Gaussian Process and Determinantal Point Process

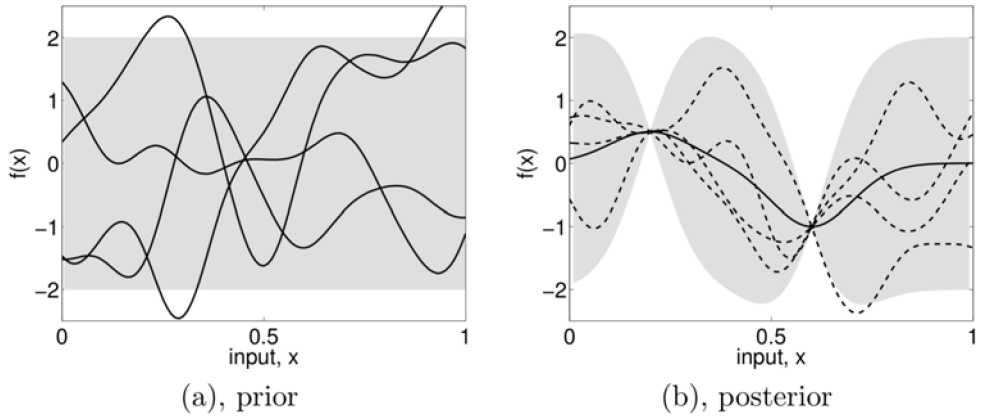
Shichang Zhang

04/29/2021

Slides adapted from Eric Xing, Richard Turner, and Pengtao Xie

Outline

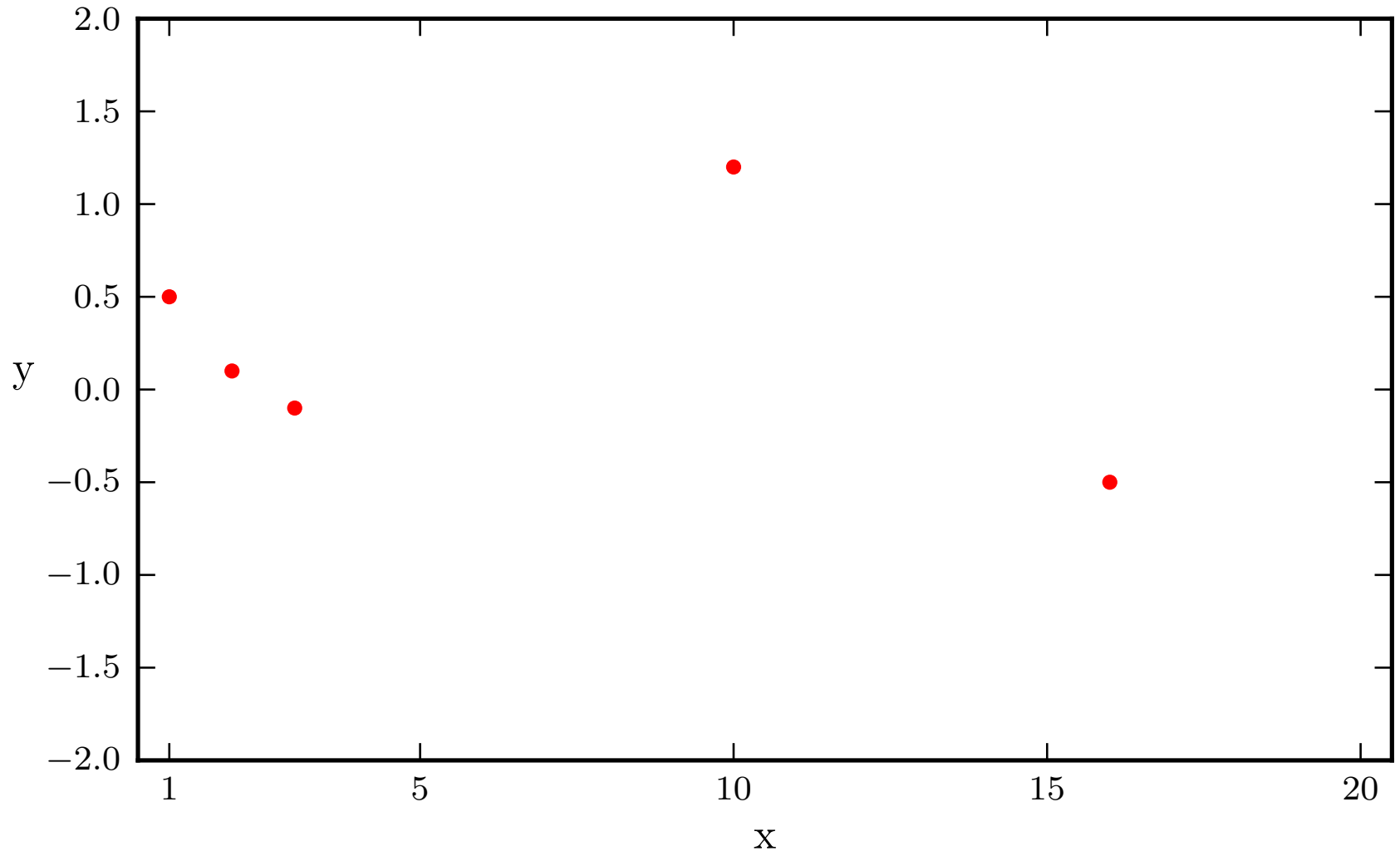
- Part 1: Gaussian Processes (GPs)
  - Motivating example of GPs
  - Parametric model and non-parametric model
  - Definition of GPs
  - Inference and learning with GPs
  - Deep kernel learning
  - Graph Convolutional Gaussian Processes (ICML 2019)
- Part 2: Determinantal Point Process (DPP)



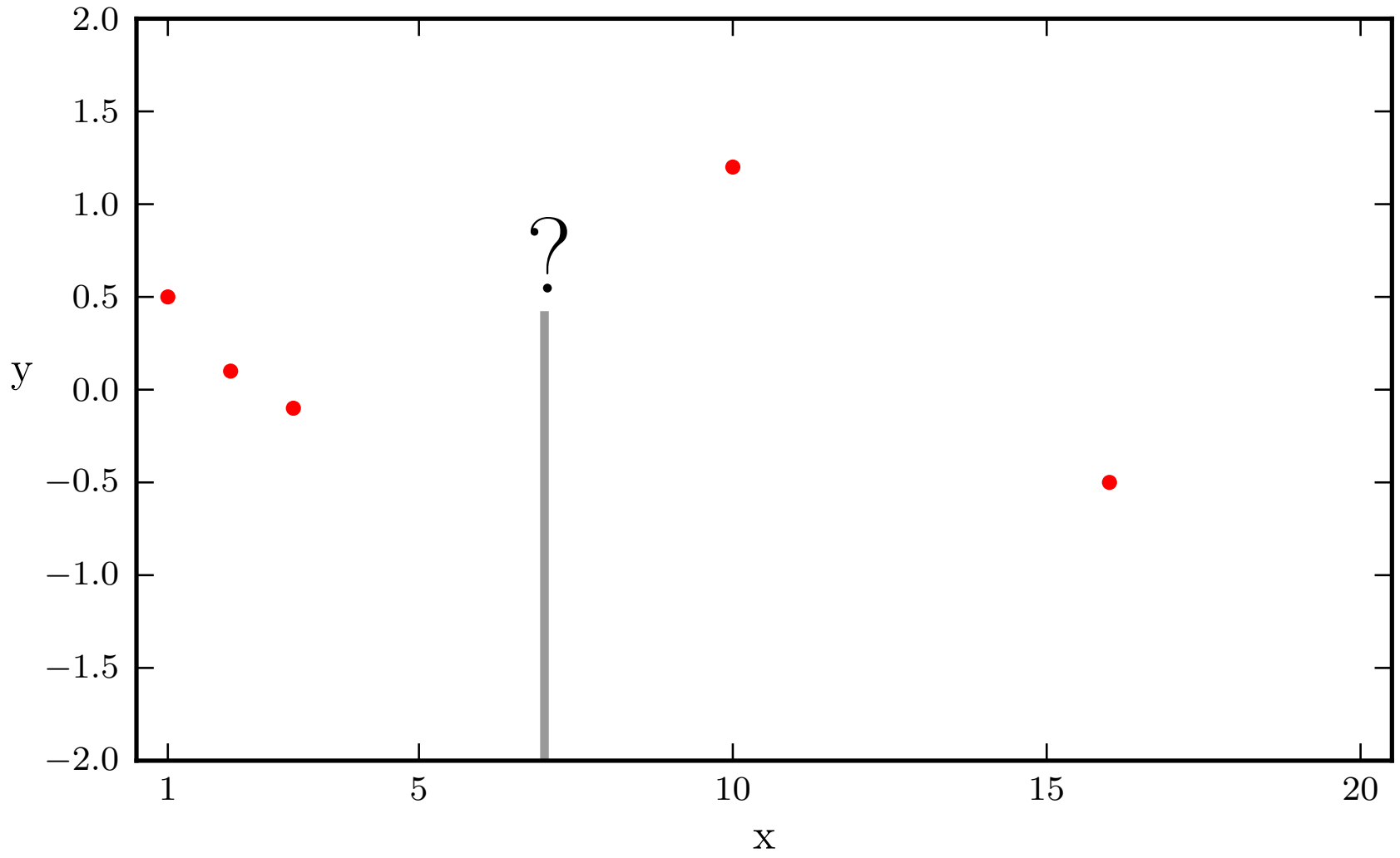
Walker, I., & Glocker, B. (2019, May). Graph convolutional Gaussian processes. In *International Conference on Machine Learning* (pp. 6495-6504). PMLR.

Gaussian Process (GP)

