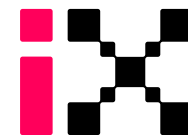


IMPERIAL



Schmidt Sciences

AIMS Week 3

Bayesian models for reaction kinetics



Laura Helleckes
17/02/2026

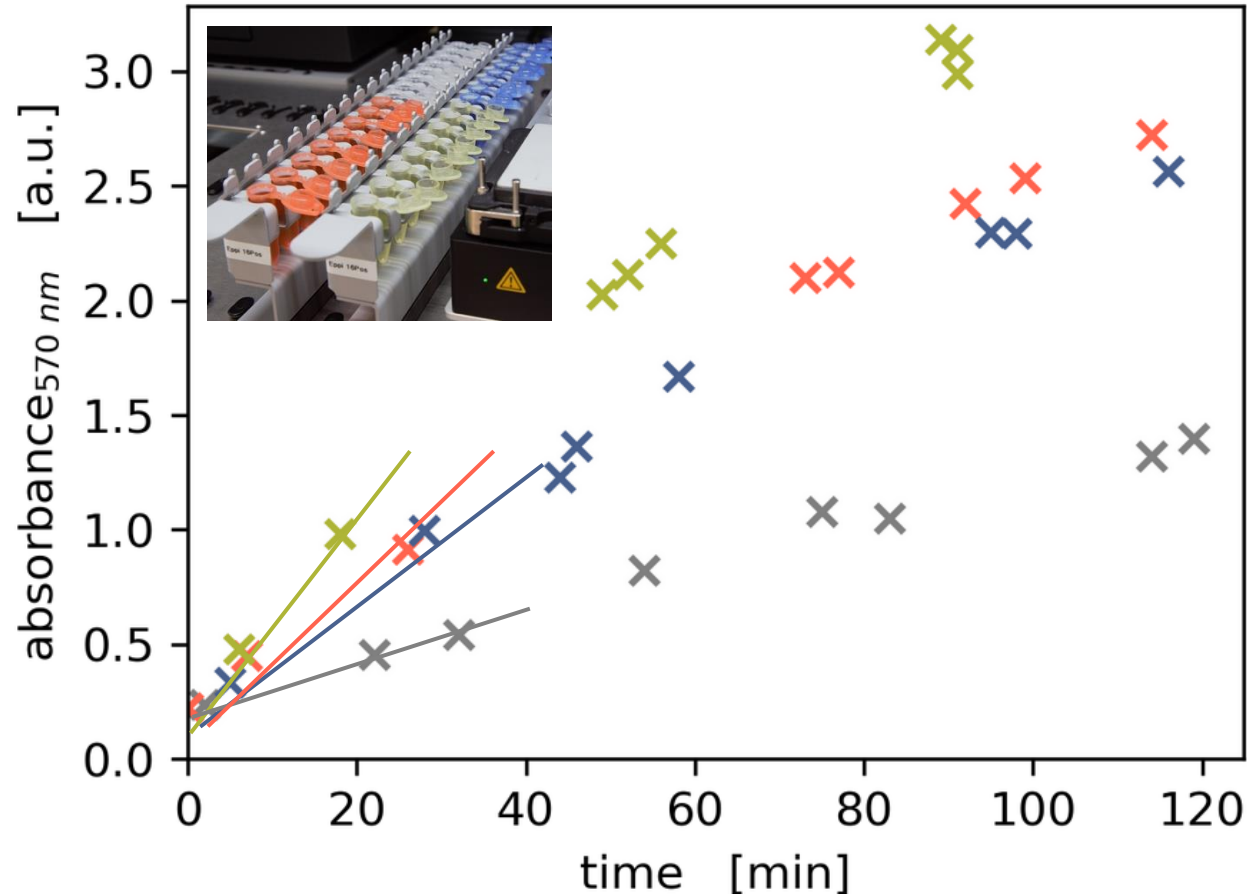
Learning outcomes

After today's lecture and the exercise, you should be able to

1. Build and compare **mechanistic vs. hybrid Bayesian models**
2. Diagnose **MCMC convergence** using trace plots, \hat{R} , and ESS
3. Use **posterior predictive checks** to assess model adequacy
4. Visualise and interpret **prediction uncertainty**

Motivation for statistical models

Robotic platform



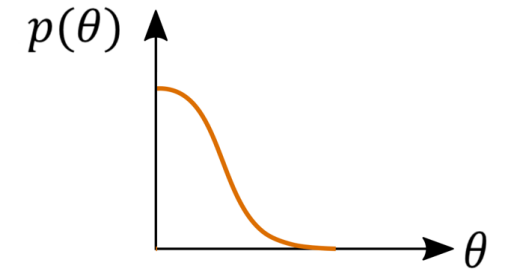
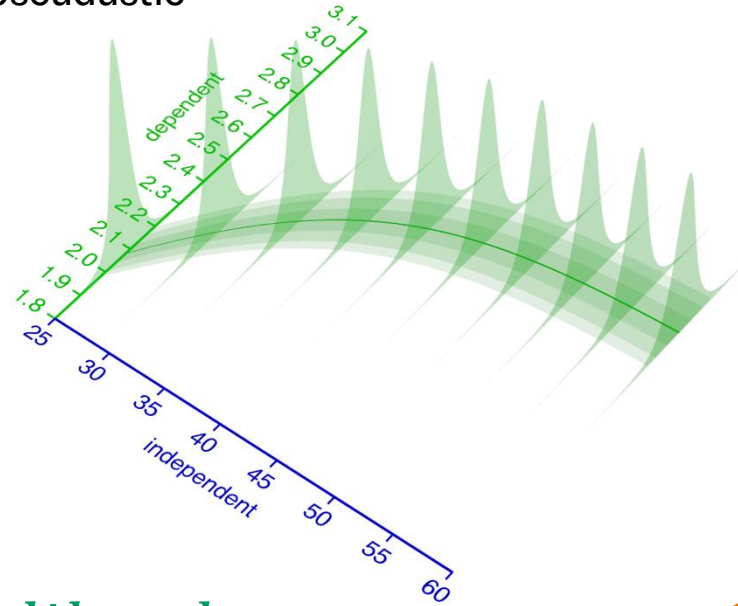
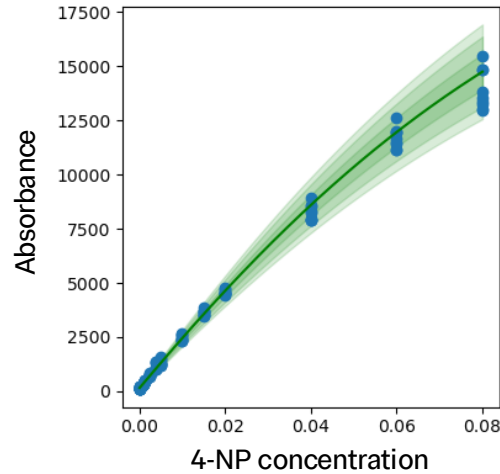
- Determine **reaction rate** of 4 different enzymes
- Product measured by **absorbance**
- Frequent: **initial rates** by linear regression
- Bias: How many points to choose?
- Do we properly take the uncertainty into account?

Can we build models that include uncertainty and systematically incorporate external effects?

Bayesian statistical modelling

Bayes theorem

non-linear, heteroscedastic



posterior \rightarrow $p(\theta|D) = \overset{\text{likelihood}}{p(D|\theta)} \cdot \frac{p(\theta)}{\int_{\theta} p(D|\theta) \cdot p(\theta) d\theta}$

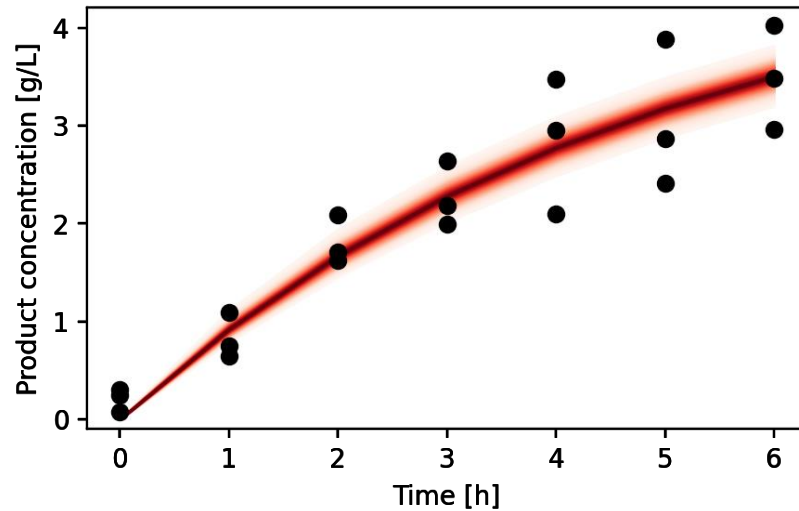
prior \rightarrow $p(\theta)$

evidence \rightarrow $\int_{\theta} p(D|\theta) \cdot p(\theta) d\theta$

Bayesian inference

Fully mechanistic model

$$\begin{array}{c}
 \text{posterior} \swarrow \\
 p(\theta|D) = \underset{\substack{\text{likelihood} \\ \searrow}}{p(D|\theta)} \cdot \frac{\underset{\substack{\text{prior} \\ \swarrow}}{p(\theta)}}{\int_{\theta} p(D|\theta) \cdot p(\theta) d\theta} \leftarrow \text{evidence}
 \end{array}$$



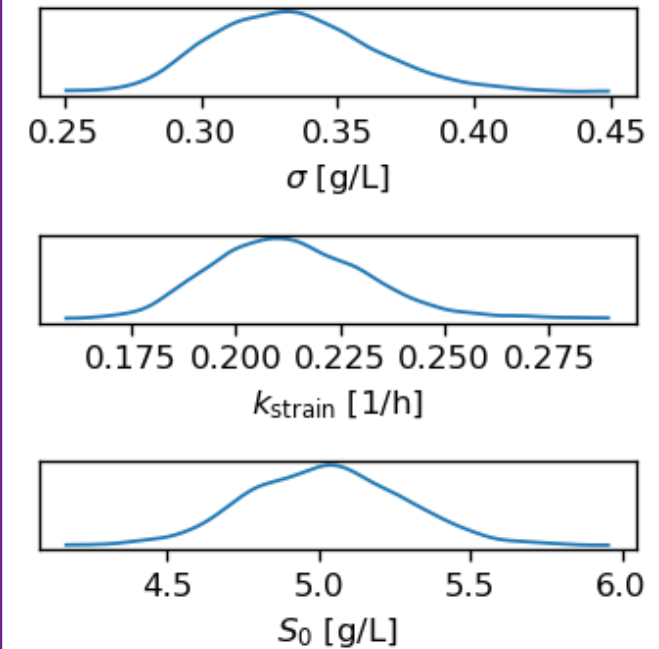
$$\mathcal{L}(Y_{\text{obs}} | \hat{Y}) \sim \mathcal{N}(\mu, \sigma^2)$$

$$\mu = S_0 \cdot (1 - e^{-k_{\text{strain}} \cdot t})$$

$$\sigma \sim |\mathcal{N}\left(0 \frac{\text{g}}{\text{L}}, 1 \frac{\text{g}}{\text{L}}\right)|$$

$$k_{\text{strain}} \sim |\mathcal{N}\left(0.5 \frac{1}{\text{h}}, 3 \frac{1}{\text{h}}\right)|$$

$$S_0 \sim \mathcal{N}\left(5 \frac{\text{g}}{\text{L}}, 0.5 \frac{\text{g}}{\text{L}}\right)$$



