned} f_\mu(x) &:= \mathbb{E}_{\mathcal{D}}[f(x; \phi(\mathcal{D}))] \\ \Rightarrow (f(x; \phi) - \mu(x))^2 &= ((f(x; \phi_{\mathcal{D}}) - f_\mu(x)) + (f_\mu(x) - \mu(x)))^2 \\ \Rightarrow \mathbb{E}_{\mathcal{D}}[(f(x; \phi) - \mu(x))^2] &= \mathbb{E}_{\mathcal{D}}[(f(x; \phi_{\mathcal{D}}) - f_\mu(x))^2] + (f_\mu(x) - \mu(x))^2 \end{aligned}$$

Evaluation: Noise, Bias and Variance

Therefore, if we take expectations over datasets, our expected loss comprises three terms:

$$\mathbb{E}_{\mathcal{D}}[\mathbb{E}_y[\mathcal{L}(x, \phi)]] = \underbrace{\mathbb{E}_{\mathcal{D}}[(f(x; \phi_{\mathcal{D}}) - f_{\mu}(x))^2]}_{\text{variance}} + \underbrace{(f_{\mu}(x) - \mu(x))^2}_{\text{bias}} + \underbrace{\sigma^2}_{\text{noise}}$$

Note: more complex for models other than least squares...

Evaluation: Noise, Bias and Variance

Therefore, if we take expectations over datasets, our expected loss comprises three terms:

$$\mathbb{E}_{\mathcal{D}}[\mathbb{E}_y[\mathcal{L}(x, \phi)]] = \underbrace{\mathbb{E}_{\mathcal{D}}[(f(x; \phi_{\mathcal{D}}) - f_{\mu}(x))^2]}_{\text{variance}} + \underbrace{(f_{\mu}(x) - \mu(x))^2}_{\text{bias}} + \underbrace{\sigma^2}_{\text{noise}}$$

Note: more complex for models other than least squares...

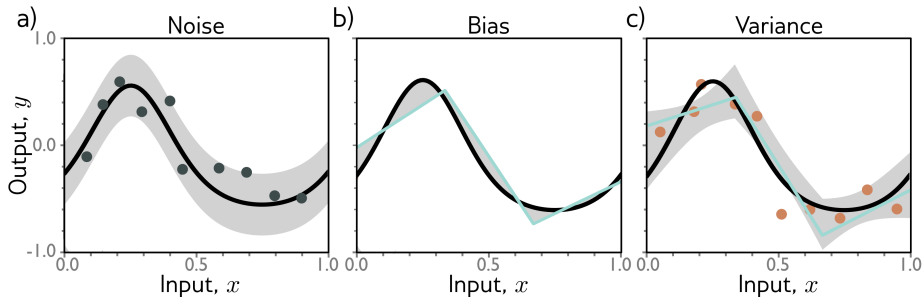
Noise Error in measurements (e.g. sensor noise, missing data, data mislabelled, ...)

Bias Systematic deviation from the true mean of the function (e.g. due to limitations of our model, ...)

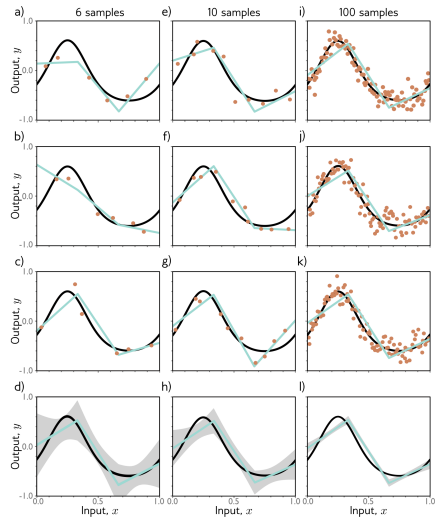
Variance Uncertainty in the fitting of our model due to limitations of the dataset (e.g. too few samples, dataset doesn't span the distribution, ...)

Evaluation: Noise, Bias and Variance

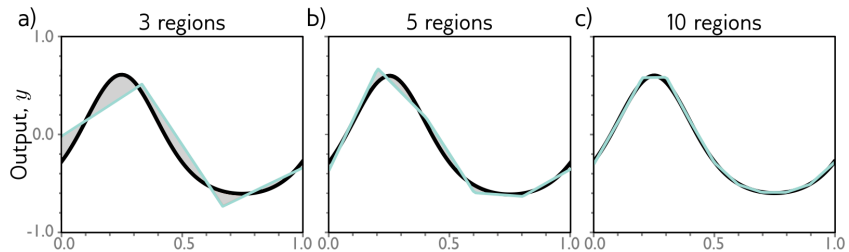
$$\mathbb{E}_{\mathcal{D}}[\mathbb{E}_y[\mathcal{L}(x, \phi)]] = \underbrace{\mathbb{E}_{\mathcal{D}}\left[\left(f(x; \phi_{\mathcal{D}}) - \overbrace{f_{\mu}(x)}^{\text{best possible model from infinite data}}\right)^2\right]}_{\text{variance}} + \underbrace{\left(f_{\mu}(x) - \overbrace{\mu(x)}^{\text{true function}}\right)^2}_{\text{bias}} + \underbrace{\sigma^2}_{\text{noise}}$$