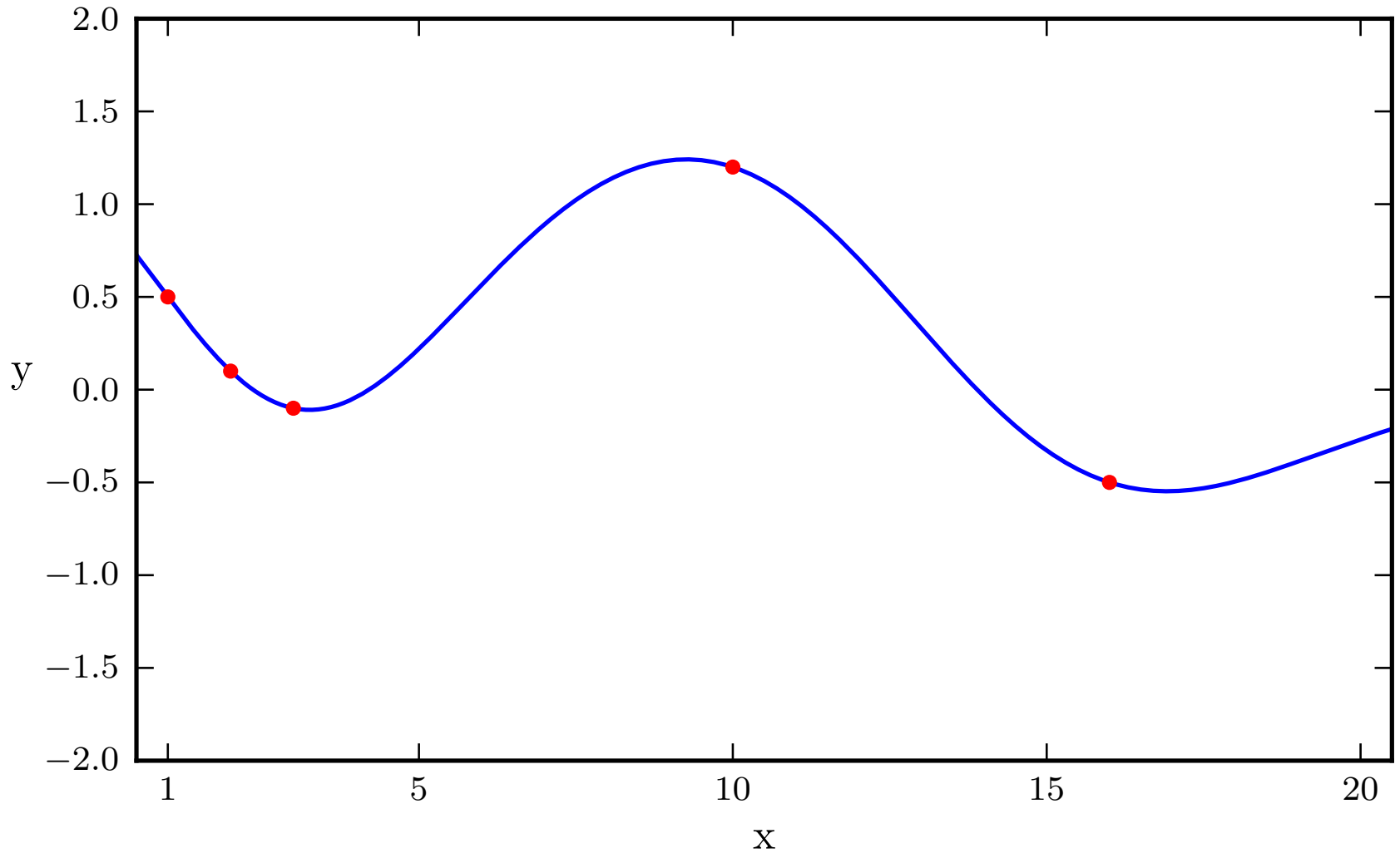
Motivation: non-linear regression



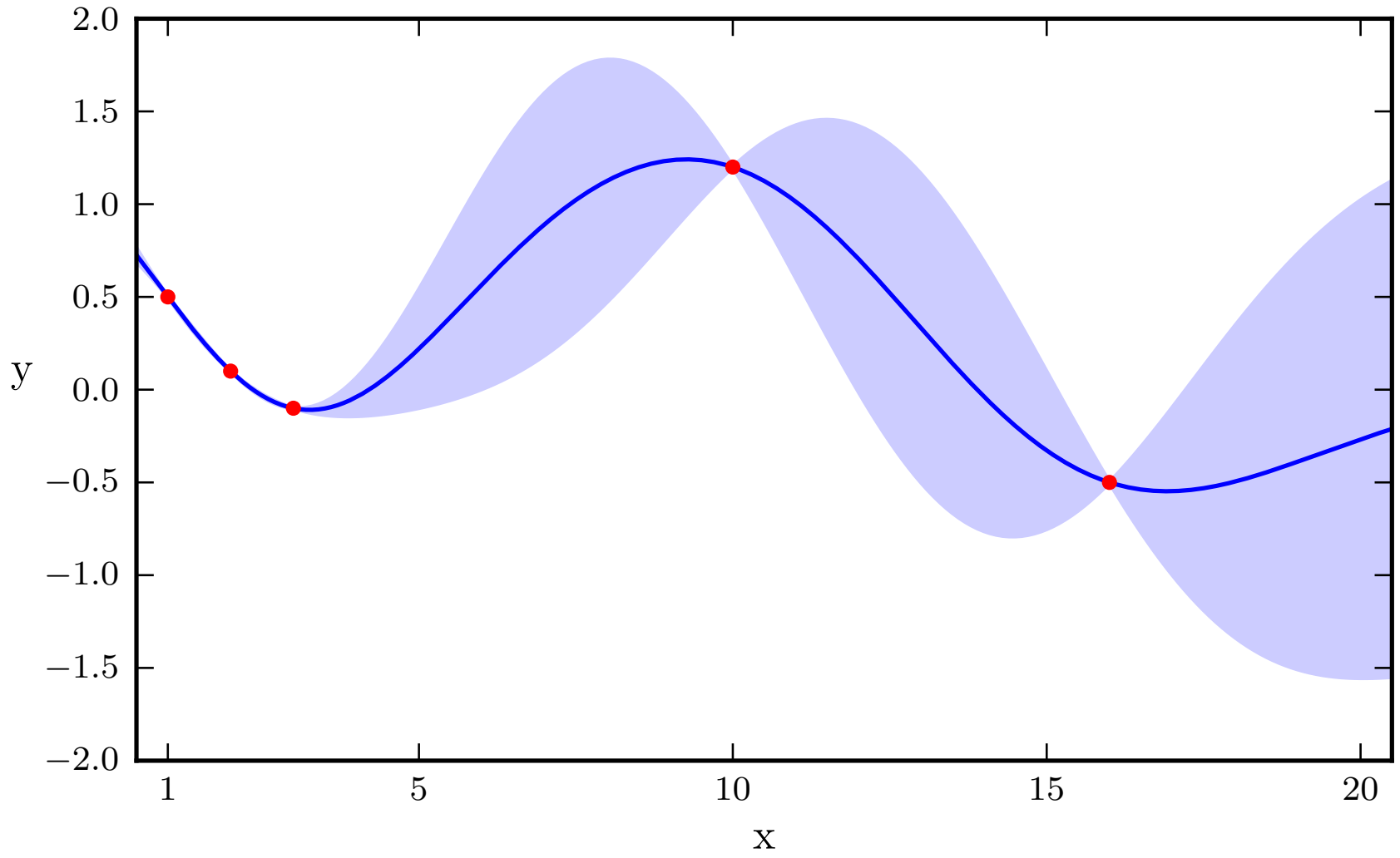
Motivation: non-linear regression



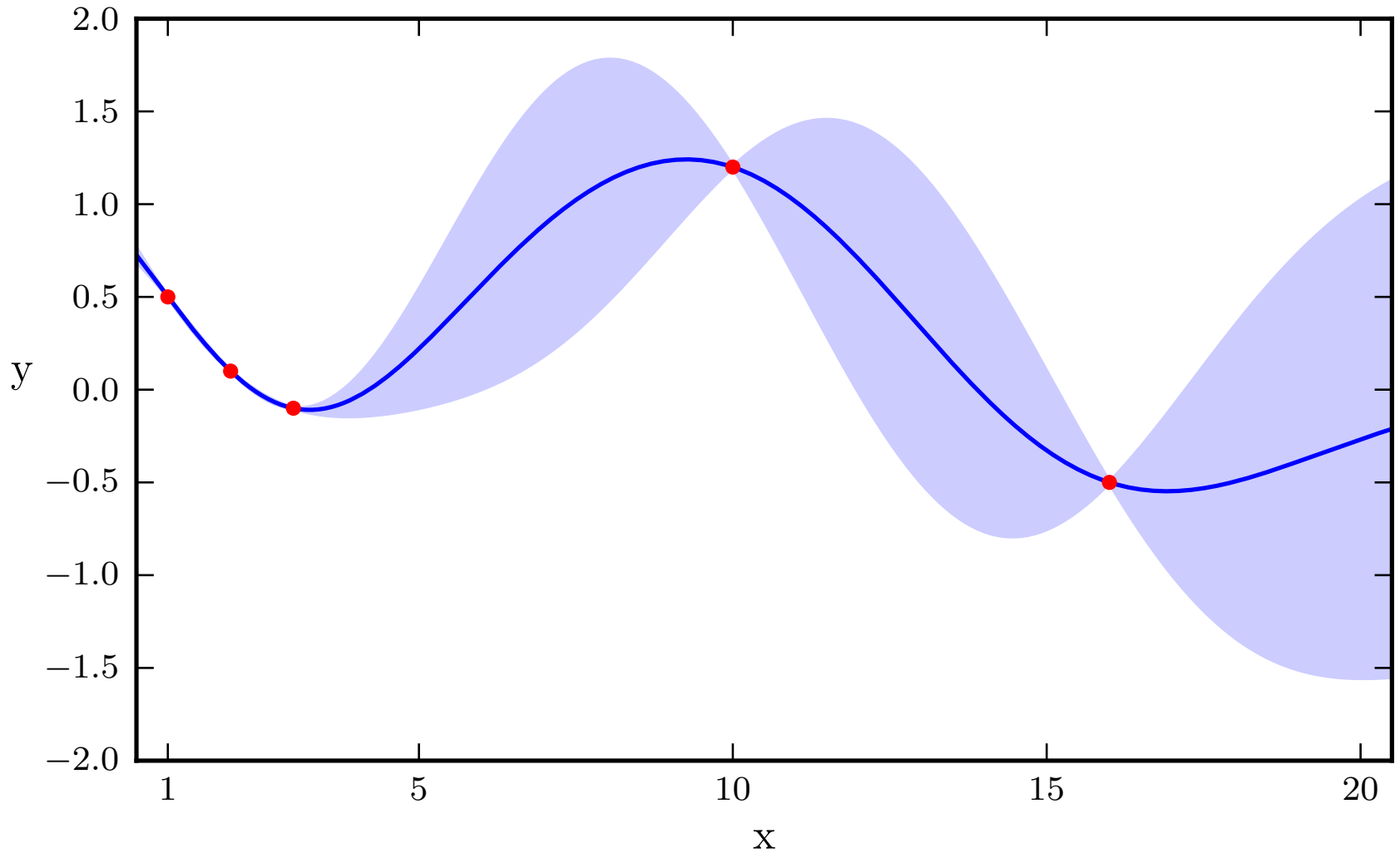
Motivation: non-linear regression



Motivation: non-linear regression



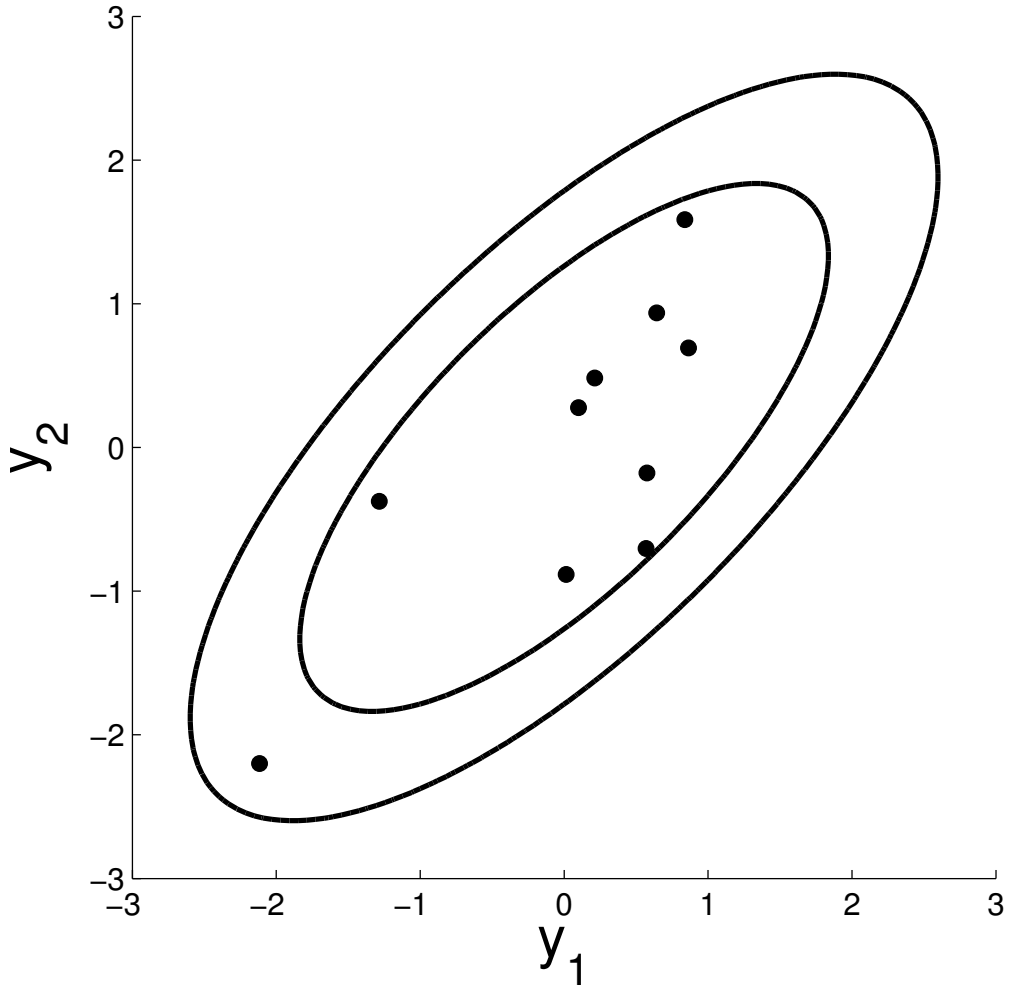
Motivation: non-linear regression



Can we do this with a plain old Gaussian?