Bayesian inference

Intractable problems

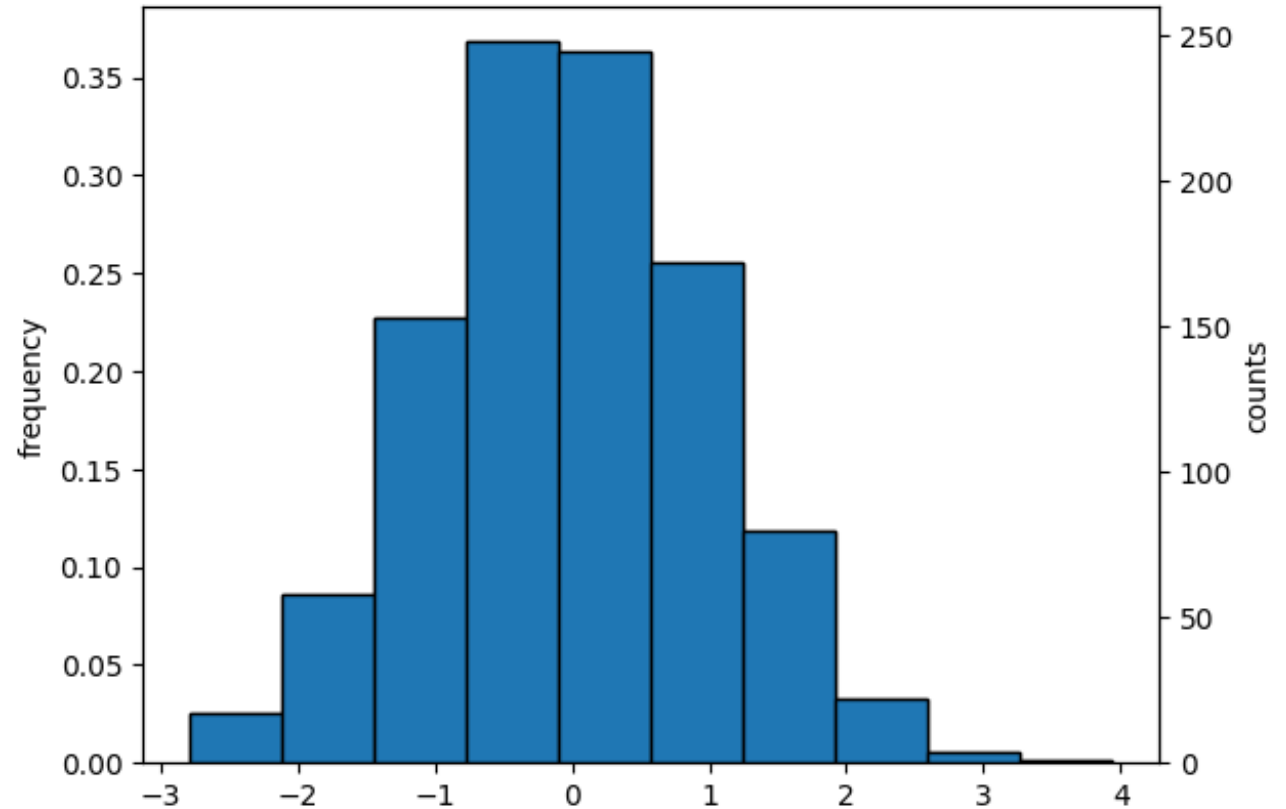
- Given prior distribution and likelihood function, Bayes' Theorem defines posterior distribution
- Practically, the **unnormalised posterior** is often available
- Normalising constant is infeasible, **because integration requires to explore the whole space first**

$$p(\mathbf{f}) = \int p(\mathbf{f} | \boldsymbol{\theta}) p(\boldsymbol{\theta}) d\boldsymbol{\theta}$$

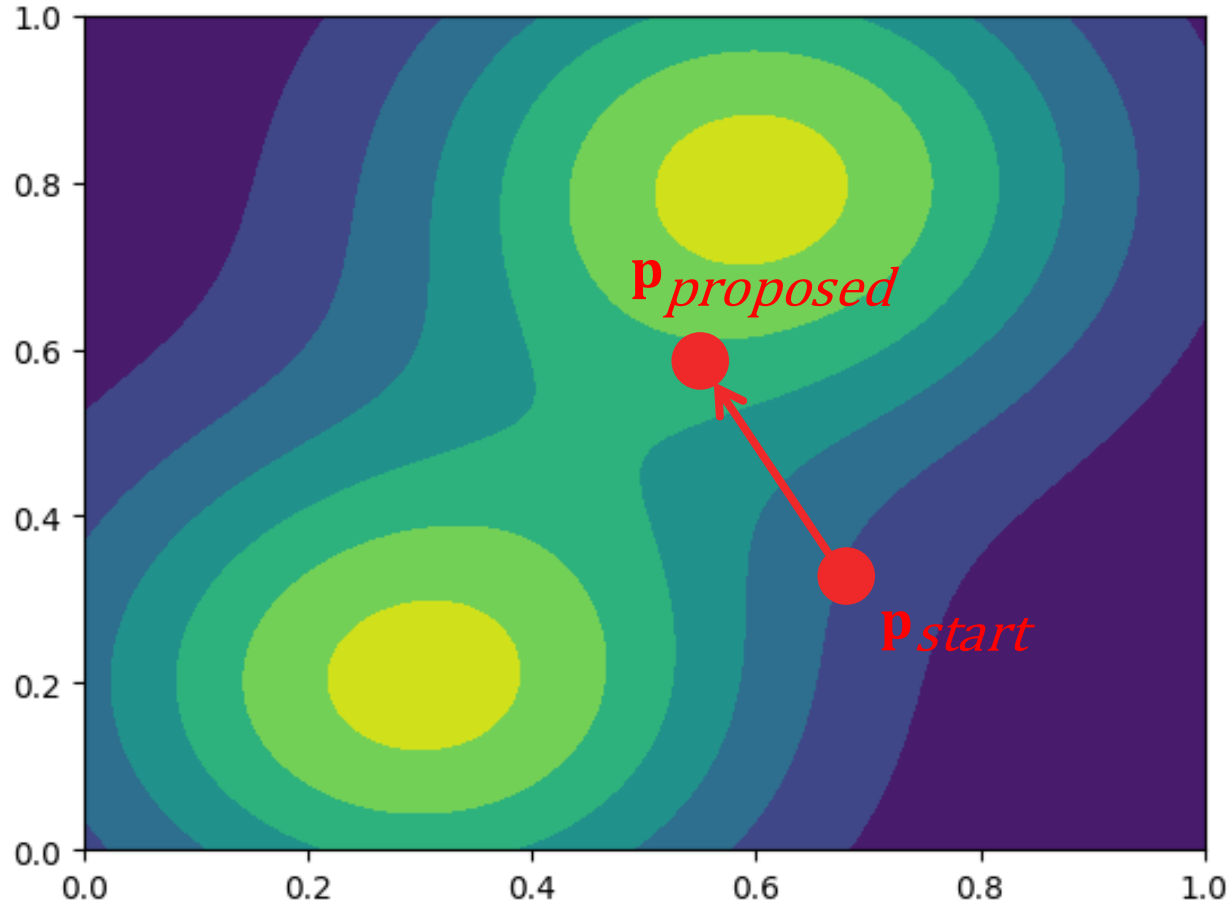
Markov chain Monte Carlo (MCMC) sampling

Basic idea

- MCMC approximates the posterior through samples – which does not require normalisation!
- Samples are drawn iteratively and randomised



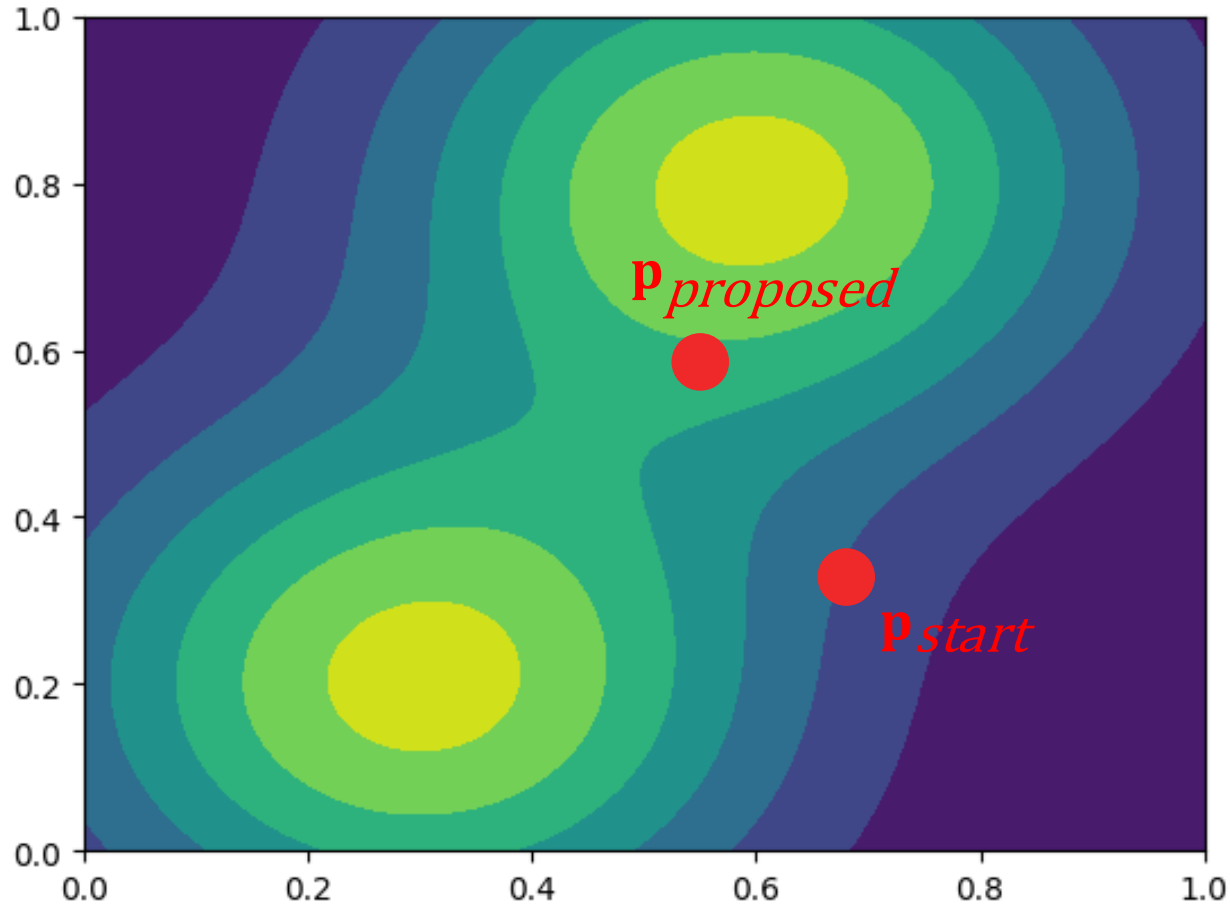
1. Proposal



- MCMC approximates the posterior through samples – which does not require normalisation!
- An iteration begins at a starting point
- From here, a new point is proposed, e.g. by

$$\mathbf{p}_{proposed} \sim \mathcal{N}(\mathbf{p}_{start}, \sigma^2 \mathbf{1})$$