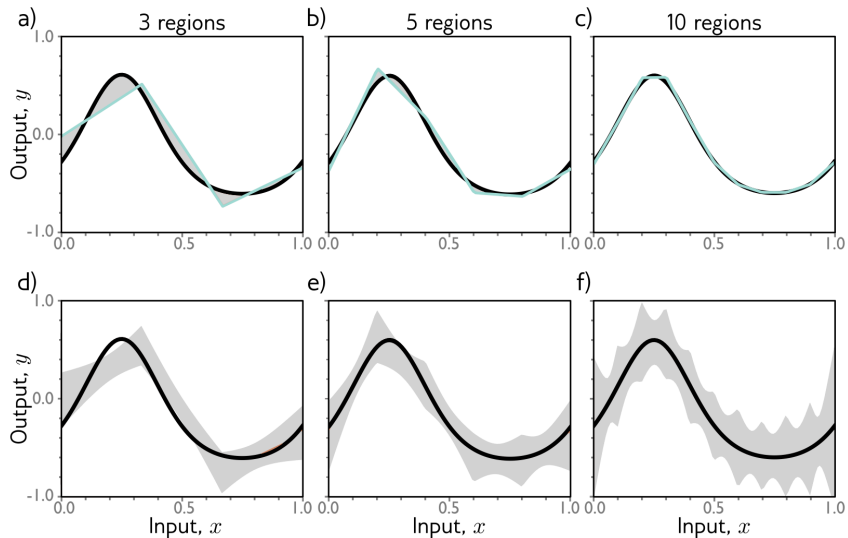
Evaluation: Variance reduction



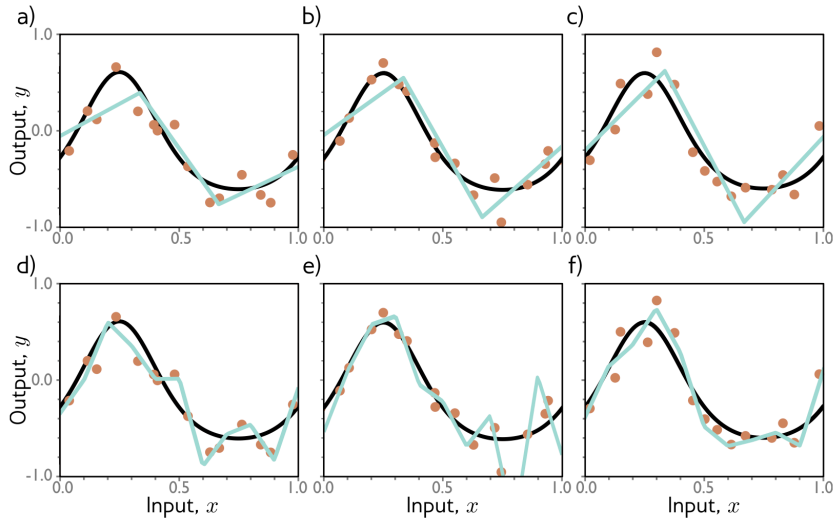
Evaluation: Bias reduction



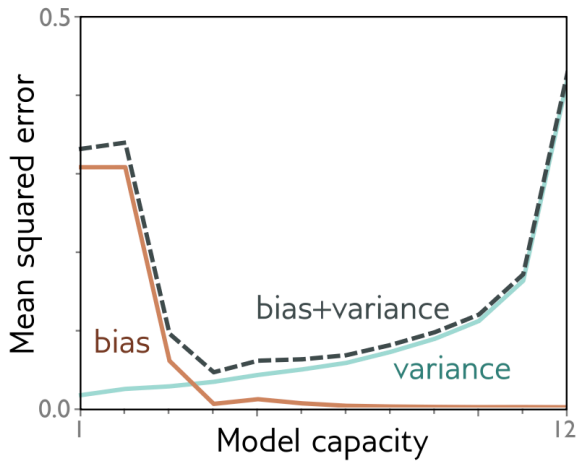
Evaluation: Bias reduction



Evaluation: Overfitting



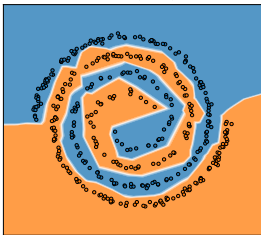
Evaluation: Bias-Variance tradeoff



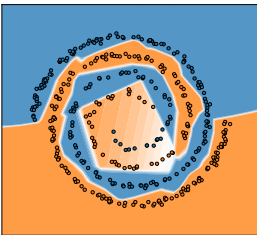
Recall: Spiral Dataset with an MLP.

Same data, varying MLP init, MLP: 32 x 2

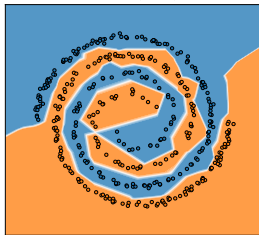
init_seed=0 (acc=1.00)



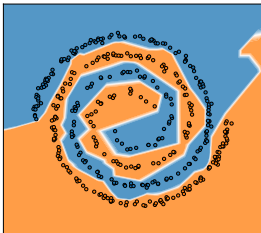
init_seed=1 (acc=0.96)



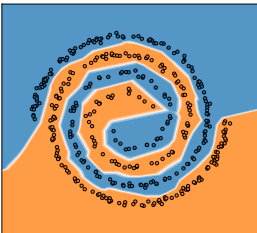
init_seed=2 (acc=1.00)



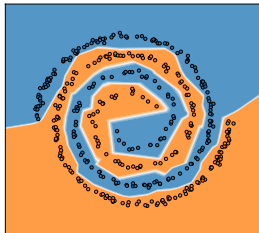
init_seed=3 (acc=1.00)



init_seed=4 (acc=1.00)



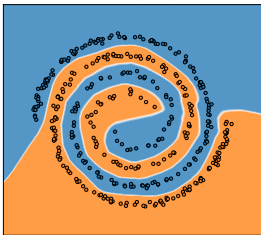
init_seed=5 (acc=1.00)



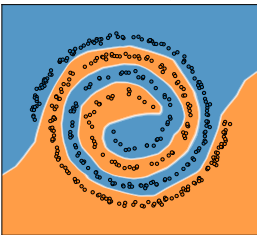
Recall: Spiral Dataset with an MLP.

Same data, varying MLP init, MLP: 64 x 4

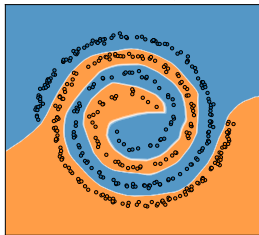
init_seed=0 (acc=1.00)



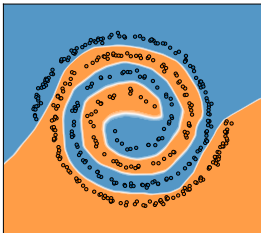
init_seed=1 (acc=1.00)



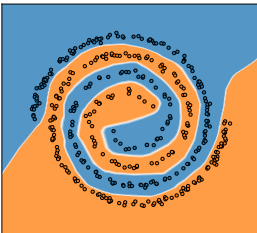
init_seed=2 (acc=1.00)



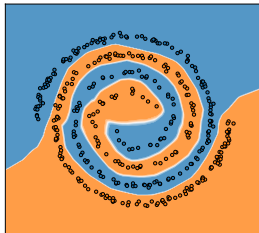
init_seed=3 (acc=1.00)



init_seed=4 (acc=1.00)



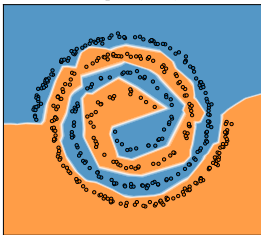
init_seed=5 (acc=1.00)



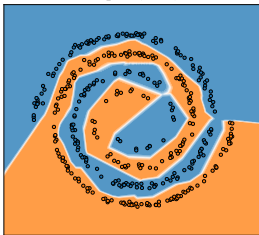
Recall: Spiral Dataset with an MLP: Different datasets..