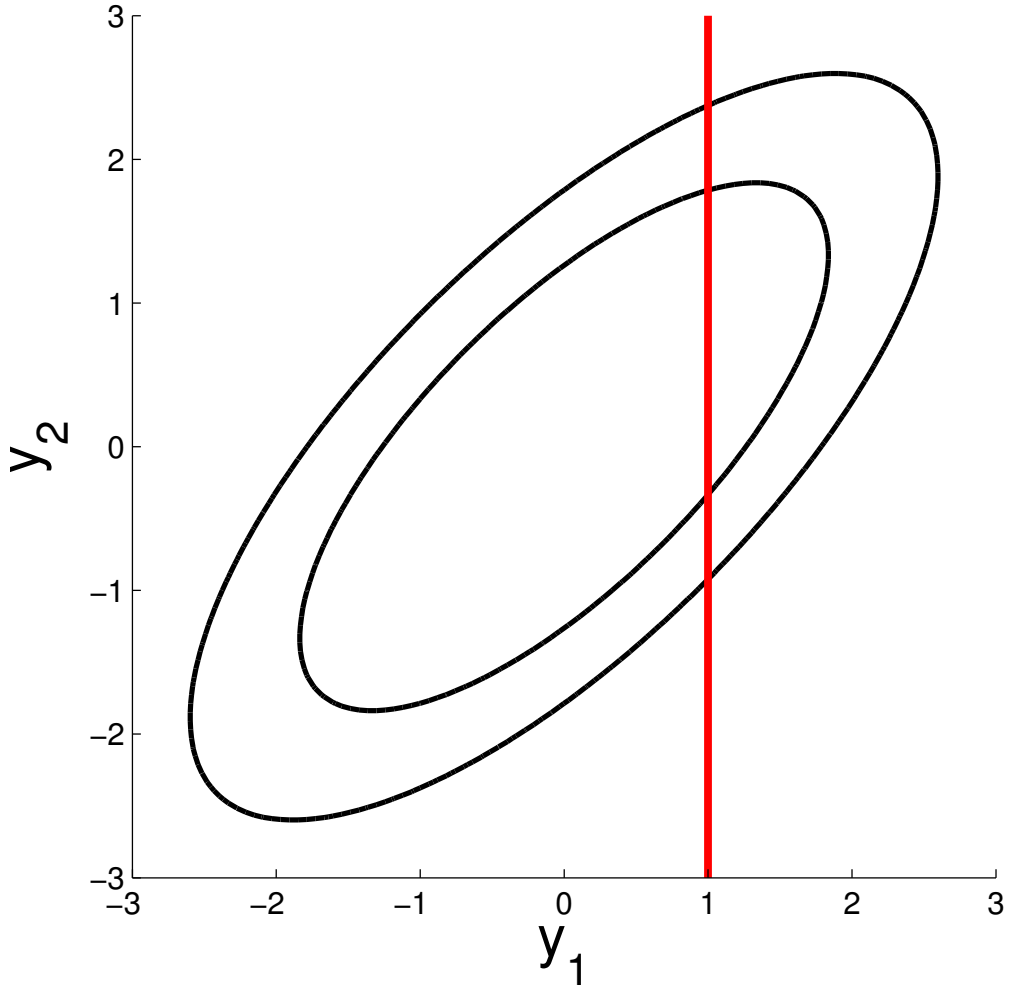
Gaussian distribution

$$p(\mathbf{y}|\Sigma) \propto \exp\left(-\tfrac{1}{2}\mathbf{y}^\intercal\Sigma^{-1}\mathbf{y}\right) \qquad \Sigma = \begin{bmatrix} 1 & .7 \\ .7 & 1 \end{bmatrix}$$



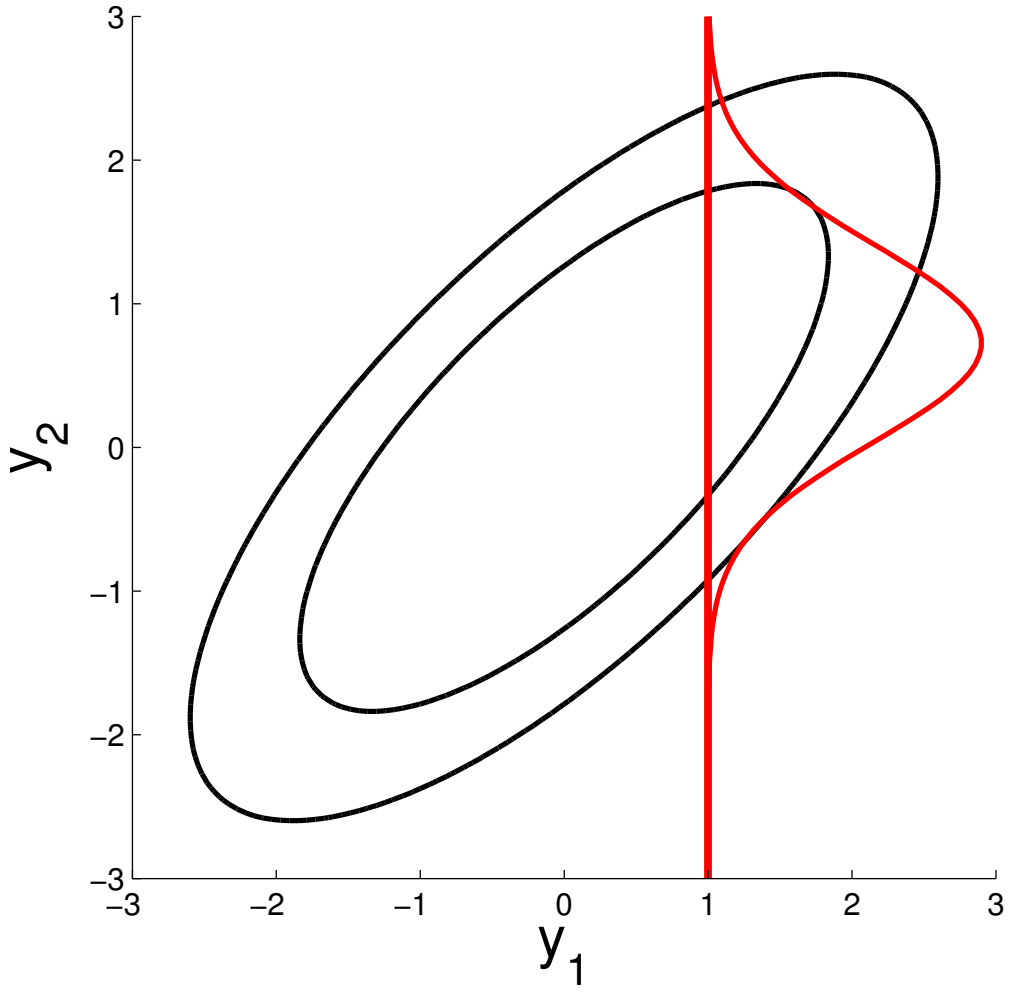
Gaussian distribution

$$p(\mathbf{y}|\Sigma) \propto \exp\left(-\tfrac{1}{2}\mathbf{y}^\intercal\Sigma^{-1}\mathbf{y}\right) \qquad \Sigma = \begin{bmatrix} 1 & .7 \\ .7 & 1 \end{bmatrix}$$



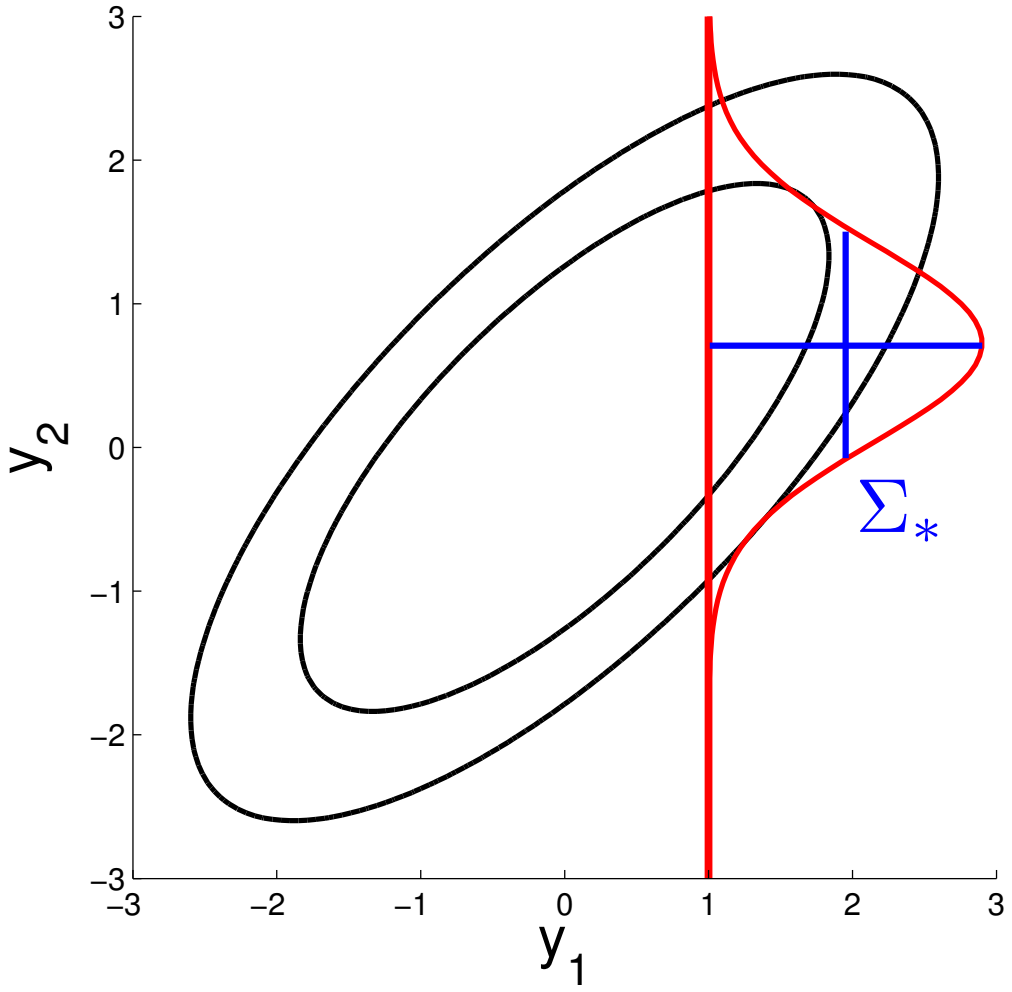
Gaussian distribution

$$p(\mathrm{y}_2|\mathrm{y}_1, \Sigma) \propto \exp\left(-\tfrac{1}{2}(\mathrm{y}_2 - \mu_*)\Sigma_*{}^{-1}(\mathrm{y}_2 - \mu_*)\right)$$