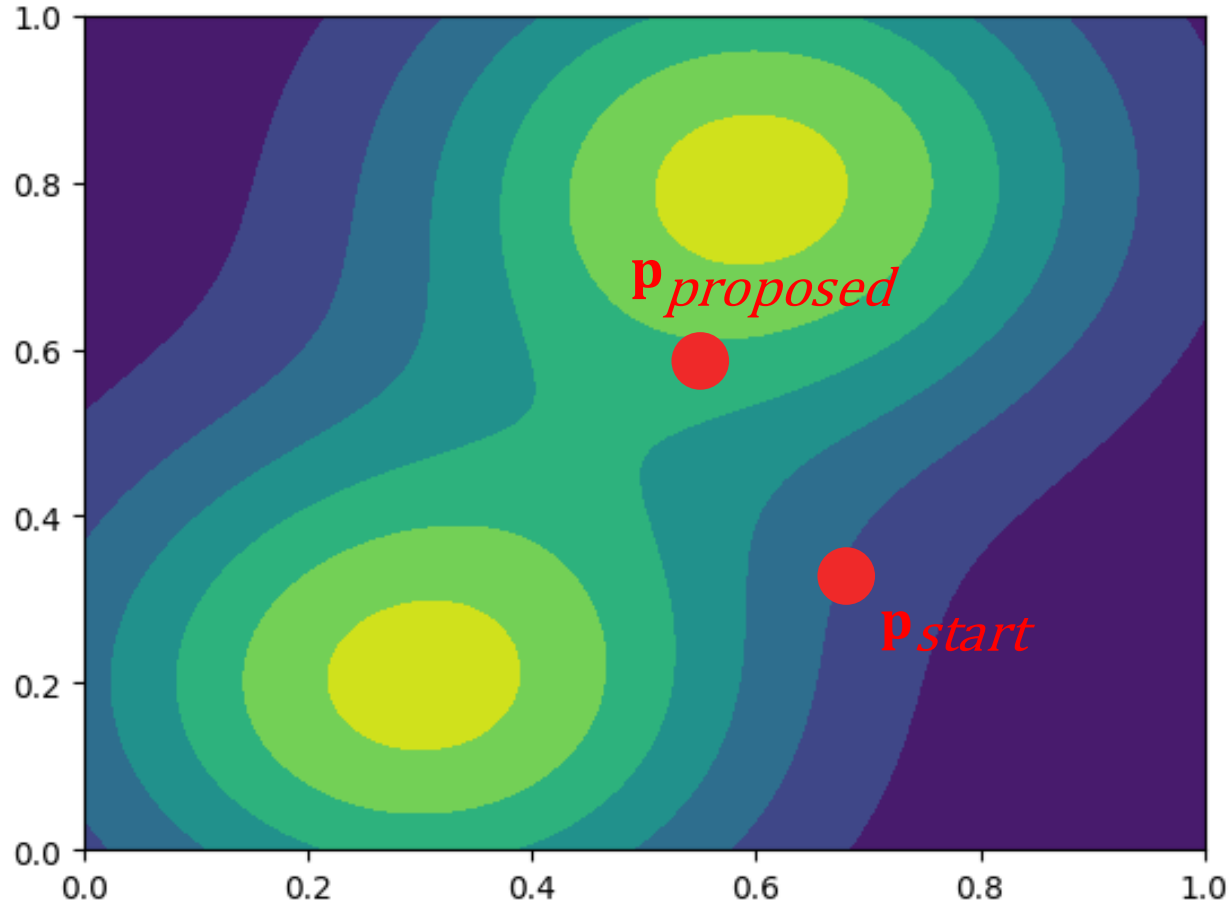
2. Acceptance probability



- To decide whether to keep the proposed point, a special probability is computed
- Compares the likelihood of the proposal to that of the start:

$$r = \frac{p(\theta^* | y)}{p(\theta^{t-1} | y)}$$

3. Acceptance/Rejection



- Whether the proposal is kept, is decided by a coin toss with the acceptance probability:

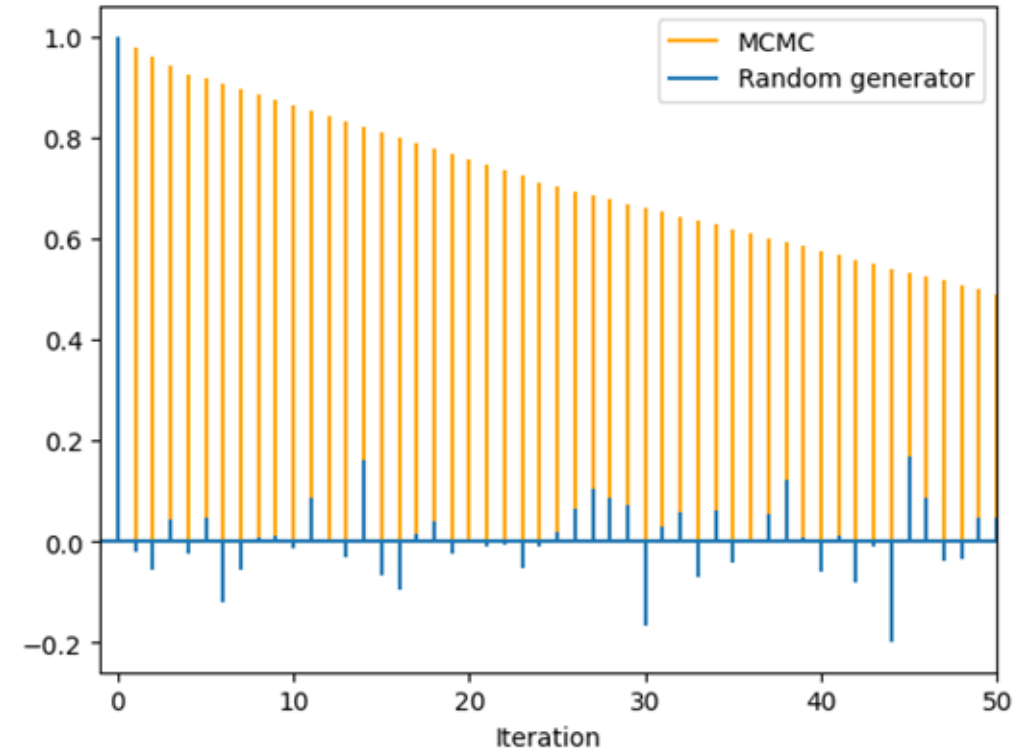
$$\theta^{t-1} = \begin{cases} \theta^* & \text{with probability } \min(r, 1) \\ \theta^{t-1} & \text{otherwise} \end{cases}$$

- Fail:** process stays at starting point
- Success:** proposal is next starting point

MCMC Demo by **Chi Feng**

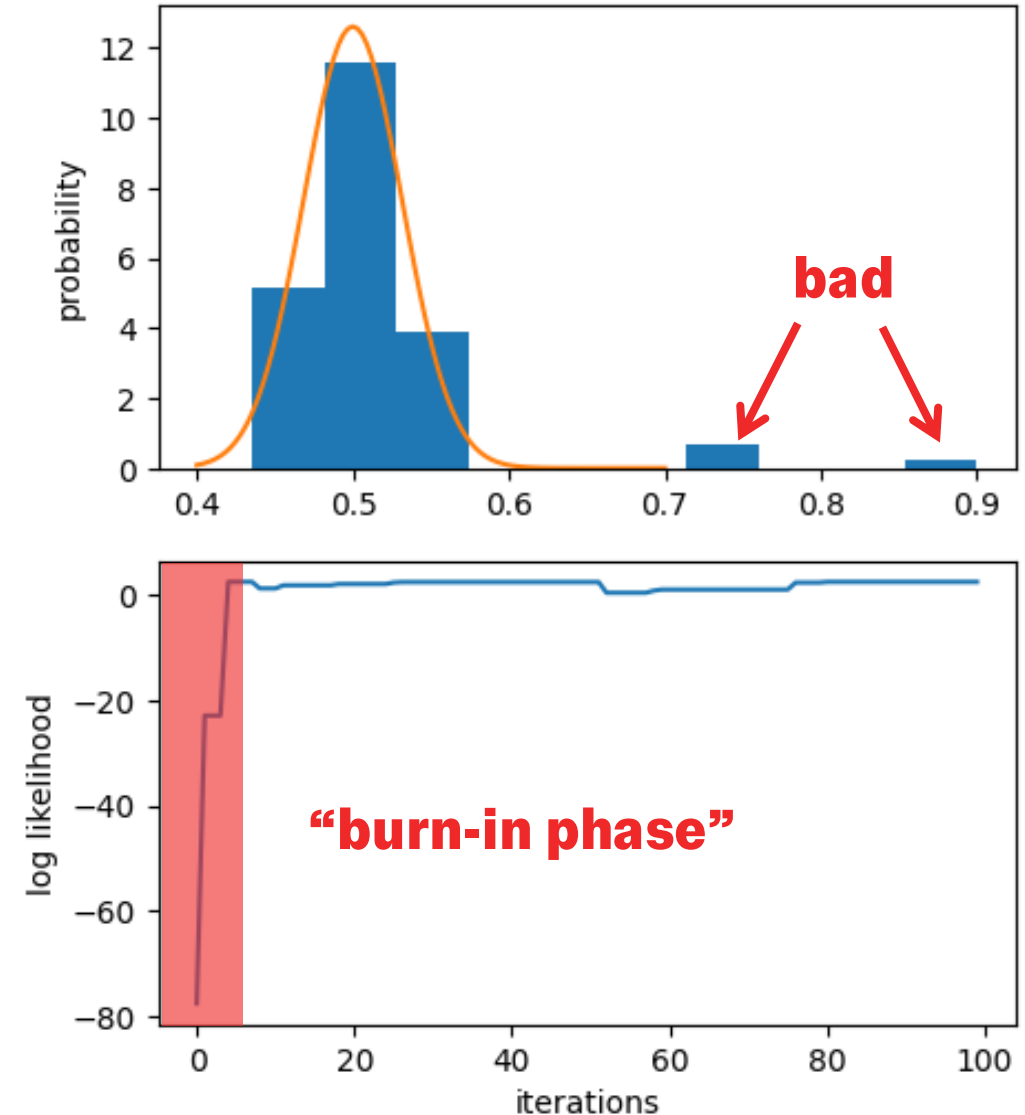
Autocorrelation

- MCMC samples are **highly correlated**
- Correlated samples have redundant information about the posterior density
- Less correlation means higher efficiency → Better proposals
- **Effective sample size (ESS)** is a measure expressing how efficient the MCMC draws are compared to random samples (>400)



Burn-in

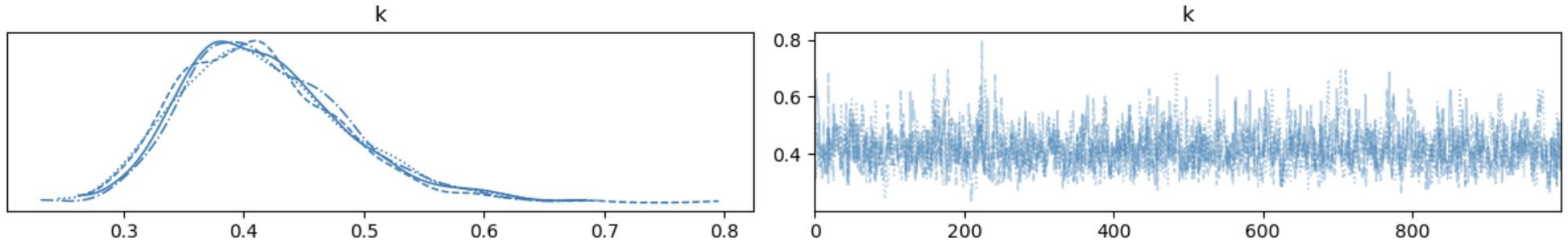
- MCMC works for every starting point
- Bad starting points lead to **overrepresented samples**
- How to get rid of this bias?
 - a) Sample long enough to outweigh bad samples (inefficient)
 - b) Discard early samples
 - c) Find good starting point



Diagnostics

How do we know that MCMC actually converged?