Varying dataset, fixed MLP initialization

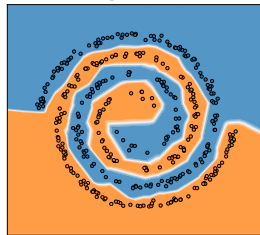
data_seed=0 (acc=1.00)



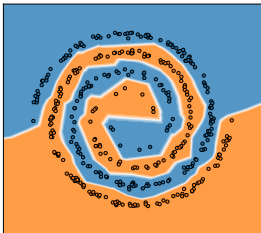
data_seed=1 (acc=1.00)



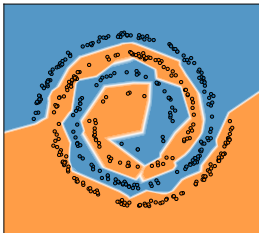
data_seed=2 (acc=1.00)



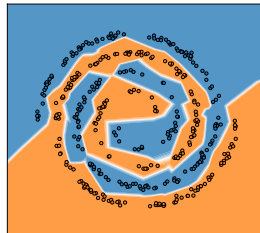
data_seed=3 (acc=1.00)



data_seed=4 (acc=1.00)

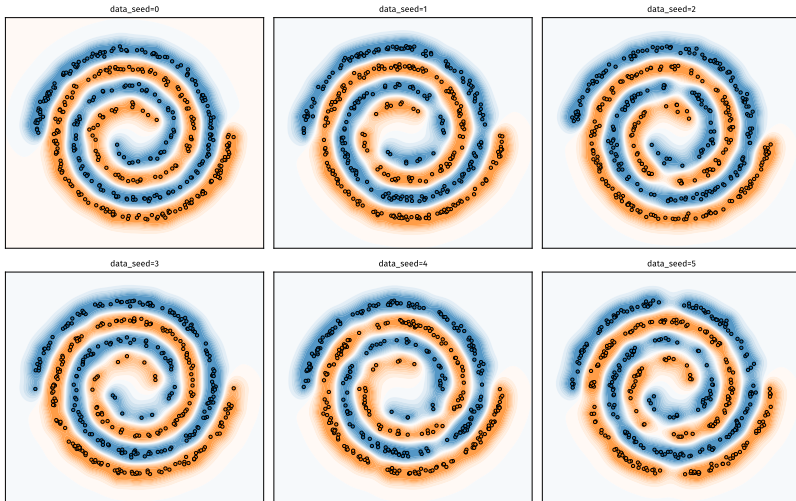


data_seed=5 (acc=1.00)



Recall Spiral Dataset with a Bayesian (GP) model: Different datasets..

Varying dataset, GP classifier (RBF kernel, Bernoulli likelihood)



How does this relate to the Bayesian Objective? {[Fong and Holmes 2020]}

Consider using the evidence (marginal log-likelihood) as an objective:

$$\mathcal{D} = \{\mathbf{x}_1, \dots, \mathbf{x}_n\}, \quad p(\mathcal{D}) = p(\mathbf{x}_1, \dots, \mathbf{x}_n) = \int \underbrace{\prod_{n=1}^N p(\mathbf{x}_n \mid \mathbf{w})}_{\text{likelihood}} \overbrace{p(\mathbf{w})}^{\text{prior}} d\mathbf{w}$$

How does this relate to the Bayesian Objective? {[Fong and Holmes 2020]}

Consider using the evidence (marginal log-likelihood) as an objective:

$$\mathcal{D} = \{\mathbf{x}_1, \dots, \mathbf{x}_n\}, \quad p(\mathcal{D}) = p(\mathbf{x}_1, \dots, \mathbf{x}_n) = \int \underbrace{\prod_{n=1}^N p(\mathbf{x}_n \mid \mathbf{w})}_{\text{likelihood}} \overbrace{p(\mathbf{w})}^{\text{prior}} d\mathbf{w}$$

For any permutation π of $\{1, \dots, N\}$

$$p(\mathcal{D}) = \prod_{k=1}^N p(\mathbf{x}_{\pi(k)} \mid \mathbf{x}_{\pi(1)}, \dots, \mathbf{x}_{\pi(k-1)})$$

Nb. $p(a, b, c) = p(a \mid b, c) p(b \mid c) p(c) = p(c \mid a, b) p(b \mid a) p(a)$ etc..

How does this relate to the Bayesian Objective? {[Fong and Holmes 2020]}

Consider using the evidence (marginal log-likelihood) as an objective:

$$\mathcal{D} = \{\mathbf{x}_1, \dots, \mathbf{x}_n\}, \quad p(\mathcal{D}) = p(\mathbf{x}_1, \dots, \mathbf{x}_n) = \int \underbrace{\prod_{n=1}^N p(\mathbf{x}_n \mid \mathbf{w})}_{\text{likelihood}} \overbrace{p(\mathbf{w})}^{\text{prior}} d\mathbf{w}$$

For any permutation π of $\{1, \dots, N\}$

$$p(\mathcal{D}) = \prod_{k=1}^N \underbrace{p(\mathbf{x}_k \mid \mathbf{x}_{\pi(1)}, \dots, \mathbf{x}_{\pi(k-1)})}_{\text{Bayesian predictive density}}$$

$$\underbrace{p(\mathbf{x}_* \mid \{\mathbf{x}_1, \dots, \mathbf{x}_K\})}_{\text{Prediction on new point}} = \int p(\mathbf{x}_* \mid \mathbf{w}) \underbrace{p(\mathbf{w} \mid \{\mathbf{x}_1, \dots, \mathbf{x}_K\})}_{\text{Posterior over the weights}} d\mathbf{w}$$