Gaussian distribution

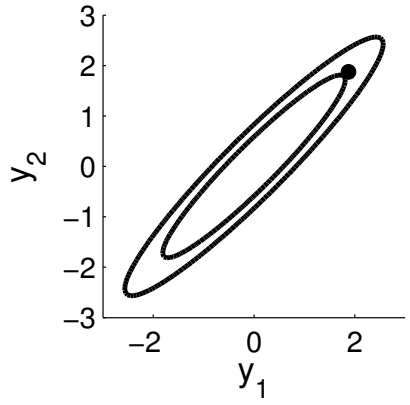
$$p(\mathrm{y}_2|\mathrm{y}_1, \Sigma) \propto \exp\left(-\tfrac{1}{2}(\mathrm{y}_2 - \mu_*)\Sigma_*{}^{-1}(\mathrm{y}_2 - \mu_*)\right)$$



$\mu_* = \mathrm{W}\mathrm{y}_1$

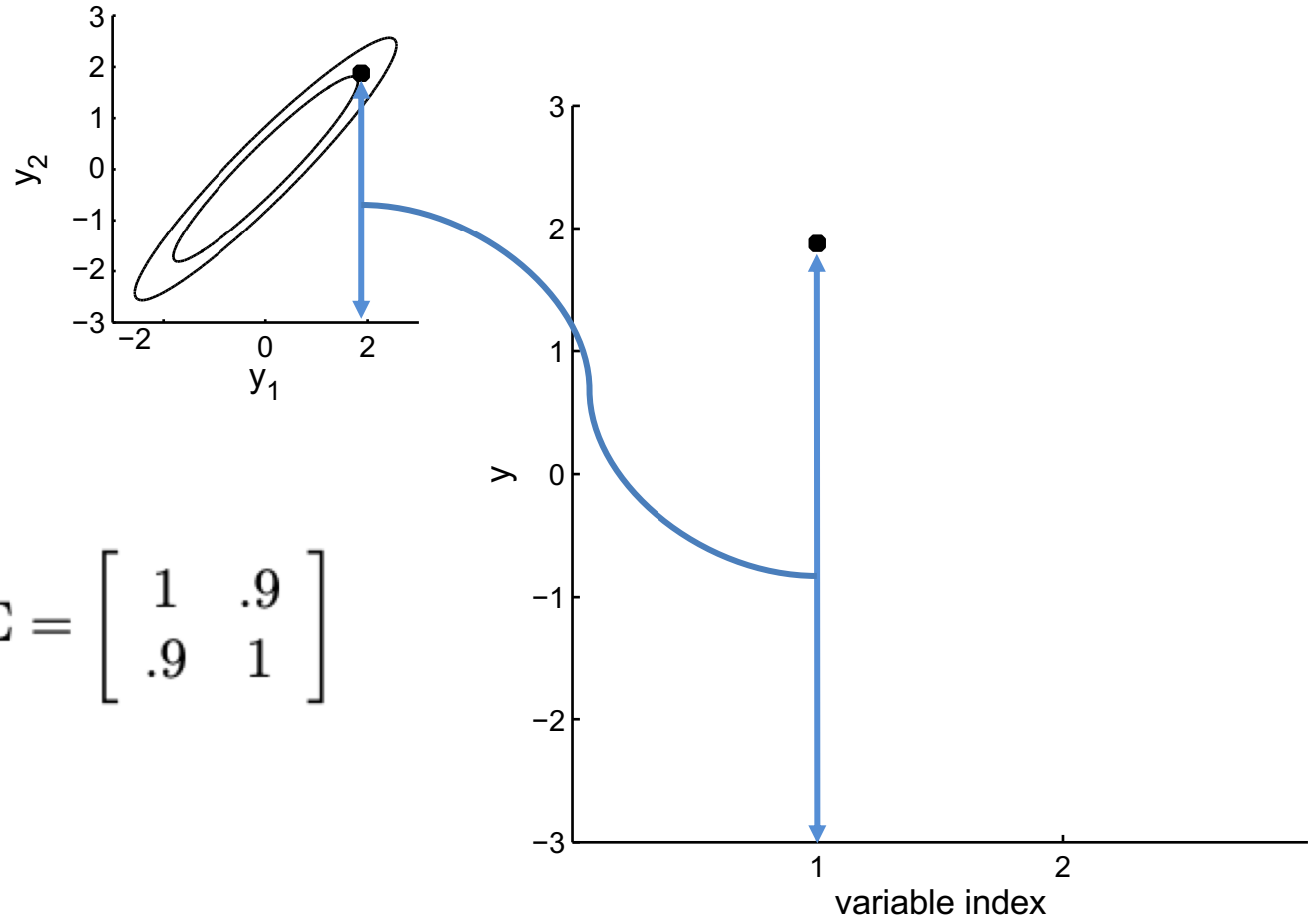
Σ_*

New visualisation



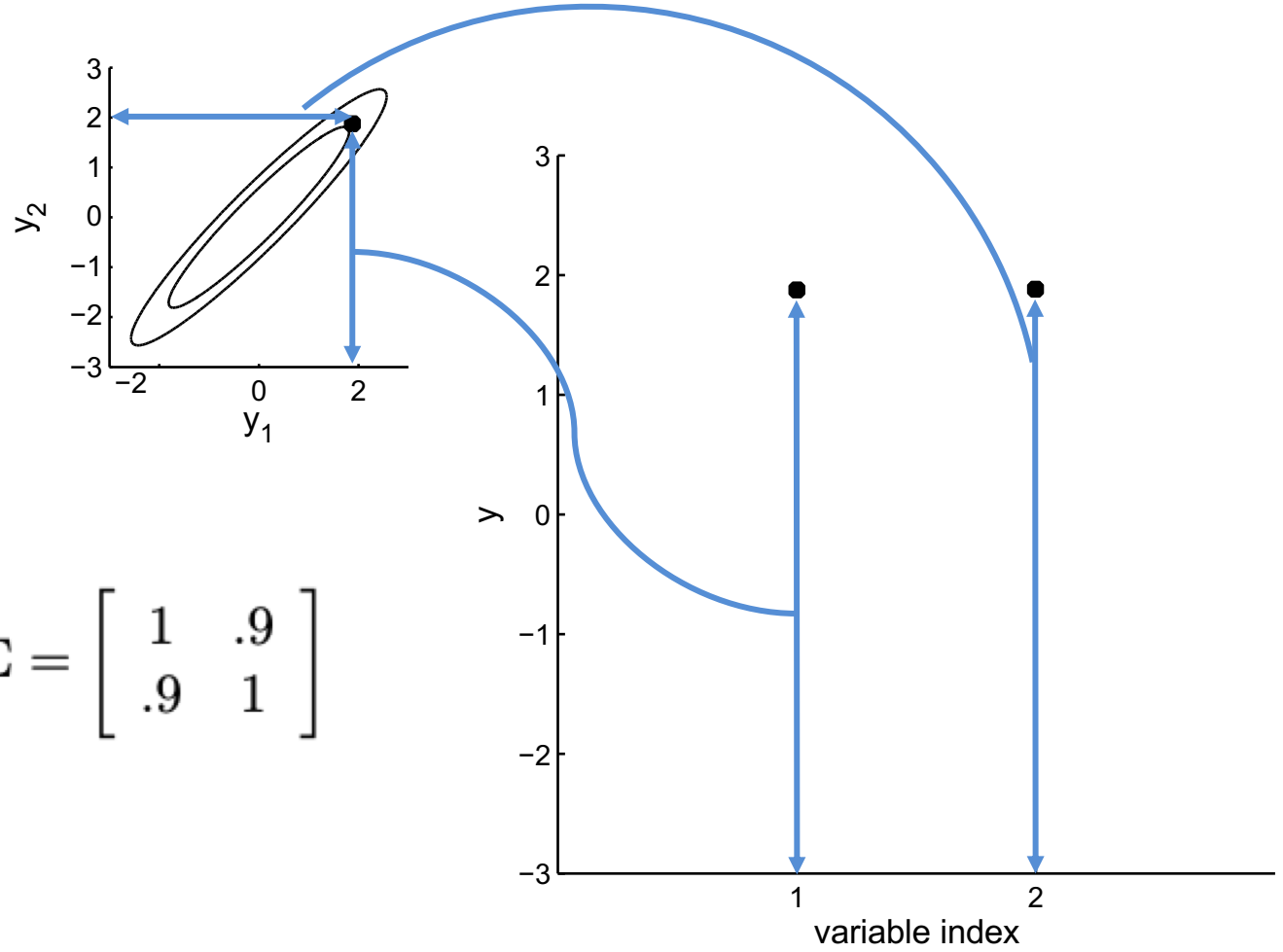
$$\Sigma = \begin{bmatrix} 1 & .9 \\ .9 & 1 \end{bmatrix}$$

New visualisation



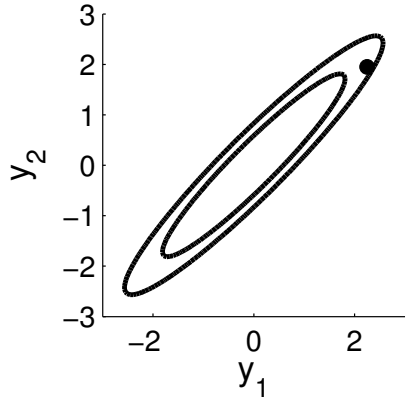
$$\Sigma = \begin{bmatrix} 1 & .9 \\ .9 & 1 \end{bmatrix}$$