→ Start at different random points and see if they all find the same solution (chains)



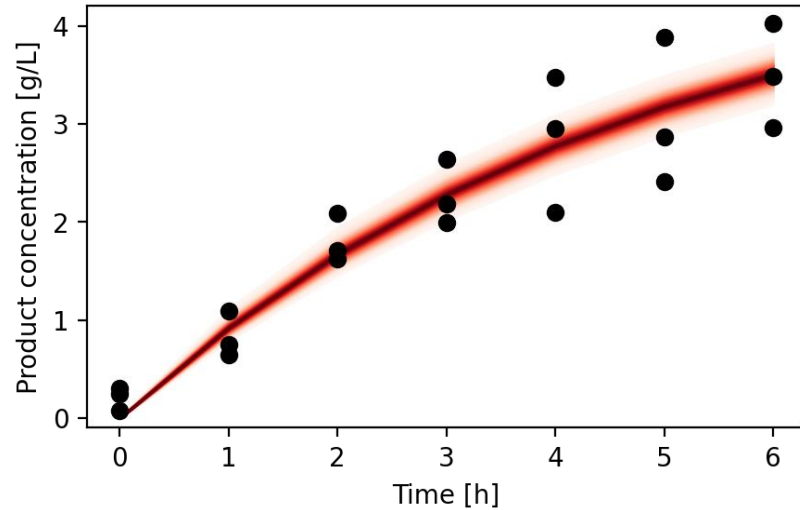
Gelman-Rubin statistic \hat{R} : Ratio of variance BETWEEN chains and variance WITHIN chains (< 1.05)

Notebook 1

Parameter estimation with MCMC

Bayesian inference

Hybrid model

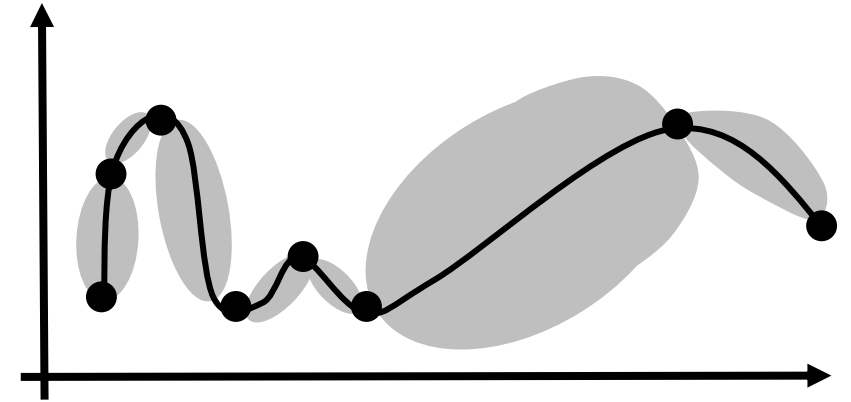


$$\mu = S_0 \cdot (1 - e^{-k \cdot t})$$

$$k = f(x)$$

where x could be influences such as temperature or pH.

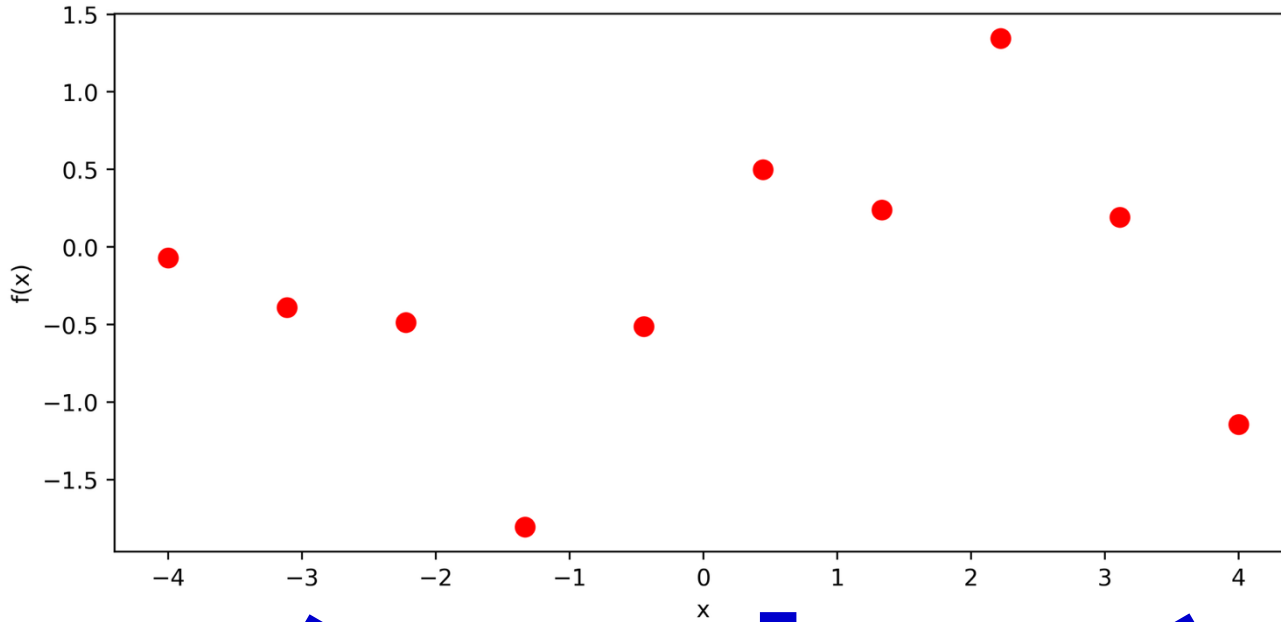
Hybrid model



Use a Gaussian process for k :

$$k \sim GP(m(x), \kappa(x, x'))$$

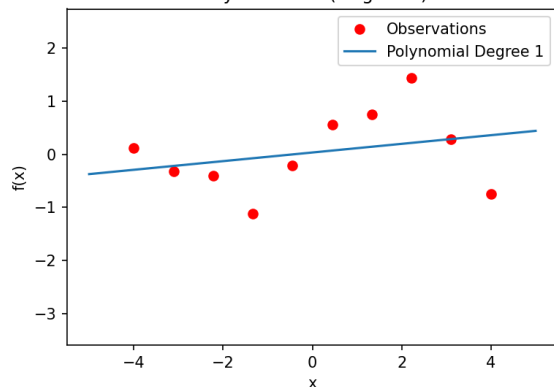
Regression task: A well-known problem



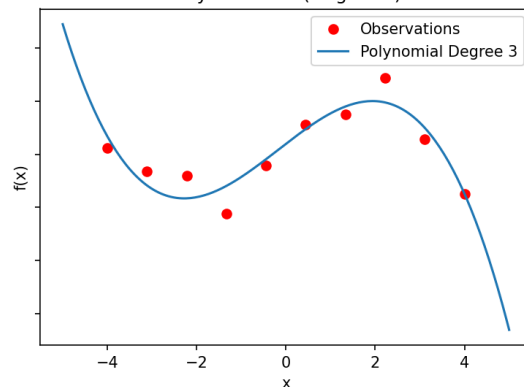
- So far, we studied how we can fit a single parametric model (e.g. linear) to our data
- Even if we do model comparison, we have to define the set of models we test (Llyod will teach you some smart ways)
- What happens if we sequentially find new data?



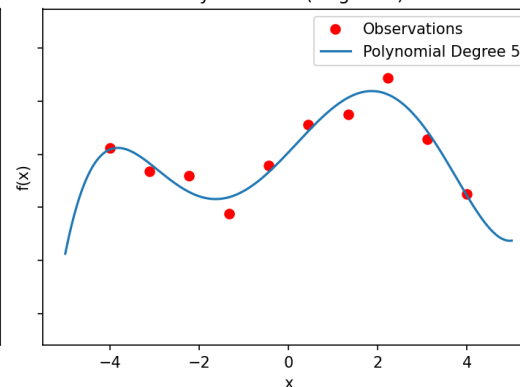
Polynomial Fit (Degree 1)



Polynomial Fit (Degree 3)



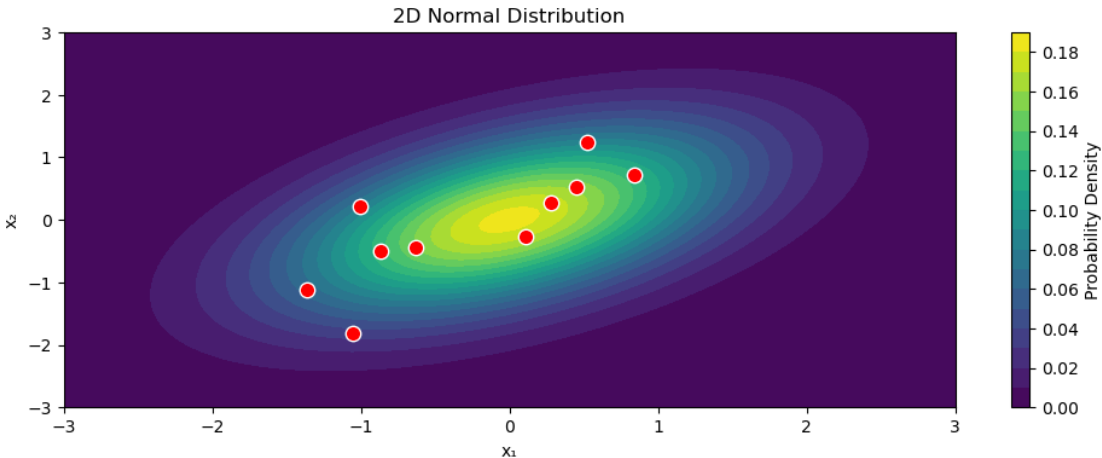
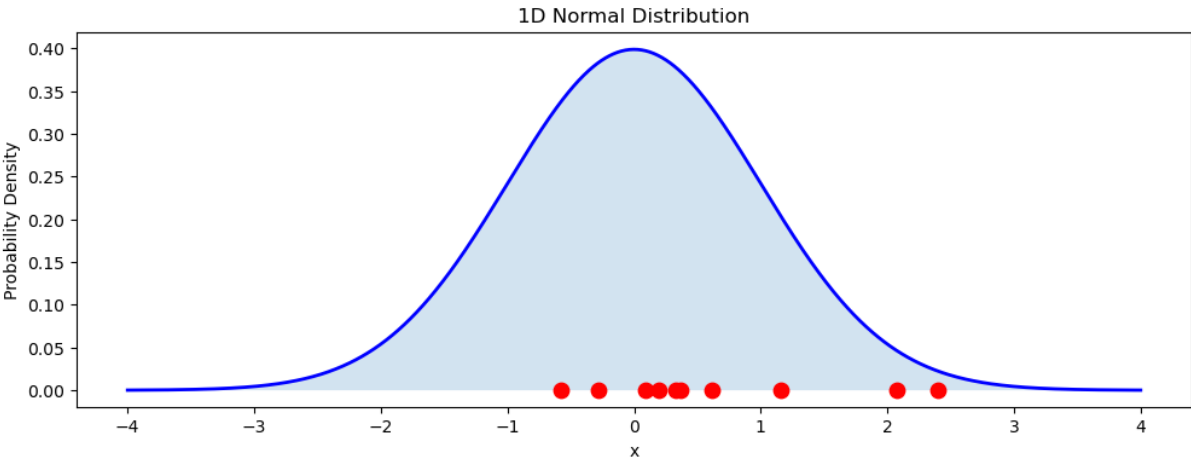
Polynomial Fit (Degree 5)



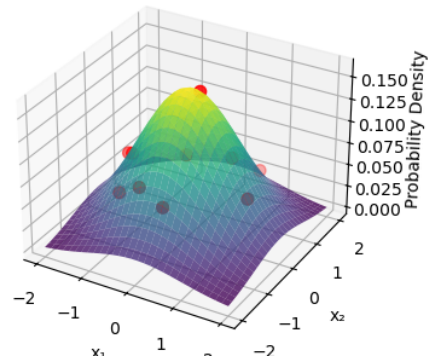
?

Can we find a universal approximator?