How does this relate to the Bayesian Objective? {[Fong and Holmes 2020]}

Consider using the evidence (marginal log-likelihood) as an objective:

$$\mathcal{D} = \{\mathbf{x}_1, \dots, \mathbf{x}_n\}, \quad p(\mathcal{D}) = p(\mathbf{x}_1, \dots, \mathbf{x}_n) = \int \underbrace{\prod_{n=1}^N p(\mathbf{x}_n \mid \mathbf{w})}_{\text{likelihood}} \overbrace{p(\mathbf{w})}^{\text{prior}} d\mathbf{w}$$

For any permutation π of $\{1, \dots, N\}$

$$\log p(\mathcal{D}) = \sum_{k=1}^N \log p(\mathbf{x}_{\pi(k)} \mid \mathbf{x}_{\pi(1)}, \dots, \mathbf{x}_{\pi(k-1)})$$

Average over all the $N!$ permutations (RHS the same since exchangeable!)

$$\log p(\mathcal{D}) = \frac{1}{N!} \sum_{\pi} \sum_{k=1}^N \log p(\mathbf{x}_{\pi(k)} \mid \mathbf{x}_{\pi(1)}, \dots, \mathbf{x}_{\pi(k-1)})$$

How does this relate to the Bayesian Objective? {[Fong and Holmes 2020]}

Consider using the evidence (marginal log-likelihood) as an objective:

$$\mathcal{D} = \{\mathbf{x}_1, \dots, \mathbf{x}_n\}, \quad p(\mathcal{D}) = p(\mathbf{x}_1, \dots, \mathbf{x}_n) = \int \underbrace{\prod_{n=1}^N p(\mathbf{x}_n \mid \mathbf{w})}_{\text{likelihood}} \overbrace{p(\mathbf{w})}^{\text{prior}} d\mathbf{w}$$

Average over all the $N!$ permutations (RHS the same since exchangeable!)

$$\log p(\mathcal{D}) = \frac{1}{N!} \sum_{\pi} \sum_{k=1}^N \log p(\mathbf{x}_{\pi(k)} \mid \mathbf{x}_{\pi(1)}, \dots, \mathbf{x}_{\pi(k-1)}) \quad (1)$$

$$= \sum_{k=1}^N \left[\frac{1}{N!} \sum_{\pi} \log p(\underbrace{\mathbf{x}_{\pi(k)}}_{\text{All remaining } \mathbf{x} \notin \mathcal{S}} \mid \underbrace{\mathbf{x}_{\pi(1)}, \dots, \mathbf{x}_{\pi(k-1)}}_{\text{All subsets } \mathcal{S} \text{ of size } (k-1)}) \right] \quad (2)$$

$$\text{Cross Validation: } \text{CV}(k) := \mathbb{E}_{|\mathcal{S}|=k, i \notin \mathcal{S}} [\log p(\mathbf{x}_i \mid \mathbf{x}_{\mathcal{S}})]$$

How does this relate to the Bayesian Objective? {[Fong and Holmes 2020]}

Consider using the evidence (marginal log-likelihood) as an objective:

$$\mathcal{D} = \{\mathbf{x}_1, \dots, \mathbf{x}_n\}, \quad p(\mathcal{D}) = p(\mathbf{x}_1, \dots, \mathbf{x}_n) = \int \underbrace{\prod_{n=1}^N p(\mathbf{x}_n \mid \mathbf{w})}_{\text{likelihood}} \overbrace{p(\mathbf{w})}^{\text{prior}} d\mathbf{w}$$

$$\Rightarrow \log p(\mathcal{D}) = \sum_{k=1}^N \text{CV}(k)$$

$$\text{Cross Validation: } \text{CV}(k) := \mathbb{E}_{|\mathcal{S}|=k, i \notin \mathcal{S}} [\log p(\mathbf{x}_i \mid \mathbf{x}_{\mathcal{S}})]$$

The Bayesian Evidence is the sum over every possible training-set size of the average train/validate log-predictive density/score (at that size)!

How does this relate to the Bayesian Objective? {[Fong and Holmes 2020]}

Consider using the evidence (marginal log-likelihood) as an objective:

$$\mathcal{D} = \{\mathbf{x}_1, \dots, \mathbf{x}_n\}, \quad p(\mathcal{D}) = p(\mathbf{x}_1, \dots, \mathbf{x}_n) = \int \underbrace{\prod_{n=1}^N p(\mathbf{x}_n \mid \mathbf{w})}_{\text{likelihood}} \overbrace{p(\mathbf{w})}^{\text{prior}} d\mathbf{w}$$

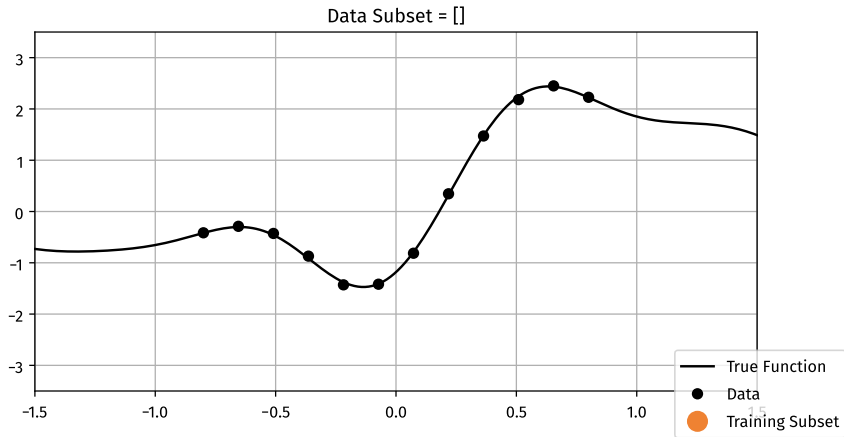
$$\Rightarrow \log p(\mathcal{D}) = \sum_{k=1}^N \text{CV}(n)$$

$$\text{Cross Validation: } \text{CV}(k) := \mathbb{E}_{|\mathcal{S}|=k, i \notin \mathcal{S}} [\log p(\mathbf{x}_i \mid \mathbf{x}_{\mathcal{S}})]$$

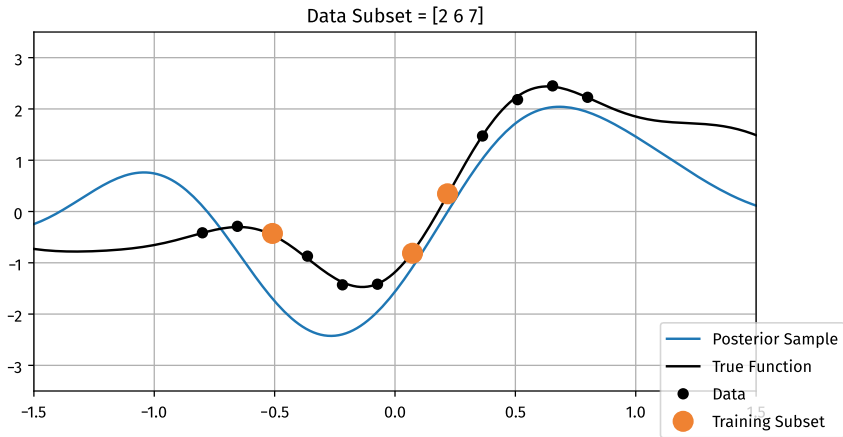
The Bayesian Evidence is the sum over every possible training-set size of the average train/validate log-predictive density/score (at that size)!