New visualisation

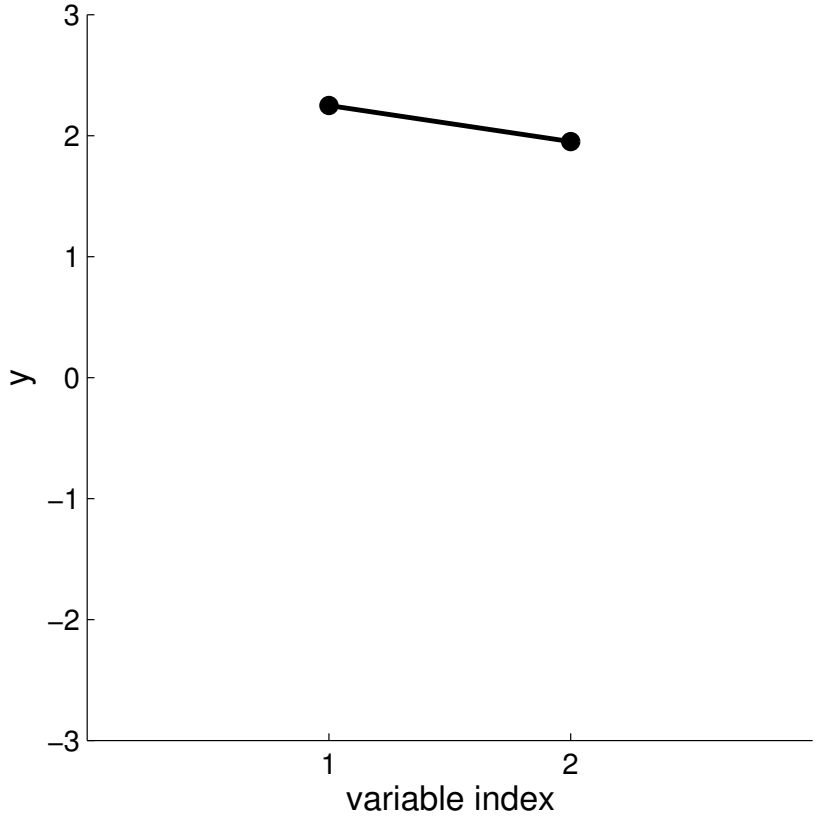


$$\Sigma = \begin{bmatrix} 1 & .9 \\ .9 & 1 \end{bmatrix}$$

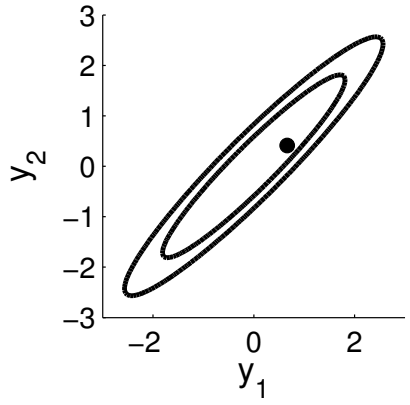
New visualisation



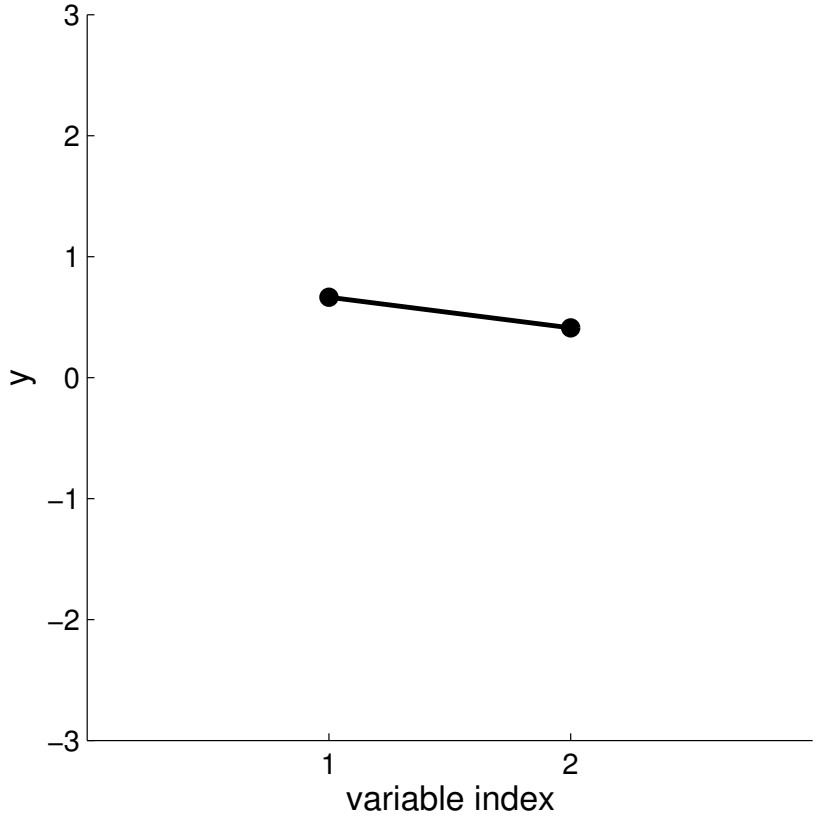
$$\Sigma = \begin{bmatrix} 1 & .9 \\ .9 & 1 \end{bmatrix}$$



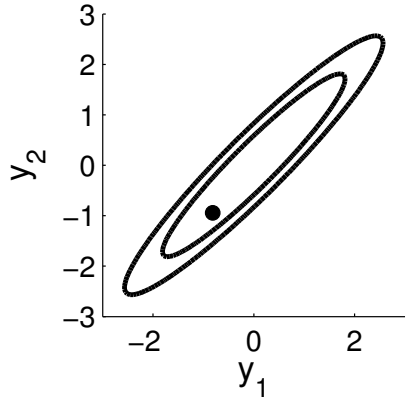
New visualisation



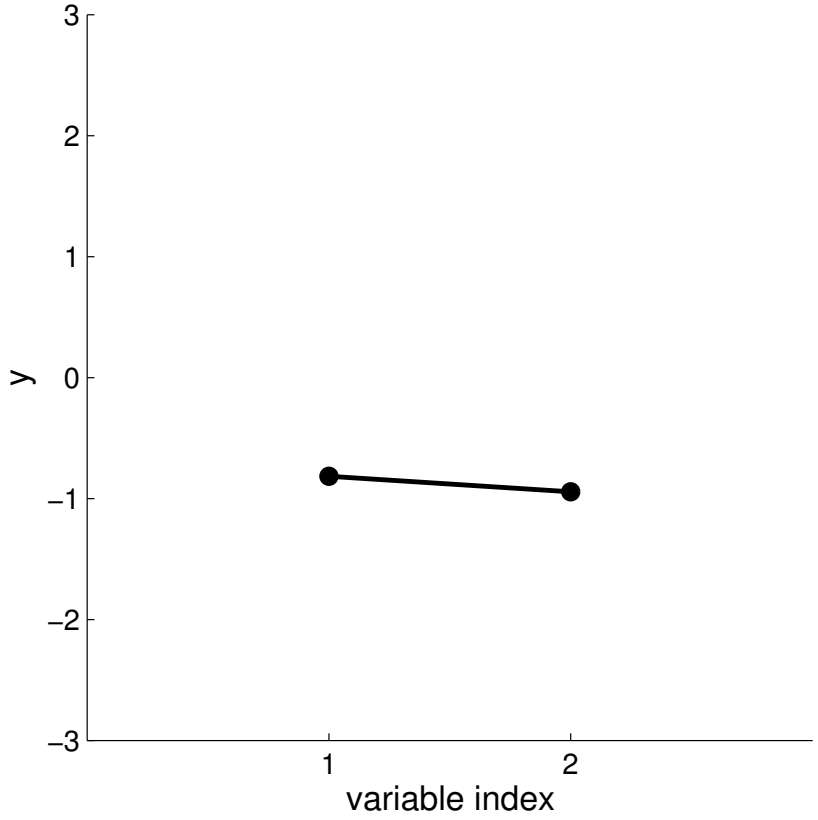
$$\Sigma = \begin{bmatrix} 1 & .9 \\ .9 & 1 \end{bmatrix}$$



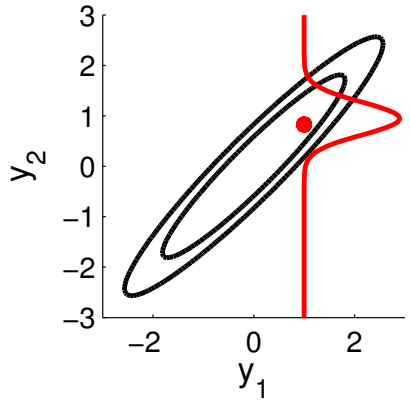
New visualisation



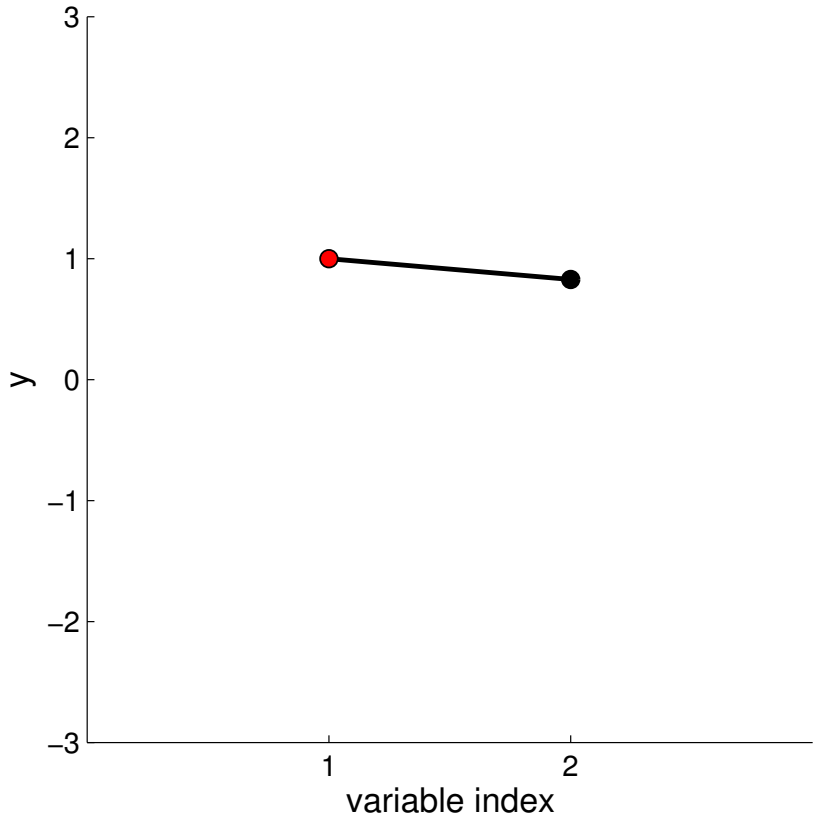
$$\Sigma = \begin{bmatrix} 1 & .9 \\ .9 & 1 \end{bmatrix}$$



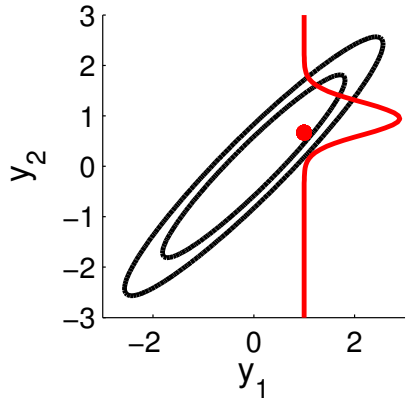
New visualisation



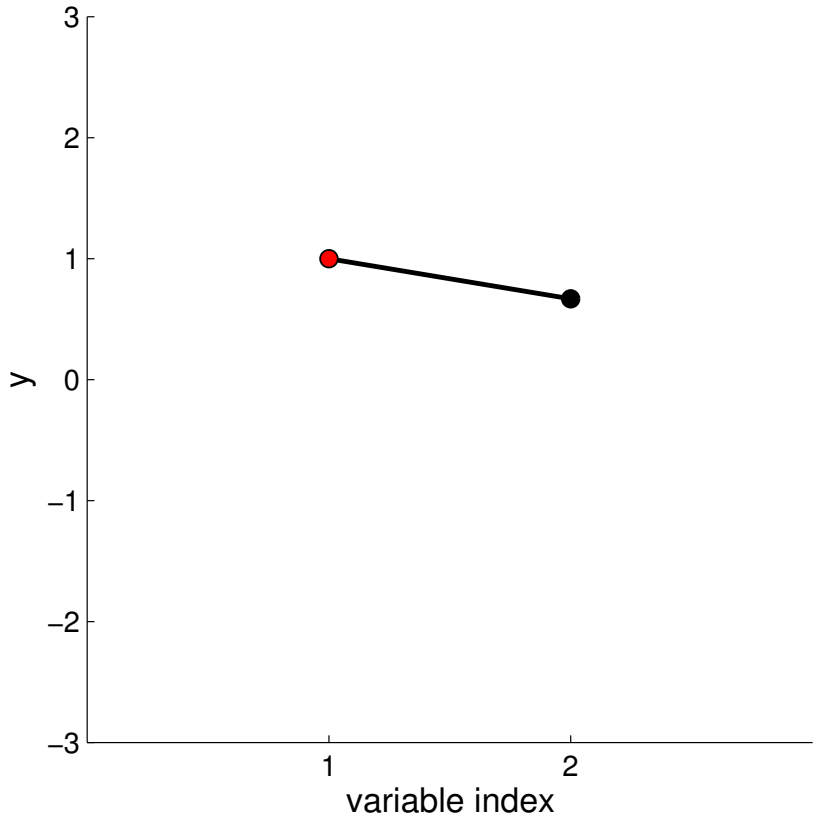
$$\Sigma = \begin{bmatrix} 1 & .9 \\ .9 & 1 \end{bmatrix}$$