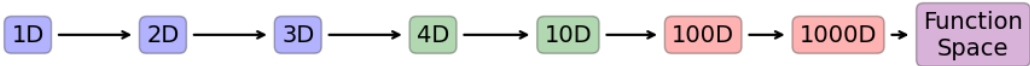
Sampling from covariance matrix to generate functions



3D Visualization of Bivariate Normal

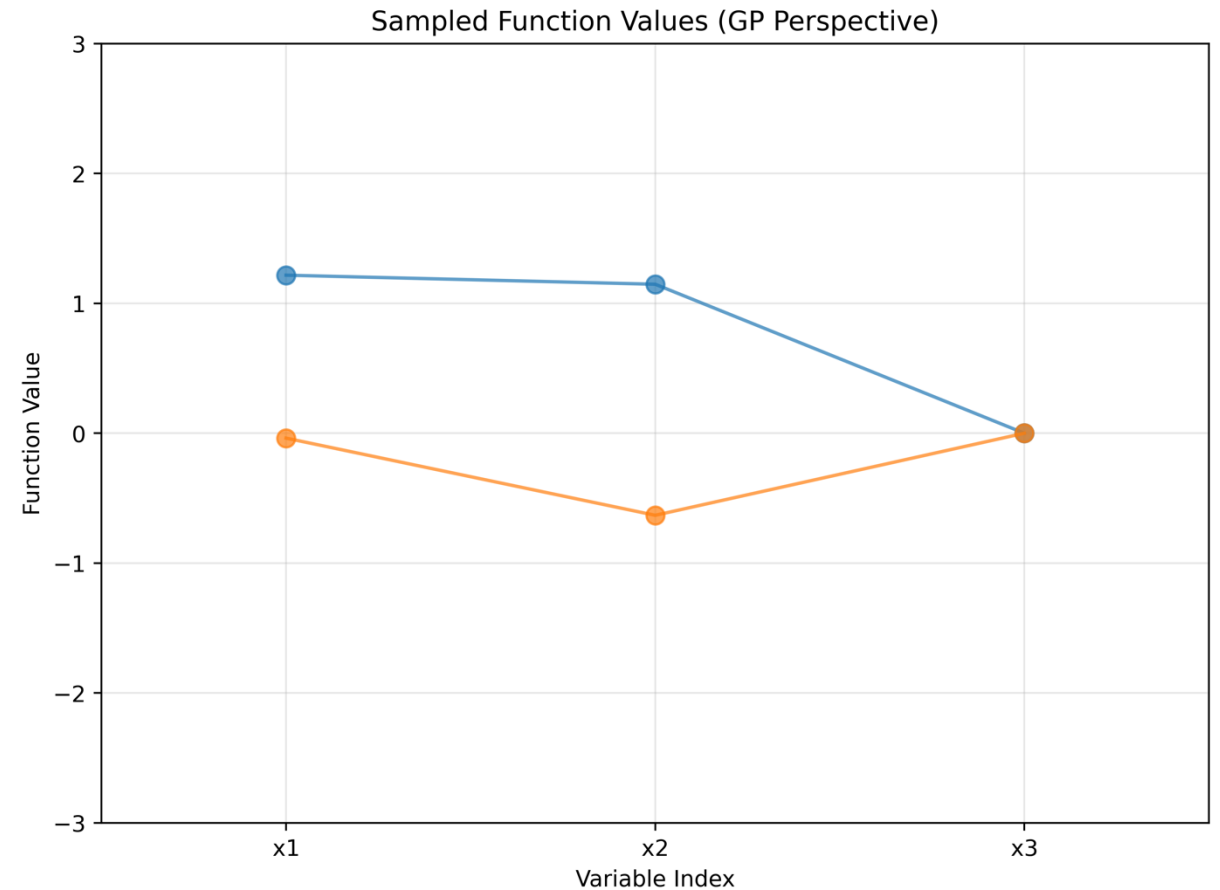
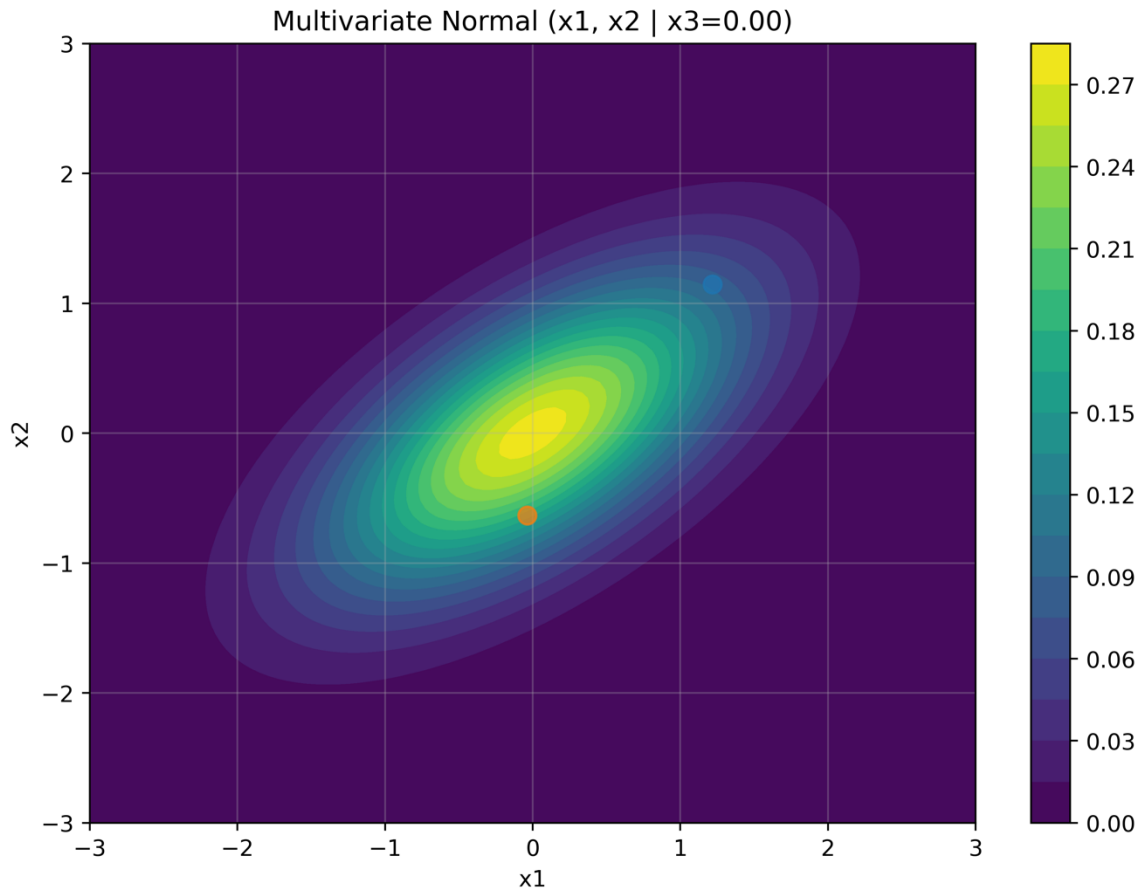


Conceptual: From Low Dimensions to Function Spaces



What does it look like if we plot the read points as a sequence of values?

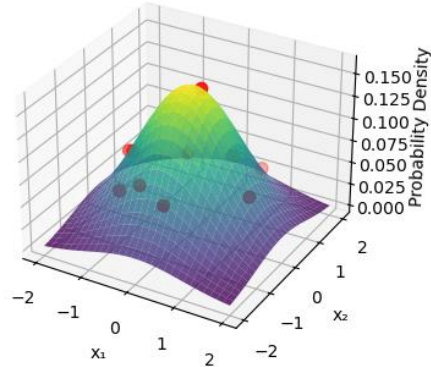
Approximate functions with Gaussian distributions



These samples look a bit like a regression problem already!

Extend to higher dimensions to approximate functions

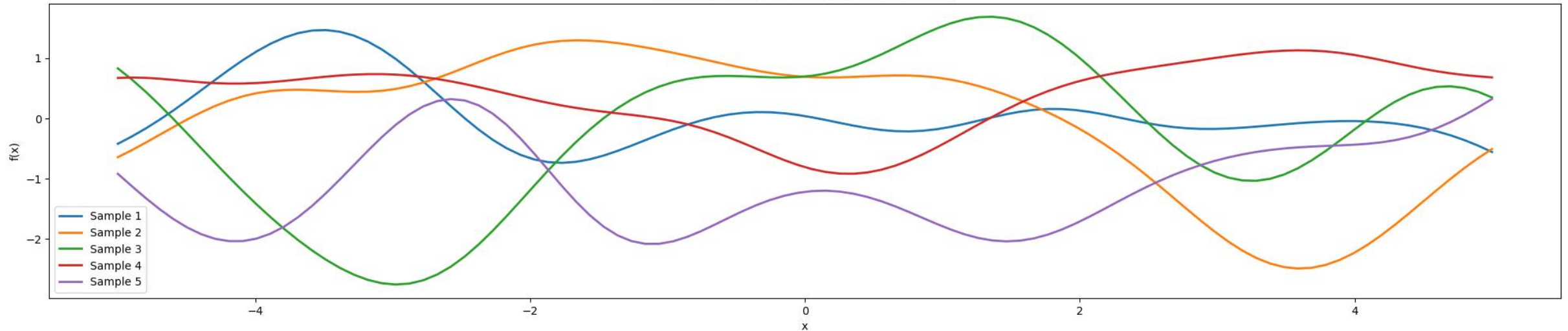
3D Visualization of Bivariate Normal



Conceptual: From Low Dimensions to Function Spaces



Samples from a 100-Dimensional Multivariate Normal
(Each function is one sample point from MVN)



In high dimension, we get curves that are quite flexible!

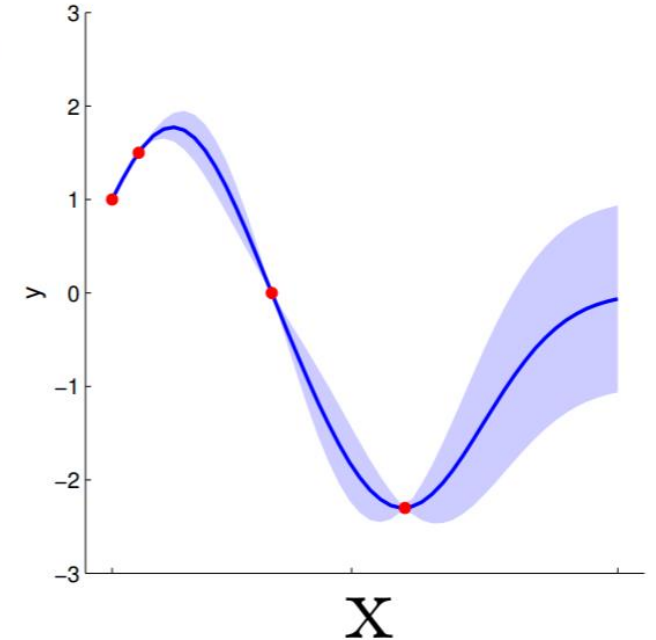
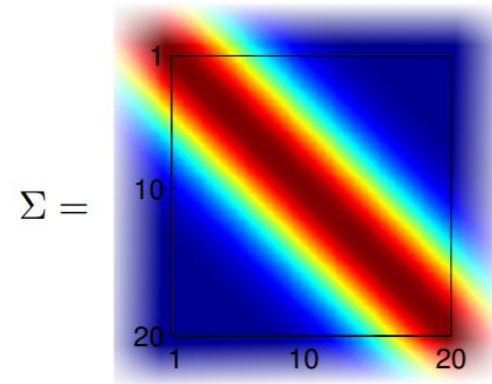
Generalisation: Kernel functions

- In practice, the covariance matrix for a specific set of points is generated from a more generalised function
- This can be interpreted as a model for covariances (it has some additional parameters that we will study soon)
- This family of models is called kernel functions

Yuge Shi, "Gaussian Processes, not quite for dummies", *The Gradient*, 2019.