Note: Weighting of the subsets is not uniform and there is a block version, see [Fong and Holmes 2020]

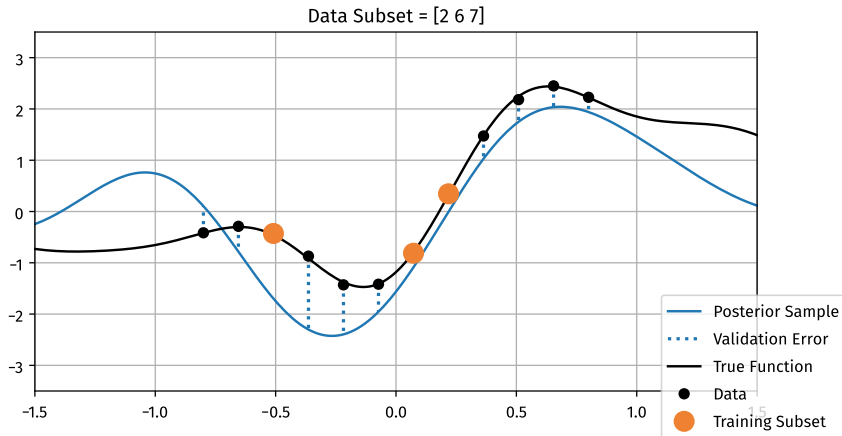
Bayesian Objective $\leftrightarrow$ Cross-Validation Illustration



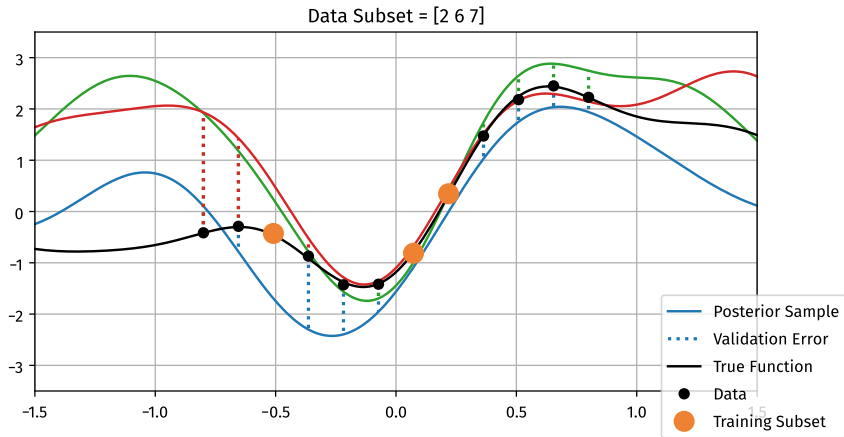
Bayesian Objective $\leftrightarrow$ Cross-Validation Illustration



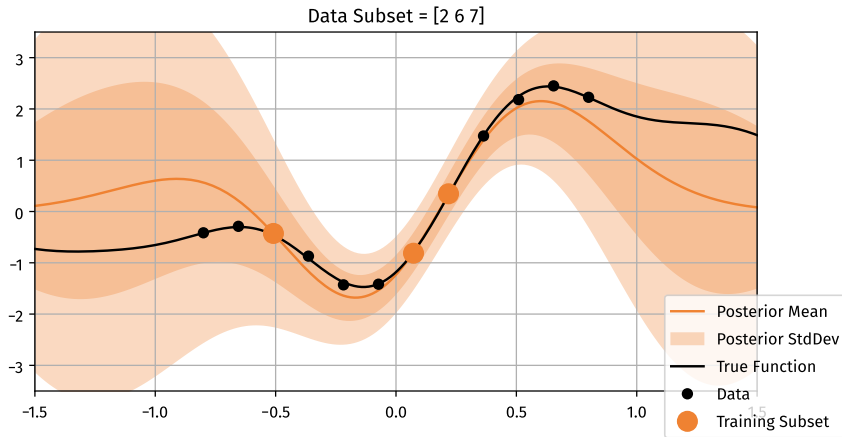
Bayesian Objective $\leftrightarrow$ Cross-Validation Illustration



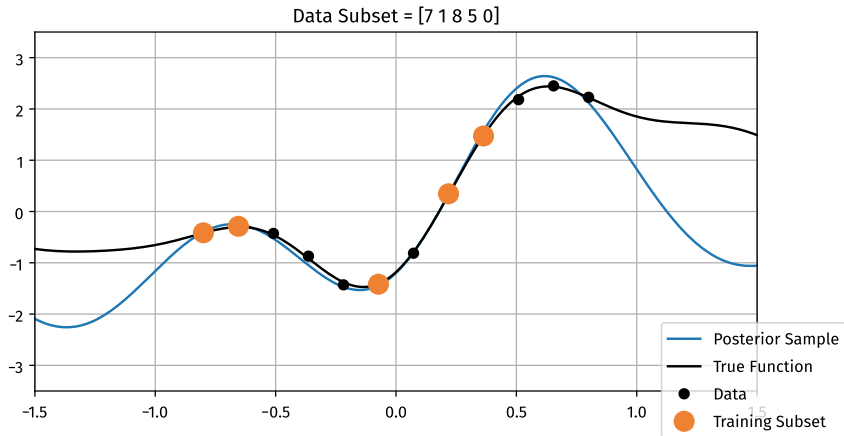
Bayesian Objective $\leftrightarrow$ Cross-Validation Illustration