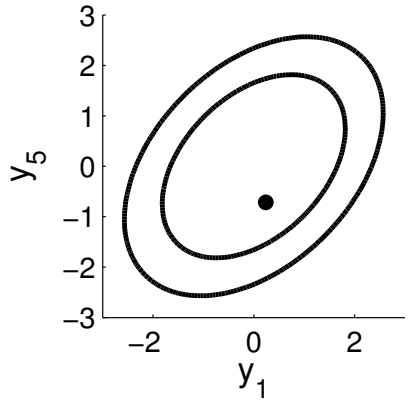
New visualisation

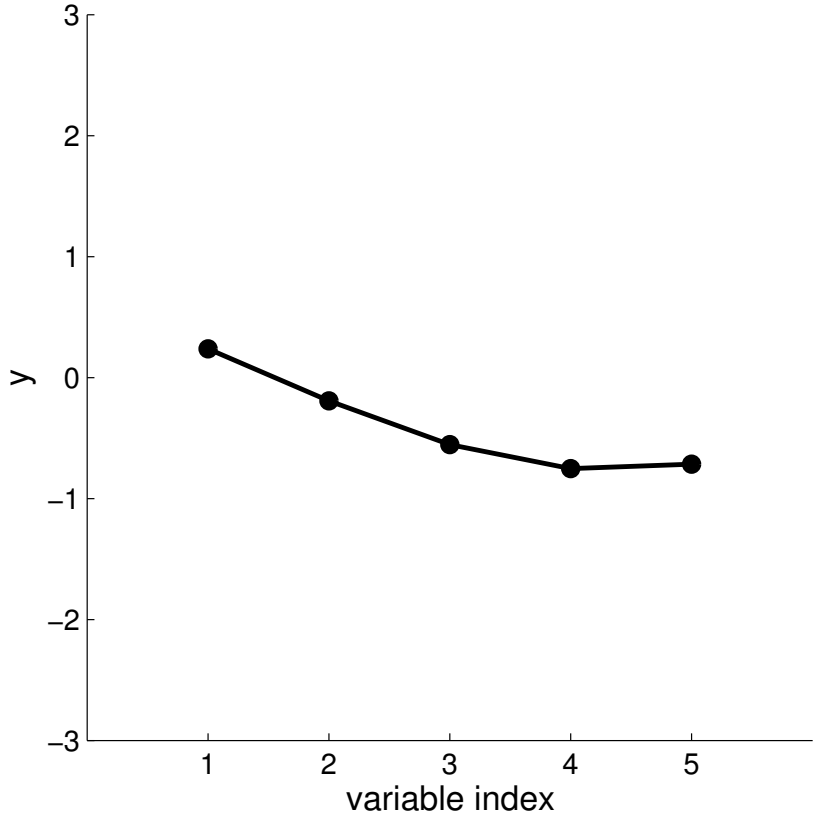


$$\Sigma = \begin{bmatrix} 1 & .9 \\ .9 & 1 \end{bmatrix}$$



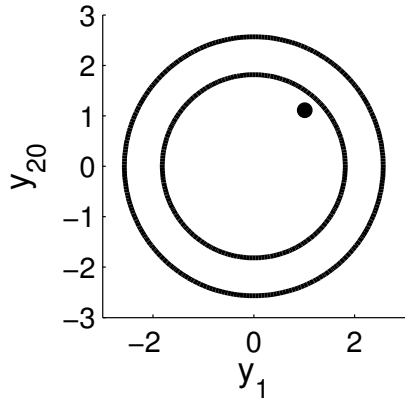
New visualisation

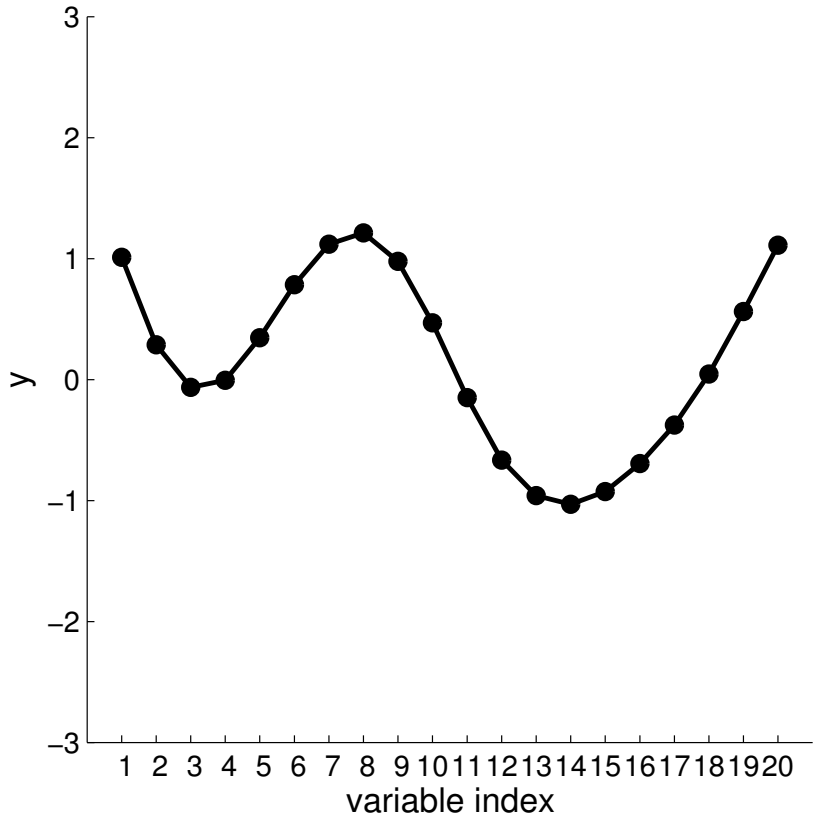




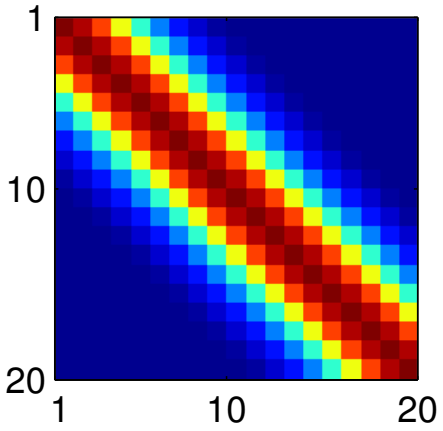
$$\Sigma = \begin{bmatrix} 1 & .9 & .8 & .6 & .4 \\ .9 & 1 & .9 & .8 & .6 \\ .8 & .9 & 1 & .9 & .8 \\ .6 & .8 & .9 & 1 & .9 \\ .4 & .6 & .8 & .9 & 1 \end{bmatrix}$$