$$\Sigma(x_1, x_2) = K(x_1, x_2) + I\sigma_y^2$$

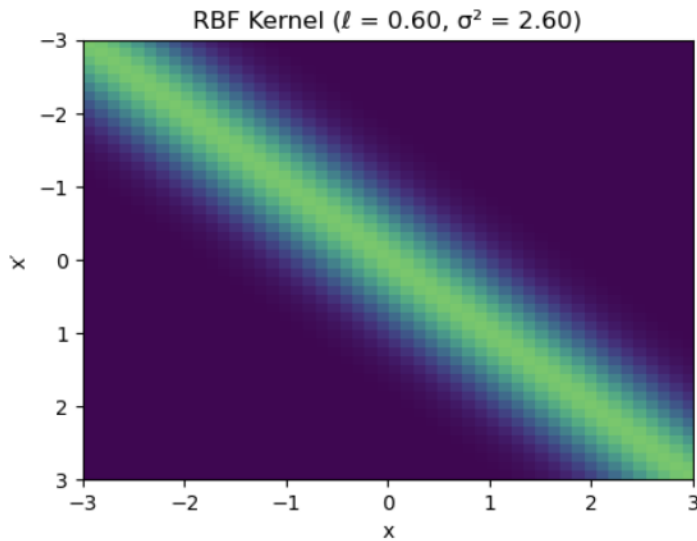
$$K(x_1, x_2) = \sigma^2 e^{-\frac{1}{2l^2}(x_1 - x_2)^2}$$



Kernel functions give us a generalisable way to generate the correlated samples

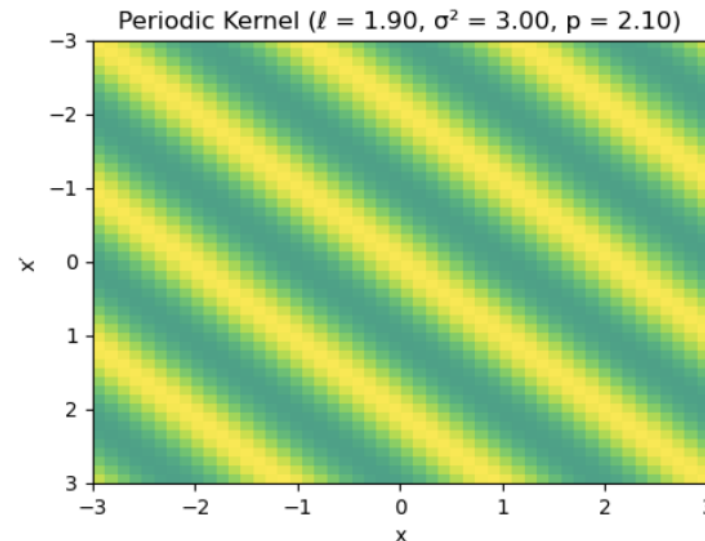
Different kernel functions

The kernel has a strong influence on the functions we generate with Gaussian processes
Popular kernels include radial base function (RBF), periodic or linear kernels



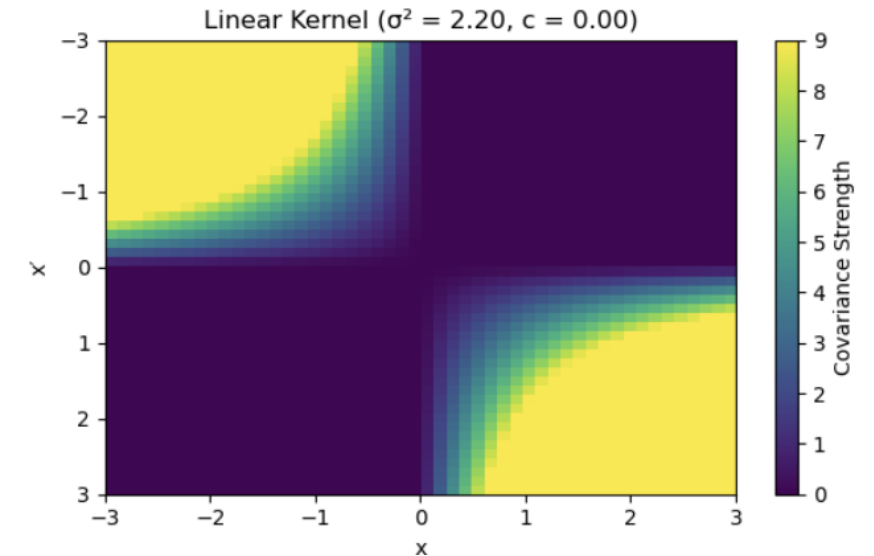
$$k(x, x') = \sigma^2 \exp\left(-\frac{1}{2} \frac{\|x - x'\|^2}{\ell^2}\right)$$

with σ^2 as output variance, ℓ as length scale and $\|x - x'\|$ as the Euclidean distance between the vectors.



$$k(x, x') = \sigma^2 \exp\left(-\frac{2 \sin^2(\pi \frac{\|x - x'\| p}{\ell})}{\ell^2}\right)$$

with σ^2 as output variance, ℓ as length scale, p as the period and $\|x - x'\|$ as the Euclidean distance between the vectors.

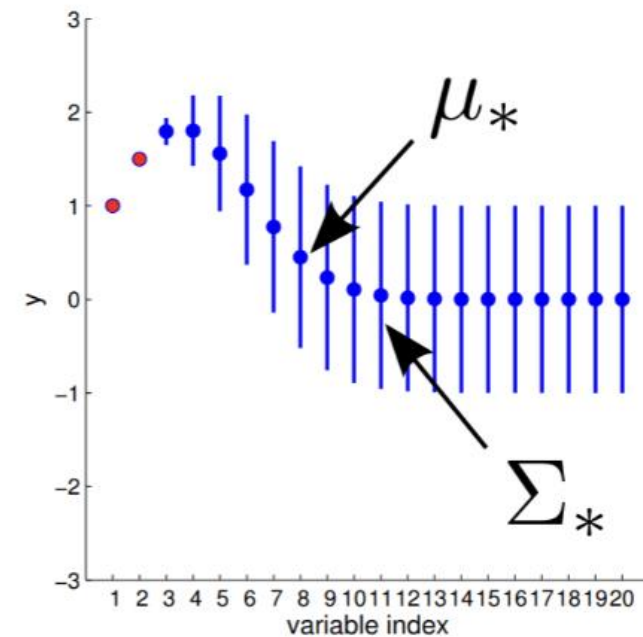
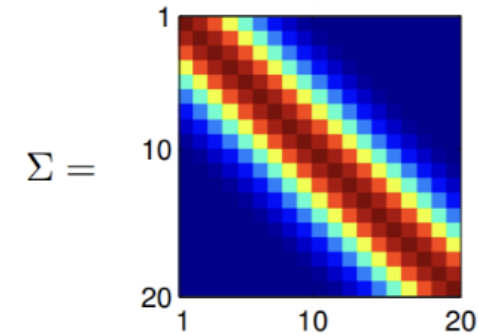


$$k(x, x') = \sigma^2 (x^T x' + c)$$

with σ^2 as output variance and c as the constant offset.

Mean and covariance functions

- If we generate many samples (i.e. functions from our kernel, we can visualise the mean and standard deviation at each point)
- This already indicates that using kernel functions in regression gives us **distributions over functions**
- We can now bring all of this together to introduce **Gaussian processes (GPs)**



Yuge Shi, "Gaussian Processes, not quite for dummies", *The Gradient*, 2019.

Bringing it all together: Gaussian processes

We have seen how **sampling variables from high-dimensional multivariate Gaussian distributions** allow us generate functions

Gaussian processes (GPs) are the extension of this concept to *infinitely many variables*

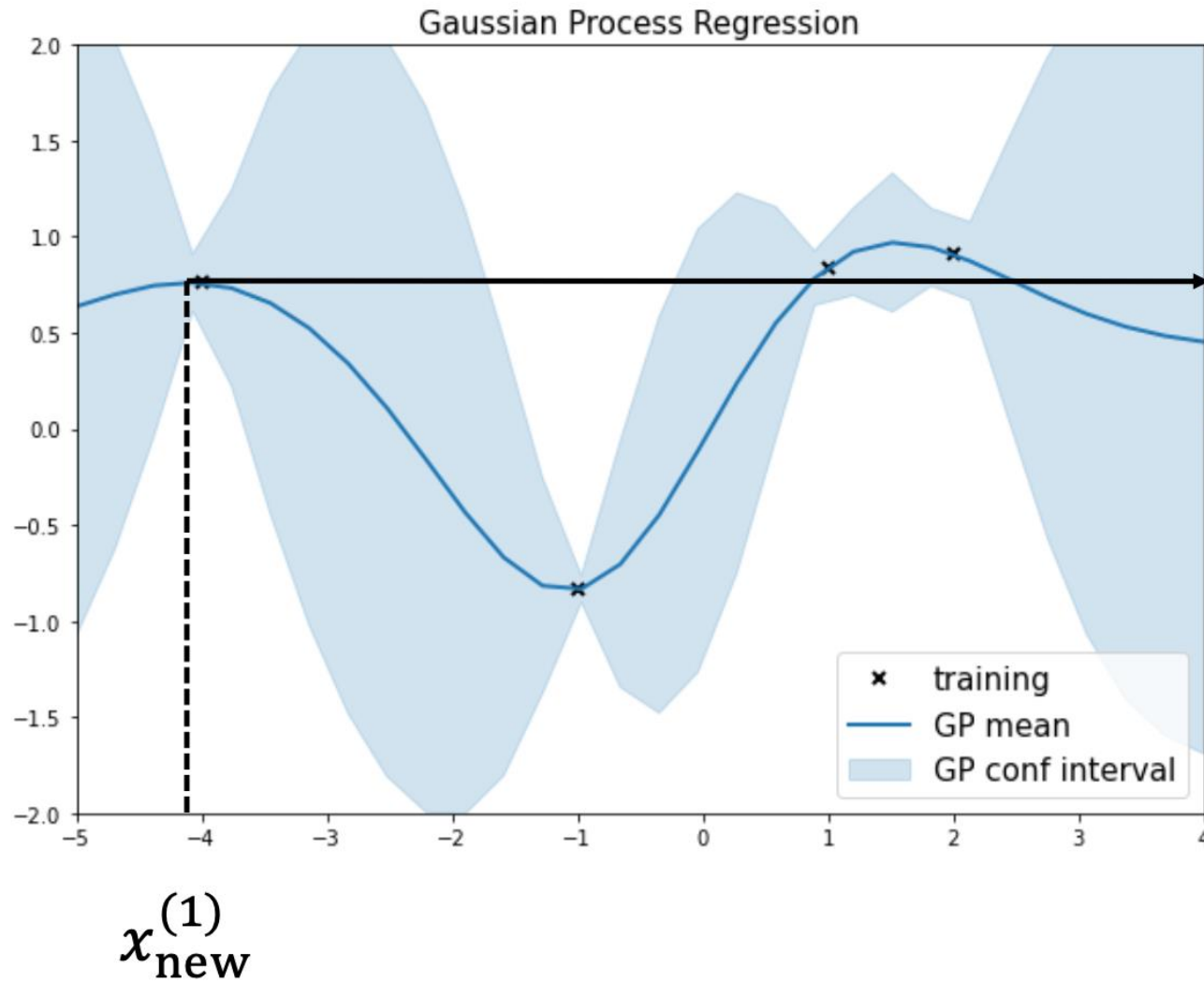
Textbook definition: *A Gaussian process is a collection of random variables, any finite number of which have consistent Gaussian distributions.*

The **classic notation** is the following: $f(x) \sim \mathcal{GP}(m(x), K(x, x'))$

It basically states that GPs, like the Normal distribution, are also fully described by a **mean function** and **covariance function (or kernel function)**