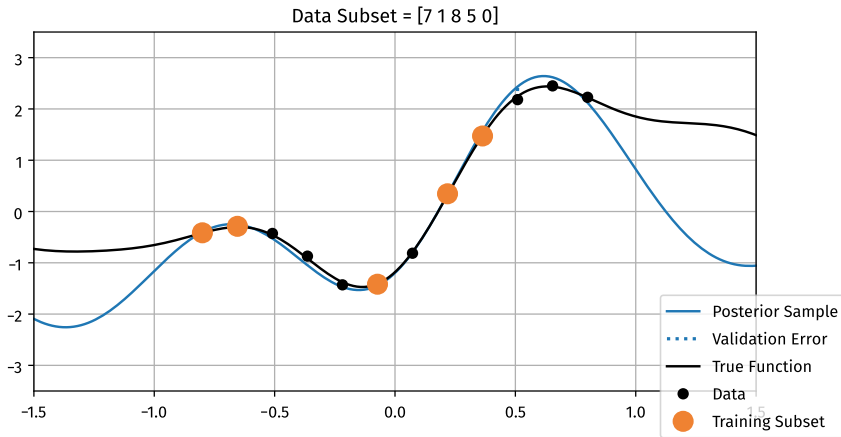
Bayesian Objective $\leftrightarrow$ Cross-Validation Illustration



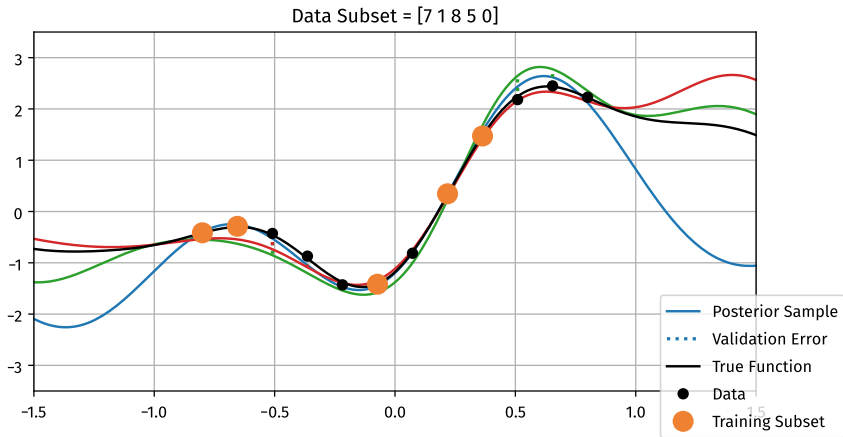
Bayesian Objective $\leftrightarrow$ Cross-Validation Illustration



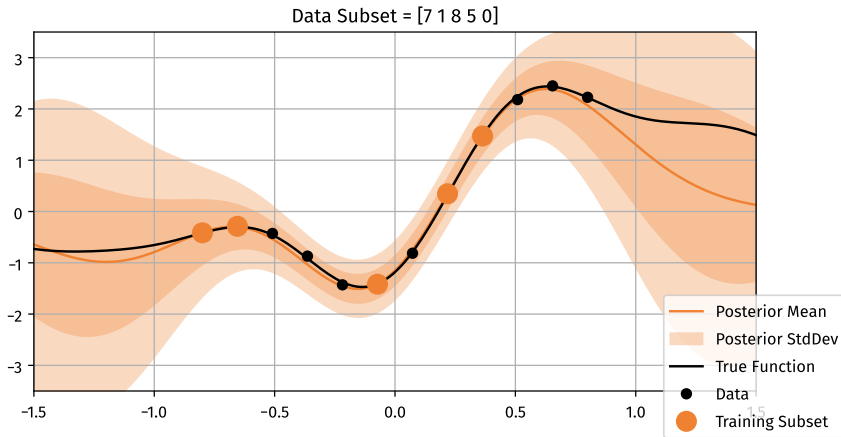
Bayesian Objective $\leftrightarrow$ Cross-Validation Illustration



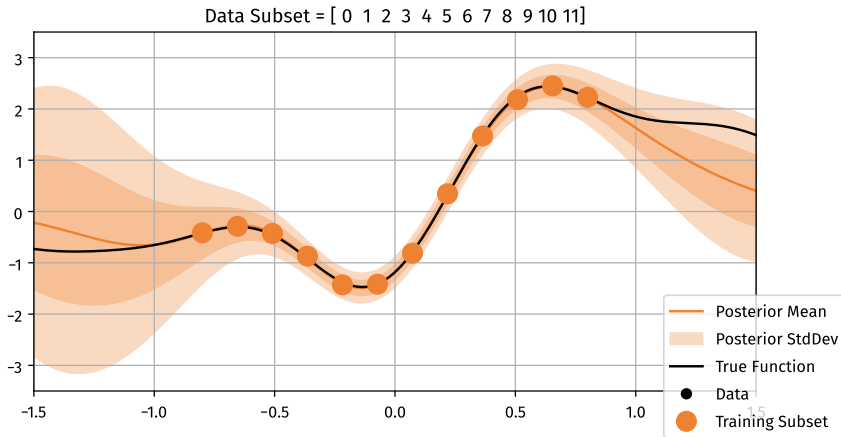
Bayesian Objective $\leftrightarrow$ Cross-Validation Illustration



Bayesian Objective $\leftrightarrow$ Cross-Validation Illustration



Bayesian Objective $\leftrightarrow$ Cross-Validation Illustration



Evaluation: Statistical Significance..

- If you have some statistics training you may be familiar with the concept of **statistical significance**
- At high-level, this concerns the overlapping of (error) distributions and whether you could distinguish reliably between two distributions

Evaluation: Statistical Significance..