New visualisation

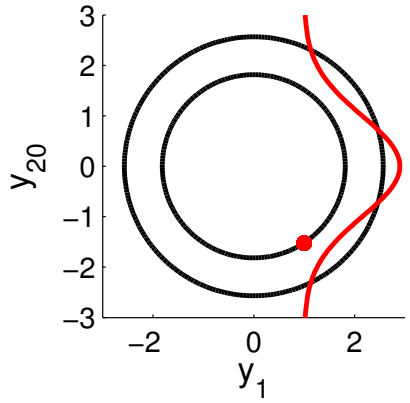


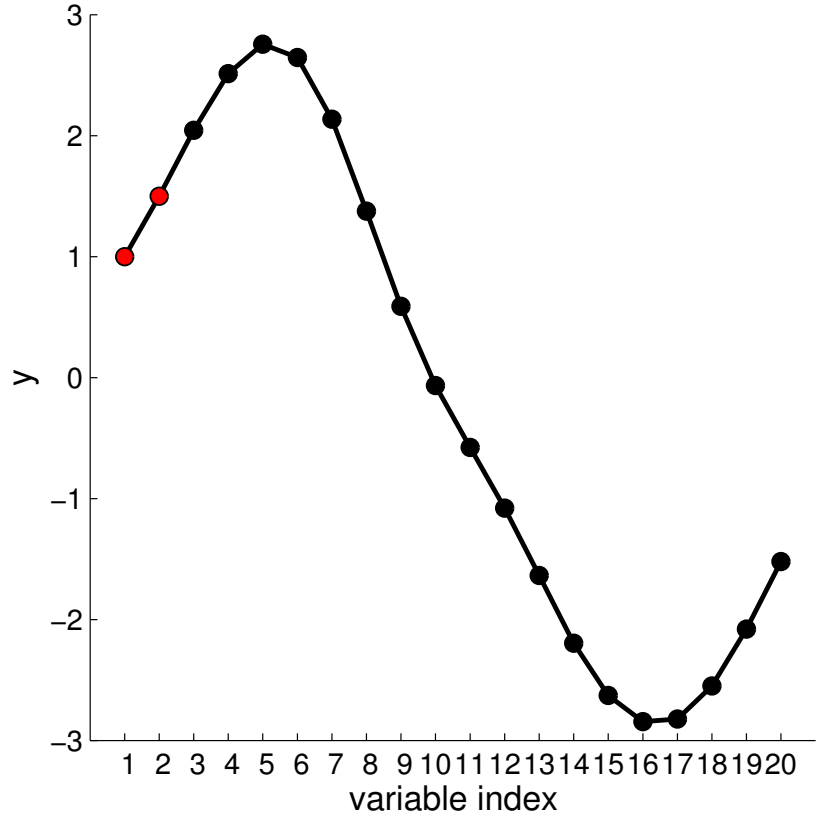


$\Sigma =$

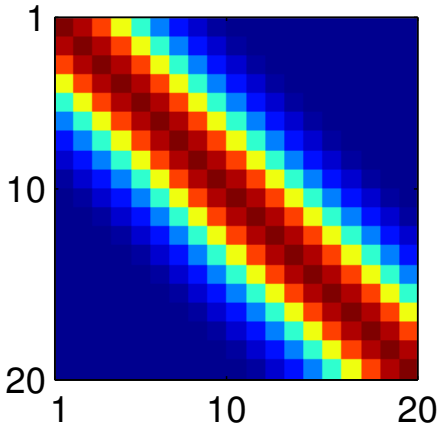


New visualisation



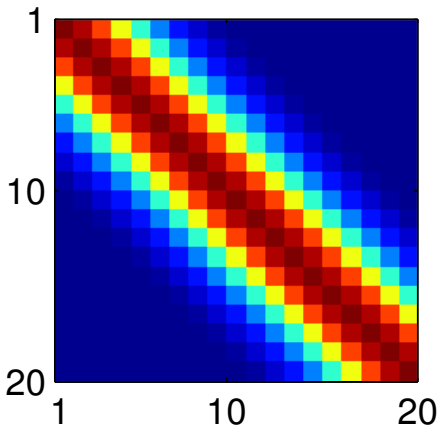


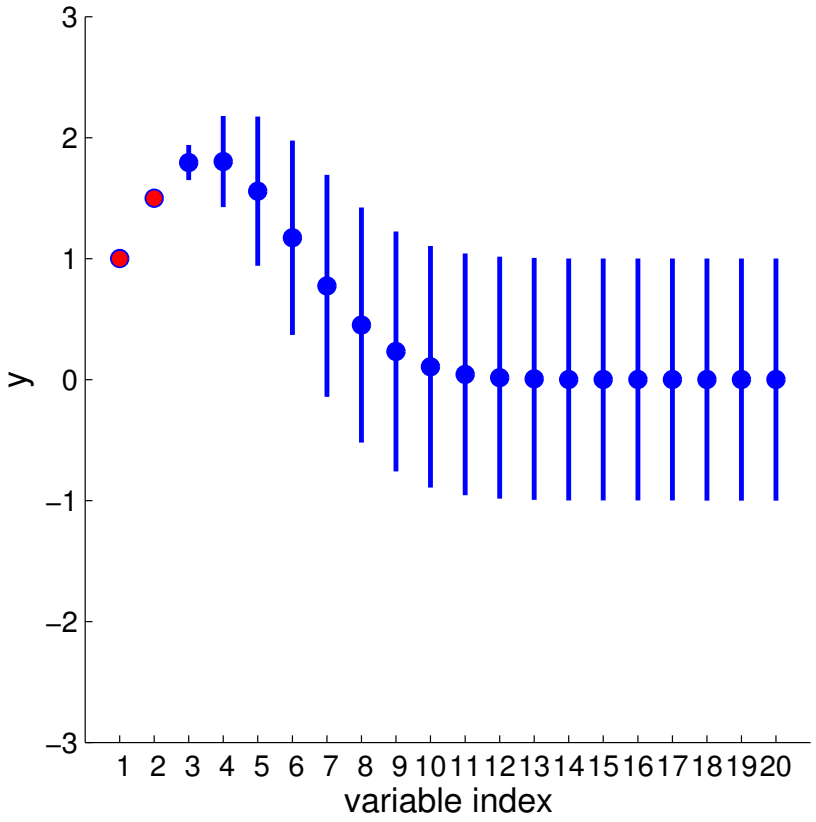
$\Sigma =$



Regression using Gaussians

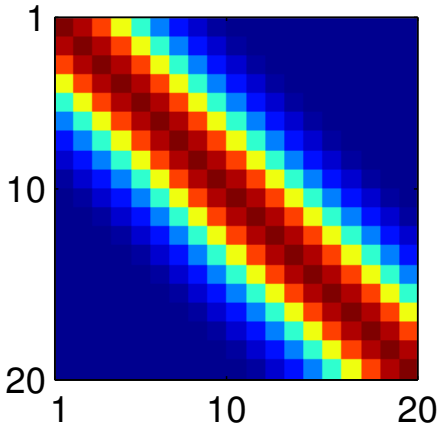
$\Sigma =$

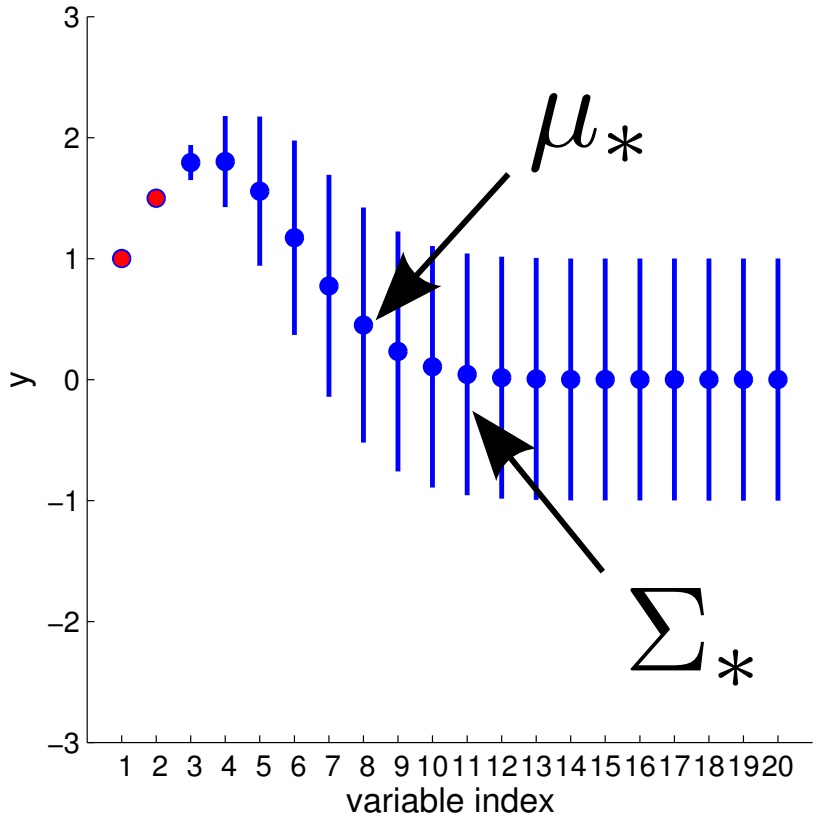




Regression using Gaussians

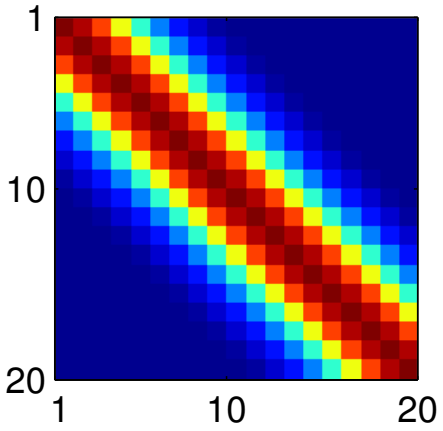
$\Sigma =$

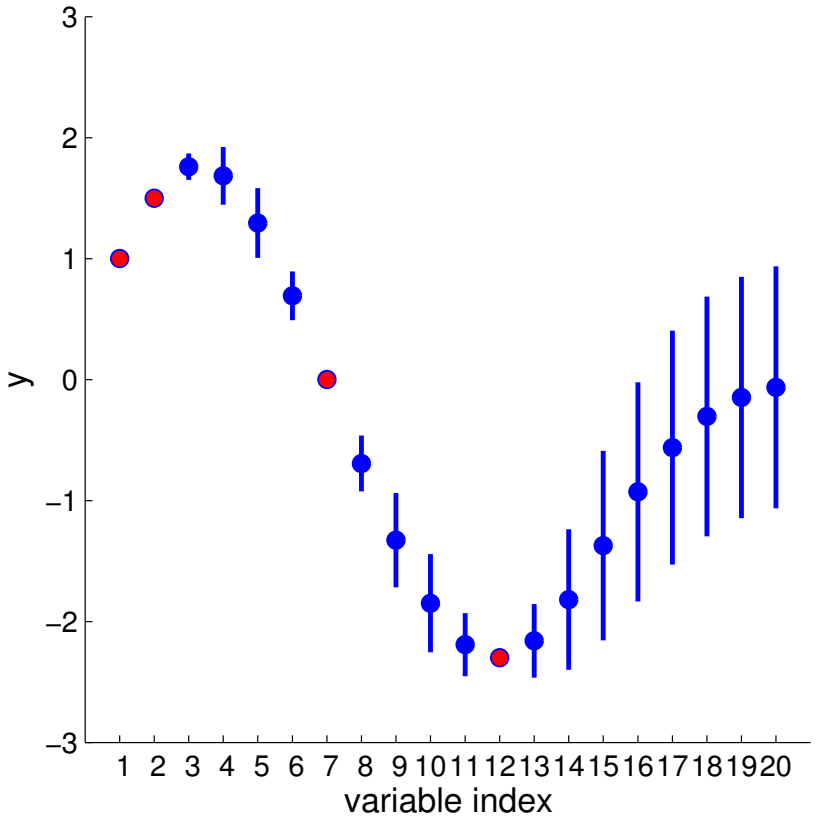




Regression using Gaussians

$\Sigma =$



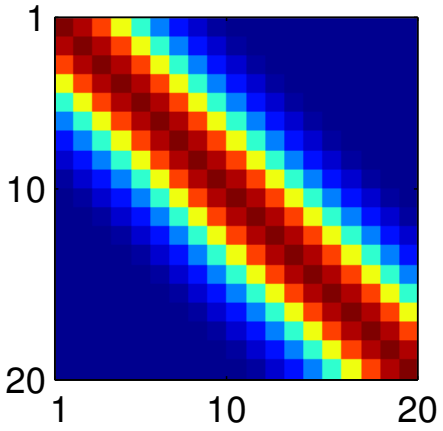


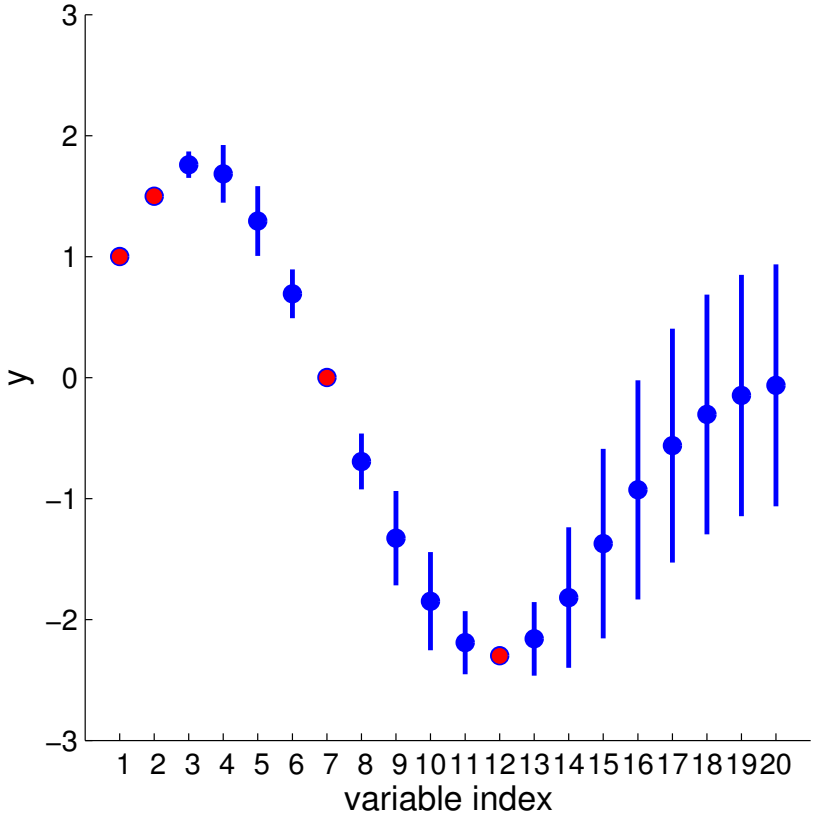
Regression using Gaussians

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

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

$\Sigma =$



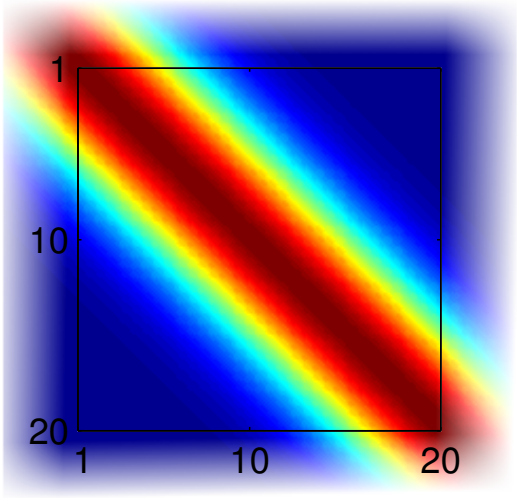


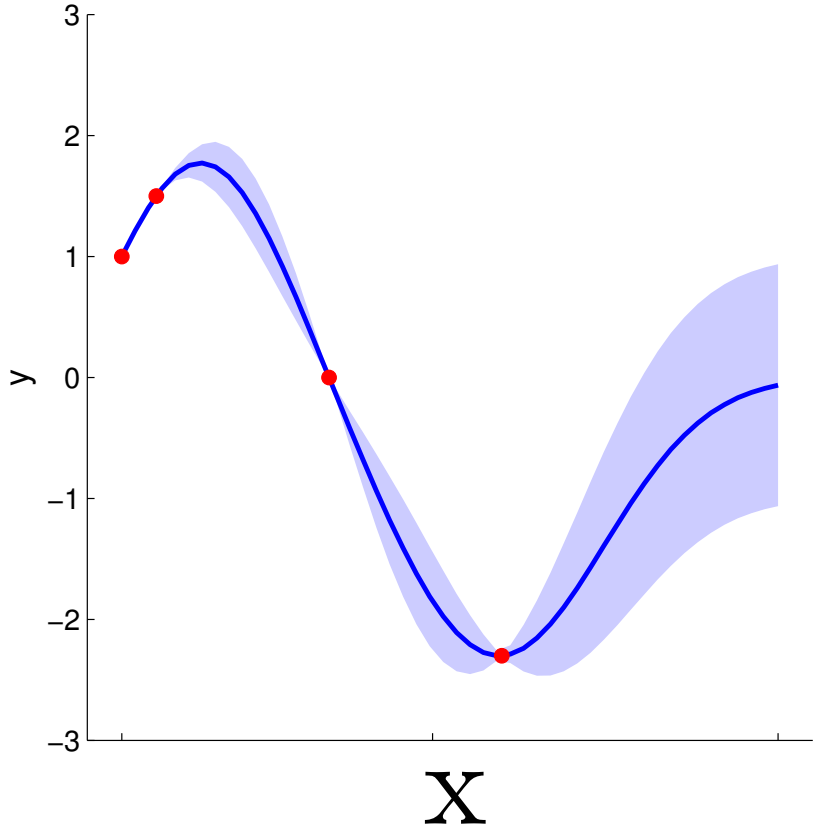
Regression: probabilistic inference in function space

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

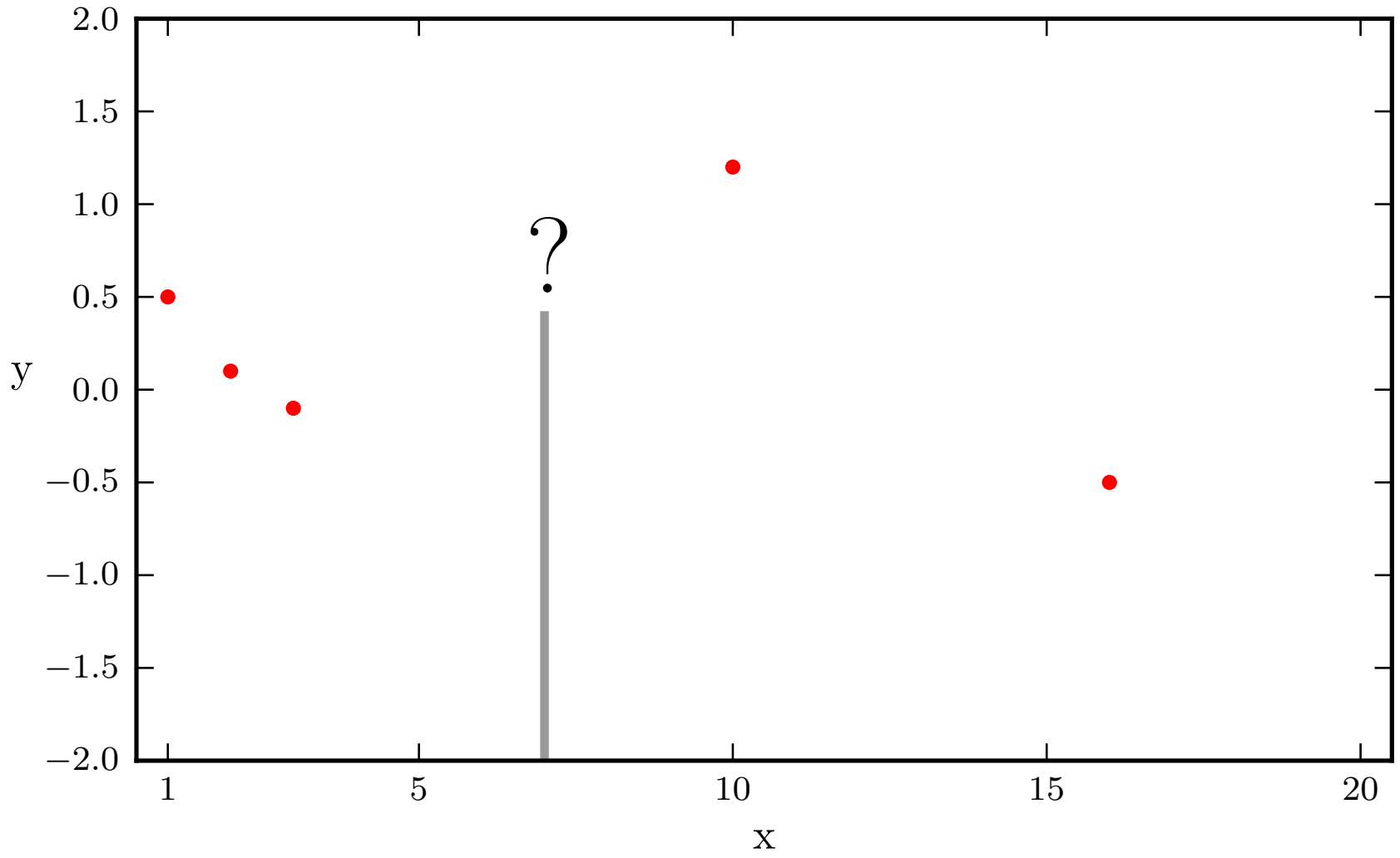
$$K(x_1, x_2) = \sigma^2 e^{-\frac{1}{2l^2}(x_1 - x_2)^2}$$

$\Sigma =$





Motivation: non-linear regression



Learning Functions from Data

Guess the parametric form of a function that could fit the data

- $f(x, \boldsymbol{w}) = \boldsymbol{w}^{\mathrm{T}} x$ [Linear function of $\boldsymbol{w}$ and x]
- $f(x, \boldsymbol{w}) = \boldsymbol{w}^{\mathrm{T}} \boldsymbol{\phi}(x)$ [Linear function of $\boldsymbol{w}$] (Linear Basis Function Model)
- $f(x, \boldsymbol{w}) = g(\boldsymbol{w}^{\mathrm{T}} \boldsymbol{\phi}(x))$ [Non-linear in x and $\boldsymbol{w}$] (E.g., Neural Network)

$\boldsymbol{\phi}(x)$ is a vector of basis functions. For example, if $\boldsymbol{\phi}(x) = (1, x, x^2)$ and $x \in \mathbb{R}^1$ then $f(x, \boldsymbol{w}) = w_0 + w_1 x + w_2 x^2$ is a quadratic function.

Slide credit: Andrew G. Wilson

Learning Functions from Data

Guess the parametric form of a function that could fit the data

- $f(x, \boldsymbol{w}) = \boldsymbol{w}^{\mathrm{T}} x$ [Linear function of $\boldsymbol{w}$ and x]
- $f(x, \boldsymbol{w}) = \boldsymbol{w}^{\mathrm{T}} \boldsymbol{\phi}(x)$ [Linear function of $\boldsymbol{w}$] (Linear Basis Function Model)
- $f(x, \boldsymbol{w}) = g(\boldsymbol{w}^{\mathrm{T}} \boldsymbol{\phi}(x))$ [Non-linear in x and $\boldsymbol{w}$] (E.g., Neural Network)

$\boldsymbol{\phi}(x)$ is a vector of basis functions. For example, if $\boldsymbol{\phi}(x) = (1, x, x^2)$ and $x \in \mathbb{R}^1$ then $f(x, \boldsymbol{w}) = w_0 + w_1 x + w_2 x^2$ is a quadratic function.

Choose an error measure $E(\boldsymbol{w})$, minimize with respect to $\boldsymbol{w}$

- $E(\boldsymbol{w}) = \sum_{i=1}^{N} [f(x_i, \boldsymbol{w}) - y(x_i)]^2$