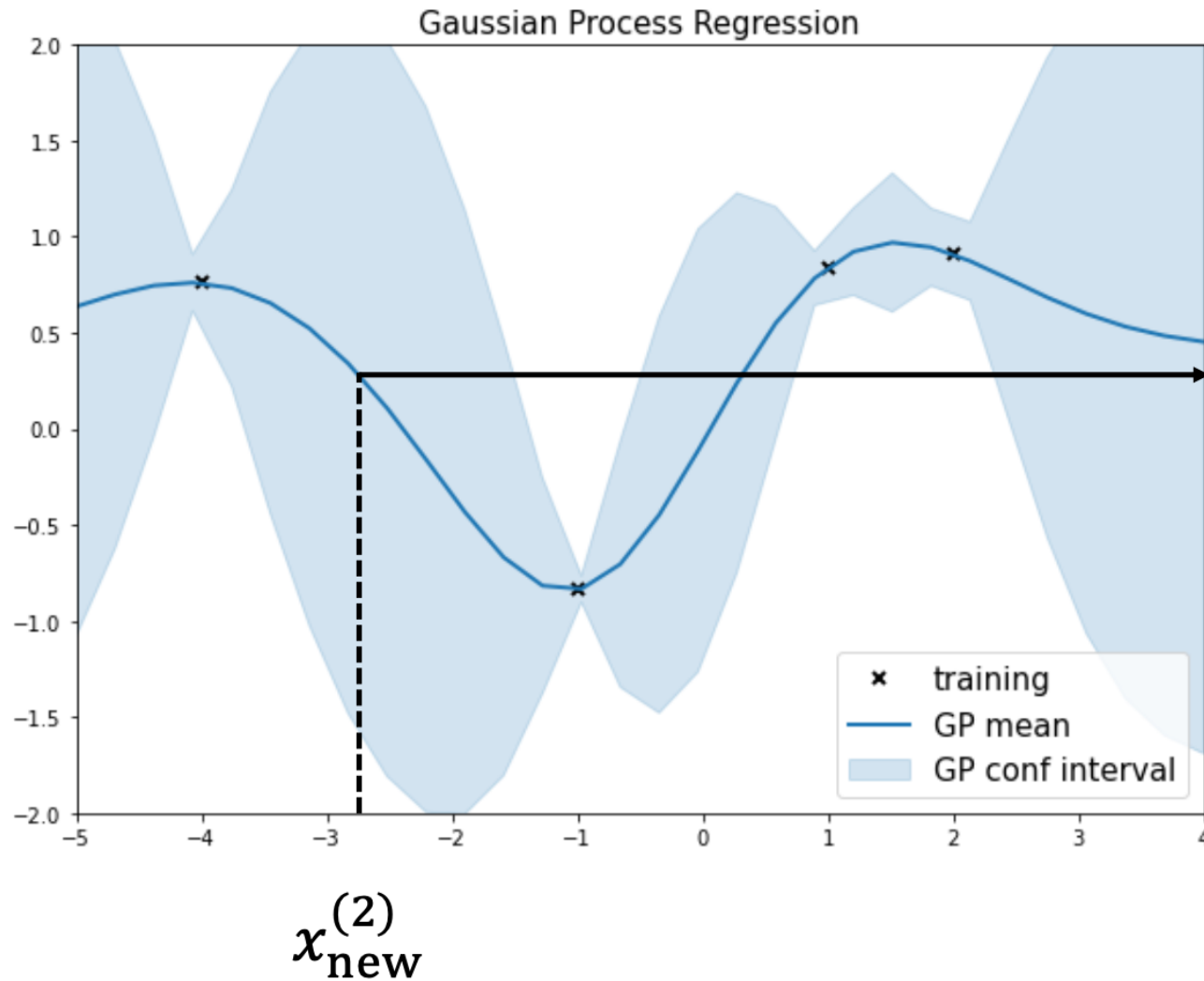
GPs are a generalisation of multivariate Normal distributions to infinitely many variables.

Mean and covariance functions



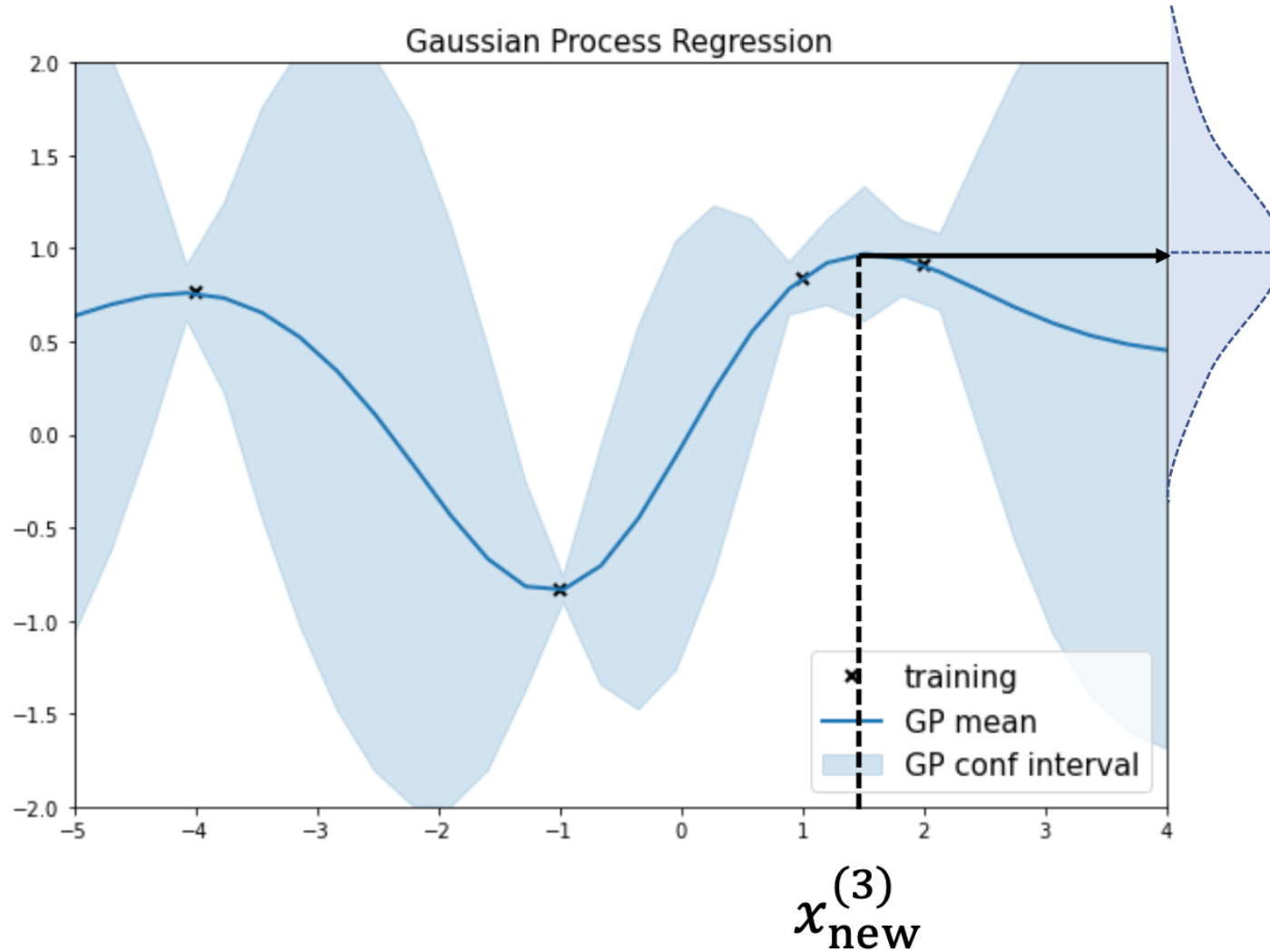
$$\hat{y}_{\text{pred}}^{(1)} \sim \mathcal{N}(\mu_y(x_{\text{new}}^{(1)}), \sigma_y^2(x_{\text{new}}^{(1)}))$$

Mean and covariance functions



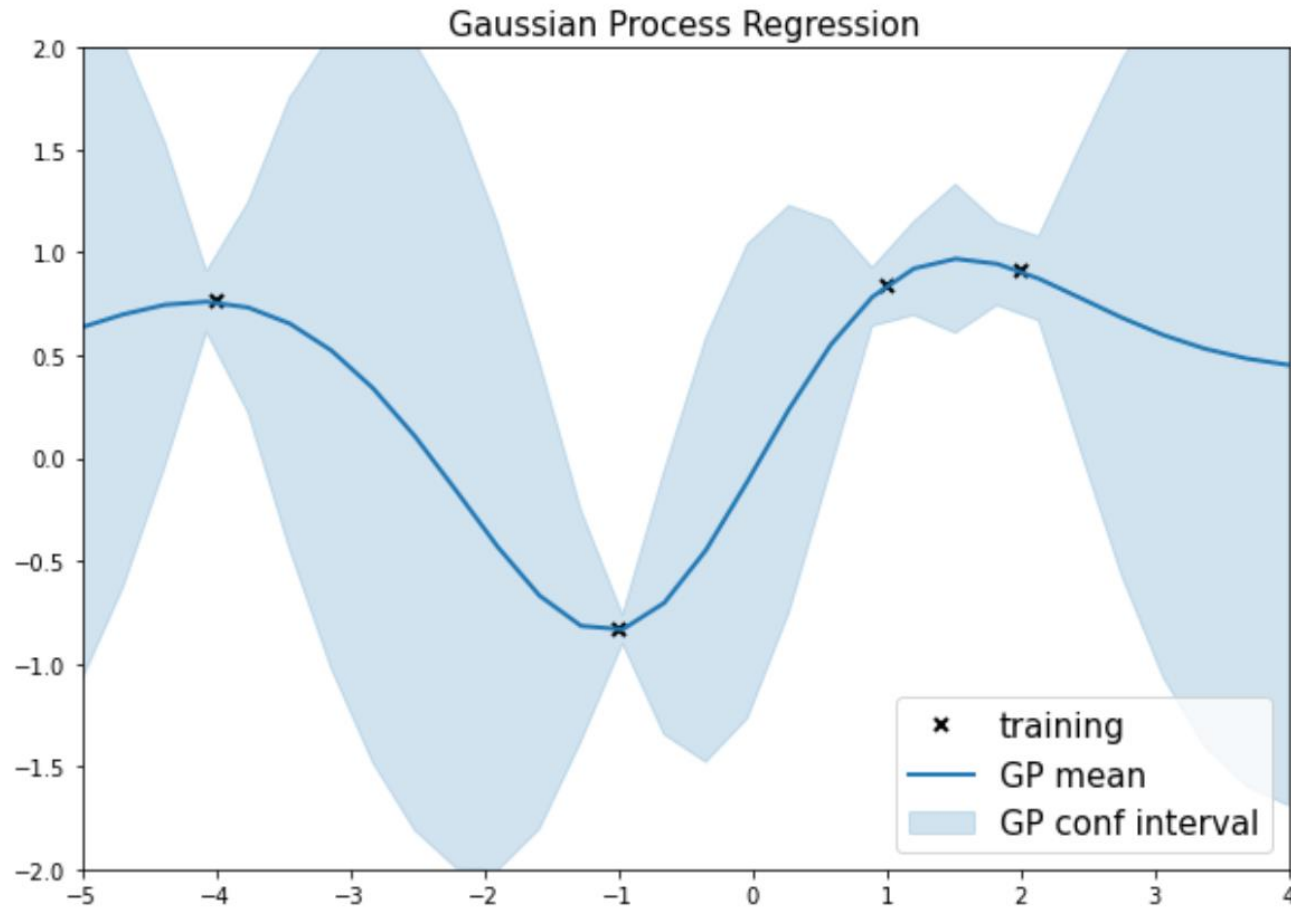
$$\hat{y}_{\text{pred}}^{(2)} \sim \mathcal{N}(\mu_y(x_{\text{new}}^{(2)}), \sigma_y^2(x_{\text{new}}^{(2)}))$$