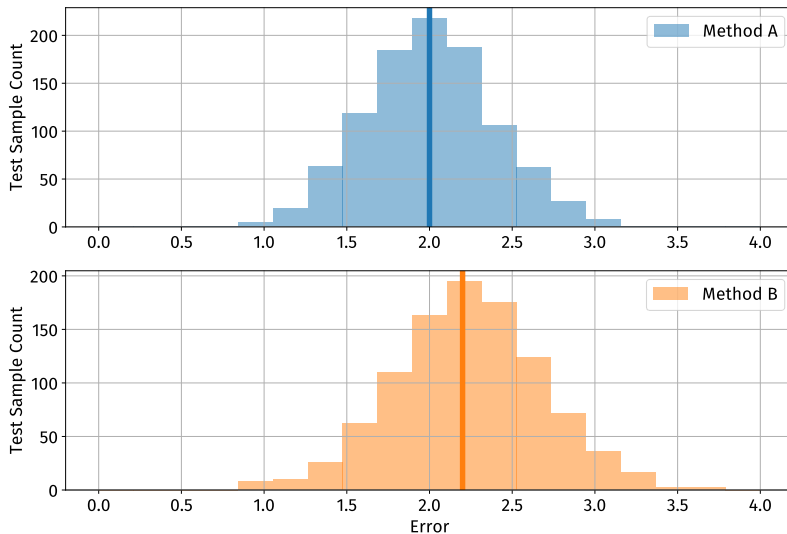
- If you have some statistics training you may be familiar with the concept of **statistical significance**
- At high-level, this concerns the overlapping of (error) distributions and whether you could distinguish reliably between two distributions
- It is possible to conduct formal tests...
- *My Advice:* try to avoid this as it is prone to all kinds of subtle decisions and arguments. **Better to show the raw data in a useful form and people can perform their own assessment...**

Evaluation: Example with statistical tests..

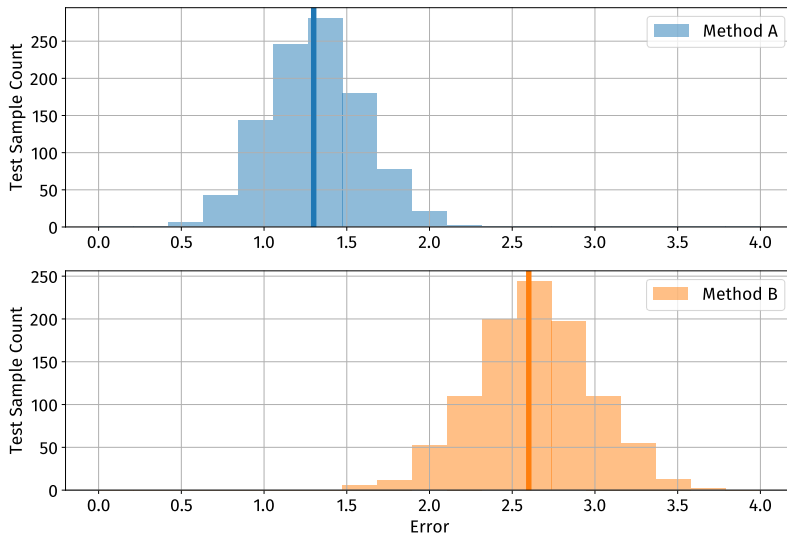
Table 1. Mean gap performance for various test functions; higher is better. The upper table shows the results after 50 objective function evaluations and the lower table after 100 evaluations. Due to computational cost, Warped GP results are only reported for 50 evaluations. Methods not significantly different from the best performing method with respect by a two-sided paired Wilcoxon signed-rank test at a 5% significance level over 20 repetitions are shown in bold (Malkomes & Garnett, 2018). For results in terms of regret, see the supplement.

Benchmark	Evals	Dim	Properties	GP	Warped GP	Homosced GP	Heterosced GP	LGP
Hartmann	50	6	boring	0.959	0.537	0.881	0.973	0.937
Griewank	50	2	oscillatory	0.914	0.493	0.752	0.913	0.897
Shubert	50	2	oscillatory	0.378	0.158	0.378	0.480	0.593
Ackley $[-10, 30]^d$	50	2	complicated, oscillatory	0.924	0.274	0.892	0.912	0.927
Cross In Tray	50	2	complicated, oscillatory	0.954	0.385	0.929	0.977	0.945
Holder table	50	2	complicated, oscillatory	0.939	0.896	0.900	0.931	0.993
Corrupted Holder Table	50	2	complicated, oscillatory	0.741	0.798	0.826	0.729	0.896

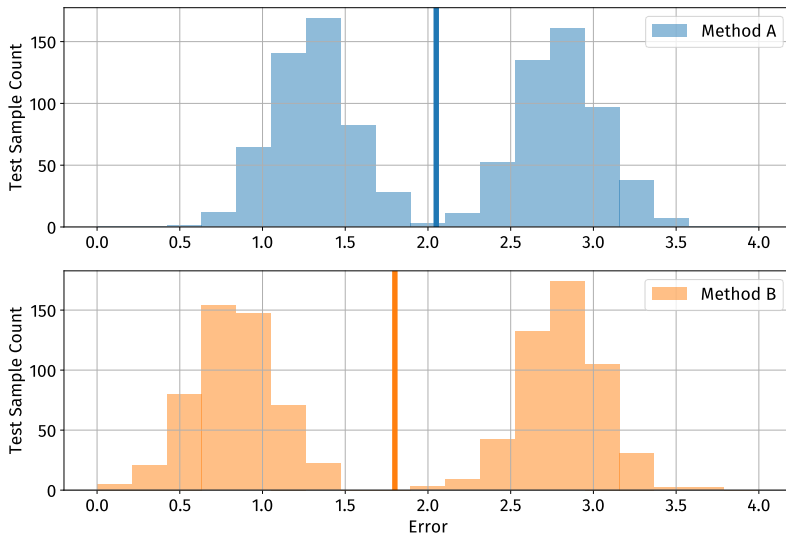
Evaluation: Look at test error distributions (e.g. histograms..)



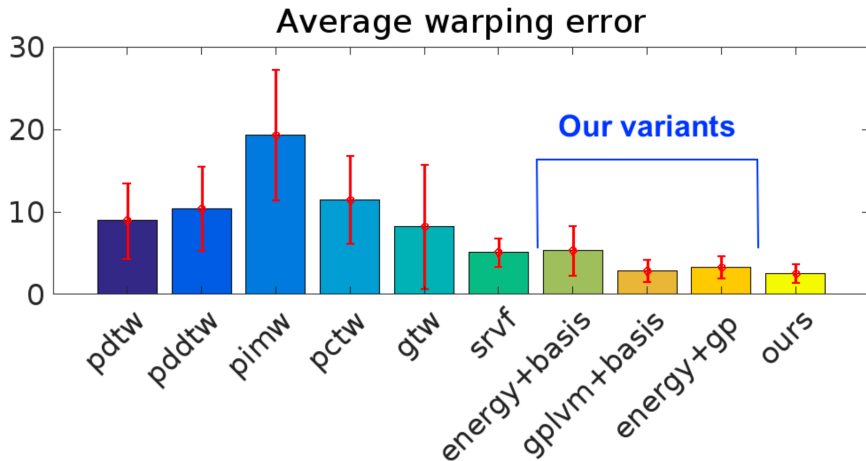
Evaluation: Look at test error distributions (e.g. histograms..)



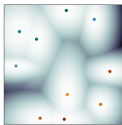
Evaluation: Look at test error distributions (e.g. histograms..)



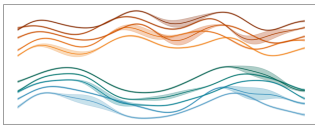
Evaluation: Alignments example with error bars..



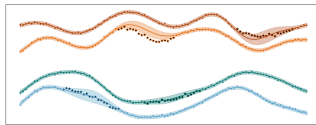
Evaluation: Alignments example with error bars..



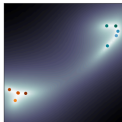
(a) MTGP $\mathbf{Z}$



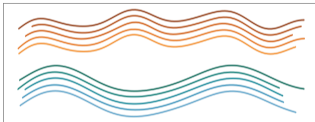
(b) MTGP function posteriors (unaligned)



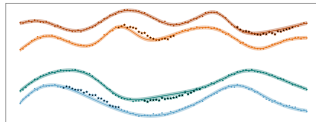
(c) MTGP missing data examples



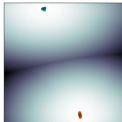
(d) GP-LVA $\mathbf{Z}$



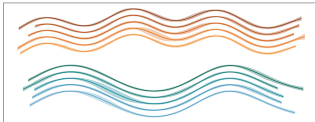
(e) GP-LVA function posteriors (aligned)



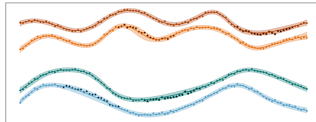
(f) GP-LVA missing data examples



(g) AMTGP $\mathbf{Z}$

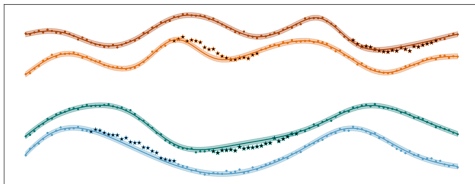


(h) AMTGP function posteriors (aligned)

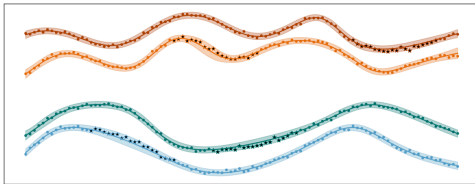


(i) AMTGP missing data examples

Evaluation: Alignments example with error bars..

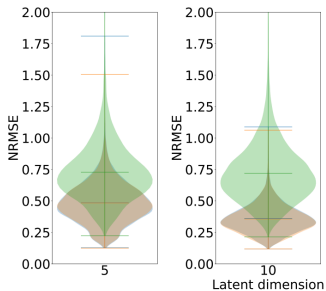


(f) GP-LVA missing data examples

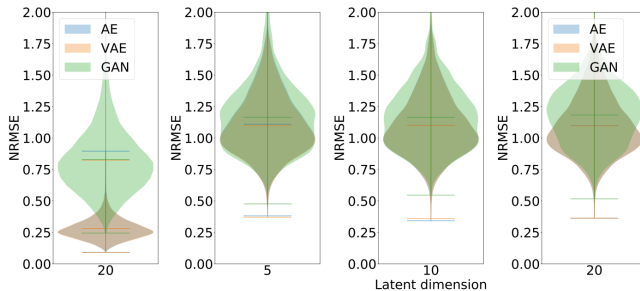


(i) AMTGP missing data examples

Evaluation: Example with histograms (e.g. violin plots)..



(a) MNIST dataset



(b) Shapes dataset

Evaluation: Histograms with illustrations!

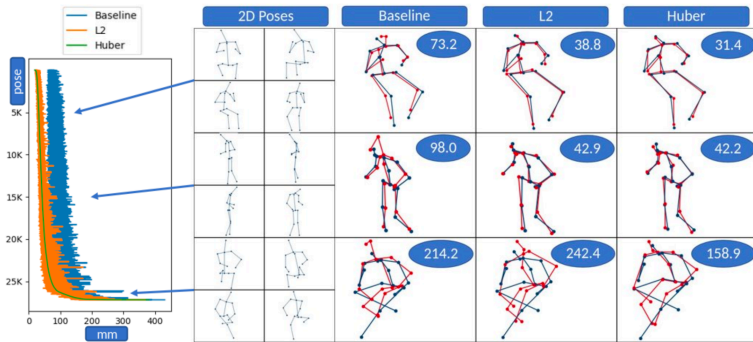


Figure 1: Error distribution on Human36M using our multi-camera model. Results are included for the untrained (Baseline) network and for the learnt shapes for the energy functions using both the ℓ_2 and the Huber loss. For visual clarity, we sort the datapoints by order of increasing error for the Huber case (i.e. our most effective approach). All models perform well on typical input instances, and fail in a limited number of cases. These outliers, however, have a noticeable, negative impact on the average error. We also provide reconstruction examples from the low, medium and tail part of the distribution.

Conclusions

Overview...

Motivation

No Free Lunch

Whirlwind Introduction to Inverse Probabilities

Model Selection

Gaussian Processes (a Prior over Functions)

Application: Bayesian Optimisation

Evaluation

Conclusions

Conclusions

- Hopefully motivated the need for uncertainty in ML (?)
- These techniques are important!
- Make ML safe, trustworthy and robust for applications
- Rigour in experimental validation
- There is still much work to be done here!

Further Reading

- Understanding Deep Learning, Simon J.D. Prince
 - <https://udlbook.github.io/udlbook/>
- Information Theory, Inference, and Learning Algorithms, David MacKay
 - <https://www.inference.org.uk/itprnn/book.html>
- Pattern Recognition and Machine Learning, Christopher M. Bishop
 - Microsoft Website with PDF
- Computer Vision: Models, Learning, and Inference, Simon J.D. Prince
 - <http://www.computervisionmodels.com/>
- Probabilistic Numerics: Computation as Machine Learning, Philipp Hennig, Michael A. Osborne, Hans Kersting
 - <https://www.probablistic-numerics.org/textbooks/>

That's all folks..