Mean and covariance functions



$$\hat{y}_{\text{pred}}^{(3)} \sim \mathcal{N}(\mu_y(x_{\text{new}}^{(3)}), \sigma_y^2(x_{\text{new}}^{(3)}))$$

Mean and covariance functions



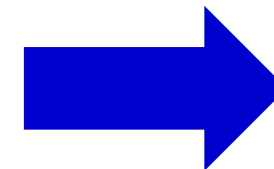
$$\hat{y}_{\text{pred}}(x) \sim \mathcal{N}(\mu_y(x), \sigma_y^2(x))$$

We can define these predictions via functions:

$$\mu_y(x) = \mu_y(m(x), K(x, \cdot))$$

$$\sigma_y^2(x) = \sigma_y^2(m(x), K(x, \cdot))$$

$m(x)$ is a prior for the mean



**More will be
taught by Austin!**

PyMC: A probabilistic modelling toolbox

Simple PyMC example code

```
import pymc as pm

with pm.Model():
    k = pm.Lognormal("k", mu=0, sigma=1)
    pred = 5 * (1 - pm.math.exp(-k * t))
    pm.Normal("likelihood", mu=pred, observed=P_obs)
```

Key syntax you'll use:

- `pm.Model` - define model
- `pm.HalfNormal` - priors
- `pm.gp.Latent` - Gaussian process
- `pm.sample` - run MCMC
- `pm.sample_posterior_predictive` - predictions



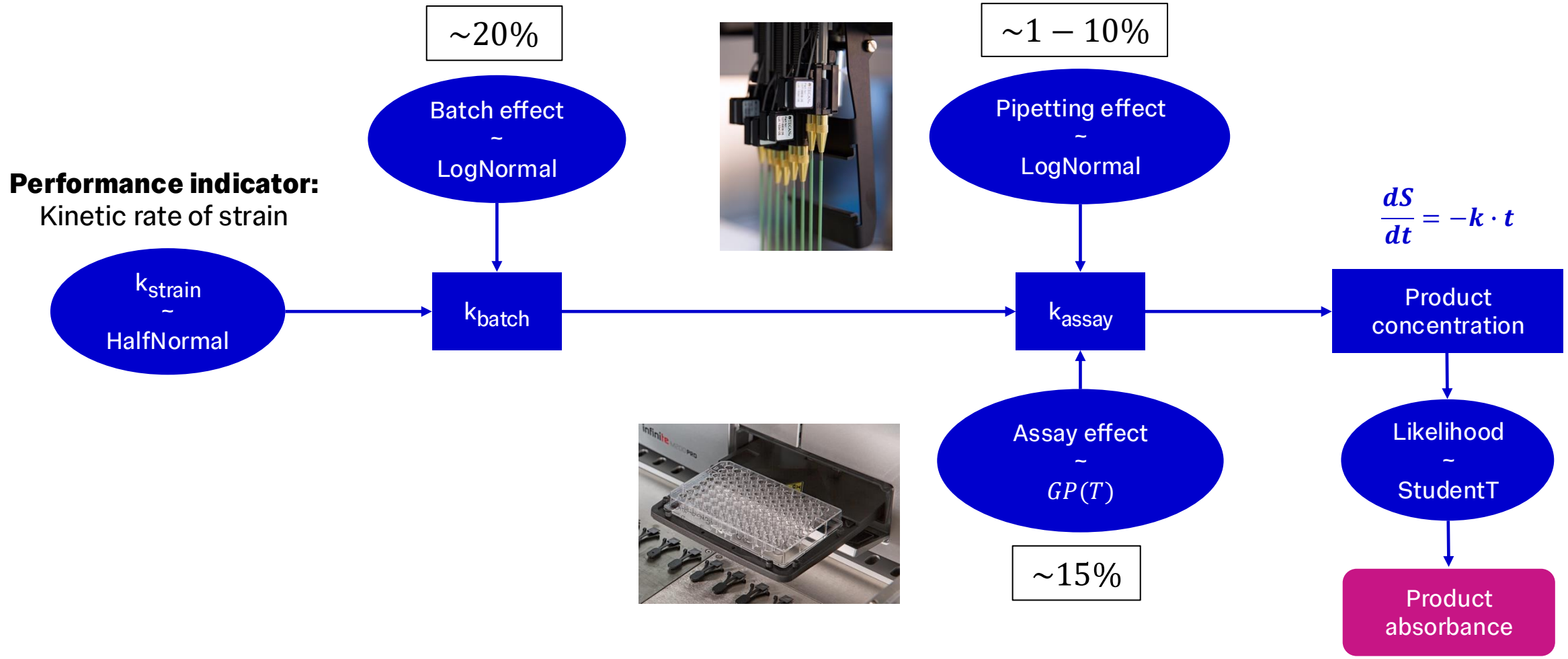
Takes care of...

... sampling from the posterior.

... model diagnostics and
visualisation (via ArviZ).

... flexible modelling (GPs,
hierarchical models, mixtures).

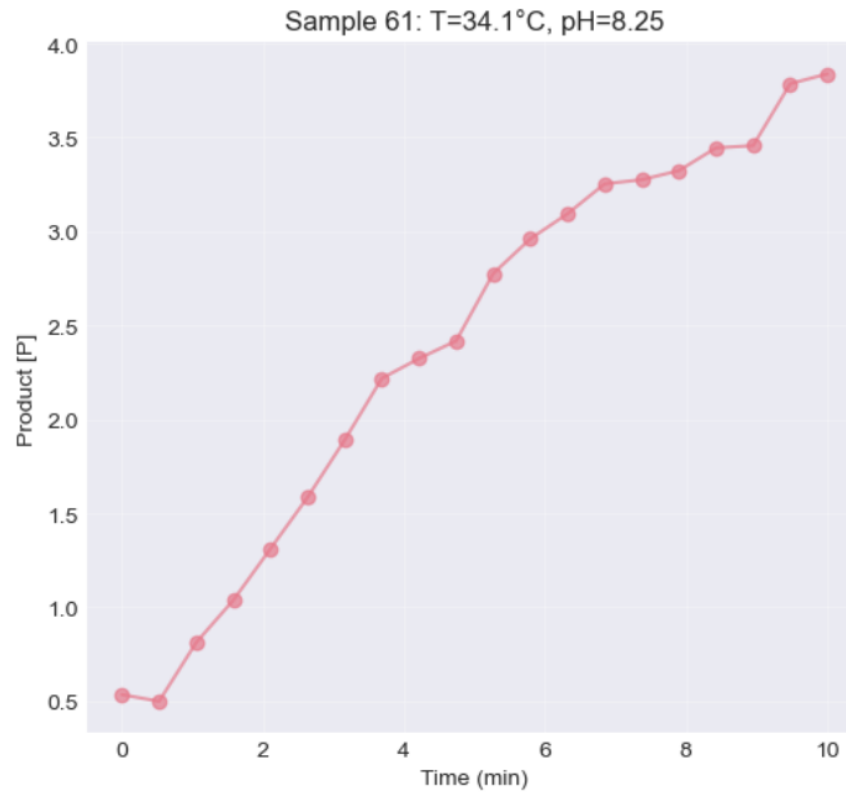
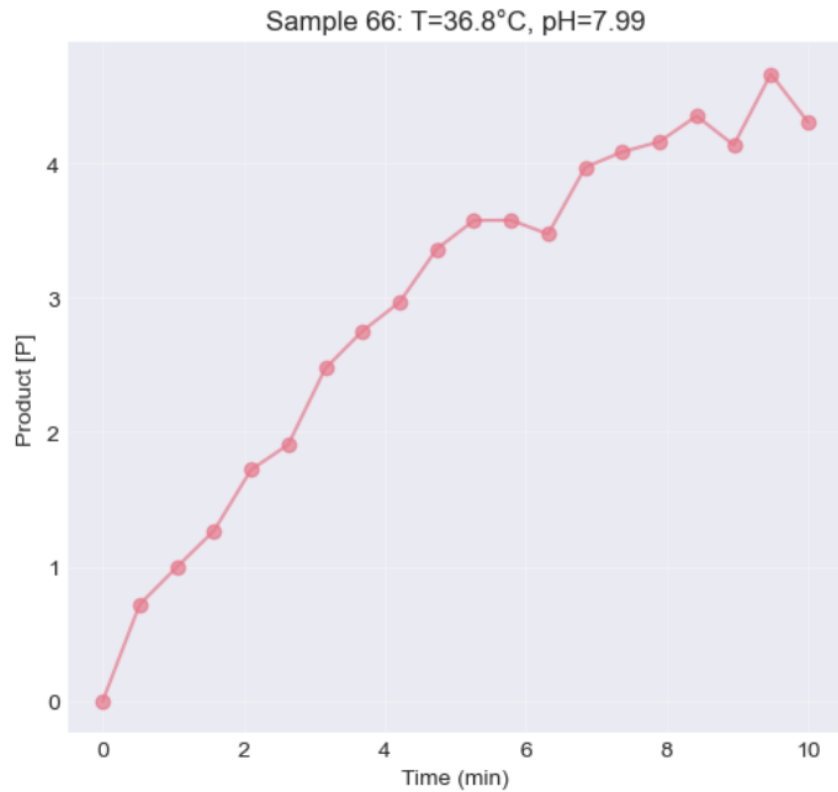
Exemplary hybrid process model



Group work

Hybrid models using Gaussian processes

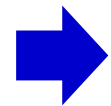
Let's try it in practice



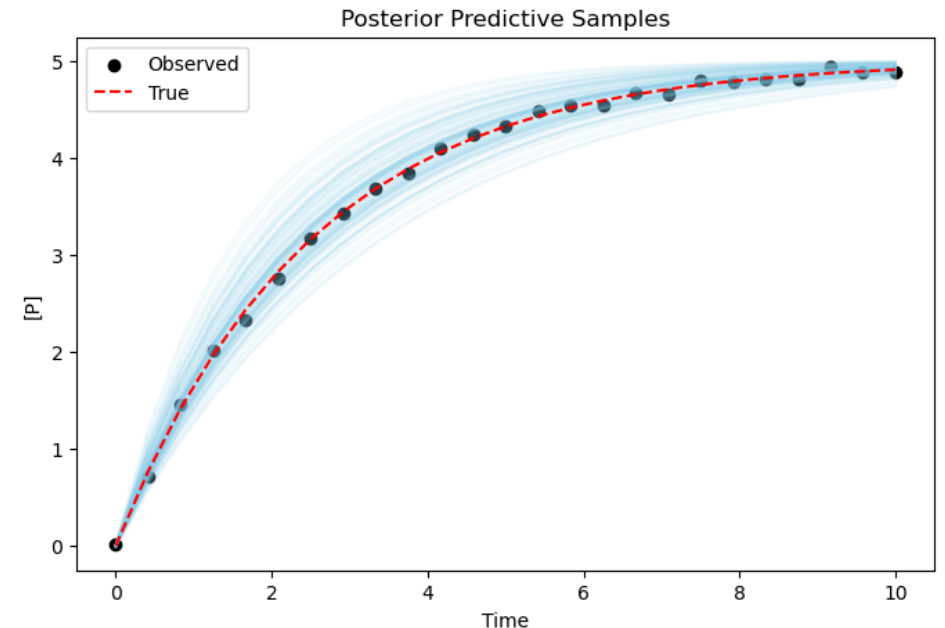
- Analyse **more complex kinetic data**
- Build **hybrid model** to fit the kinetic data
- Check **MCMC diagnostics**
- Plot **posterior and posterior predictive** to understand your model

Summary

- **Bayesian models** are useful for biological applications
- They allow to use **mechanistic knowledge** and combine it with **data-driven approaches**
- **Uncertainty quantification** is very important in real-world applications





Posterior (predictive) distributions can be used for **Bayesian optimisation** for experimental design



IMPERIAL

Thank you

AIMS Teaching Session
17/02/2026

 I.helleckes@imperial.ac.uk
 [@lhelleckes/@JuBiotech](#)