Slide credit: Andrew G. Wilson

A Probabilistic Approach

We could explicitly account for noise in our model.

- $y(x) = f(x, \boldsymbol{w}) + \epsilon(x)$, where $\epsilon(x)$ is a noise function.

One commonly takes $\epsilon(x) = \mathcal{N}(0, \sigma^2)$ for i.i.d. additive Gaussian noise, in which case

$$p(y(x)|x, \boldsymbol{w}, \sigma^2) = \mathcal{N}(y(x); f(x, \boldsymbol{w}), \sigma^2) \qquad \text{Observation Model}$$

$$p(\boldsymbol{y}|x, \boldsymbol{w}, \sigma^2) = \prod_{i=1}^{N} \mathcal{N}(y(x_i); f(x_i, \boldsymbol{w}), \sigma^2) \qquad \text{Likelihood}$$

- Maximize the likelihood of the data $p(\boldsymbol{y}|x, \boldsymbol{w}, \sigma^2)$ with respect to $\sigma^2, \boldsymbol{w}$.

Slide credit: Andrew G. Wilson

A Probabilistic Approach

- The probabilistic approach helps us interpret the error measure in a deterministic approach, and gives us a sense of the noise level σ^2.
- Probabilistic methods thus provide an intuitive framework for representing uncertainty, and model development.

Slide credit: Andrew G. Wilson

Bayesian Model

$$\text{posterior} = \frac{\text{likelihood} \times \text{prior}}{\text{marginal likelihood}}, \quad p(\boldsymbol{w}|\boldsymbol{y}, X, \sigma^2) = \frac{p(\boldsymbol{y}|X, \boldsymbol{w}, \sigma^2)p(\boldsymbol{w})}{p(\boldsymbol{y}|X, \sigma^2)}$$

Predictive Distribution

$$p(y|x_*, \boldsymbol{y}, X) = \int p(y|x_*, \boldsymbol{w})p(\boldsymbol{w}|\boldsymbol{y}, X)d\boldsymbol{w} \,.$$

Slide credit: Andrew G. Wilson

Bayesian Model

$$\text{posterior} = \frac{\text{likelihood} \times \text{prior}}{\text{marginal likelihood}}, \quad p(\boldsymbol{w}|\boldsymbol{y}, X, \sigma^2) = \frac{p(\boldsymbol{y}|X, \boldsymbol{w}, \sigma^2)p(\boldsymbol{w})}{p(\boldsymbol{y}|X, \sigma^2)}$$

Predictive Distribution

$$p(y|x_*, \boldsymbol{y}, X) = \int p(y|x_*, \boldsymbol{w})p(\boldsymbol{w}|\boldsymbol{y}, X)d\boldsymbol{w}\,.$$

- Average of infinitely many models weighted by their posterior probabilities.
- No over-fitting, automatically calibrated complexity.
- Typically more interested in distribution over functions than in parameters $\boldsymbol{w}$.

Slide credit: Andrew G. Wilson

Parametric vs. Nonparameteric Modeling

- Parametric models:
  - Assume data can be represented using a fixed, finite number of parameters.
  - Mixture of K Gaussians, polynomial regression, neural nets, etc.
- Nonparameteric models:
  - Number of parameters can grow with sample size.
  - Kernel density estimation.
- Bayesian nonparameterics:
  - Allow for an infinite number of parameters a priori.
  - Models of finite datasets will have only finite number of parameters.
  - Other parameters are integrated out.

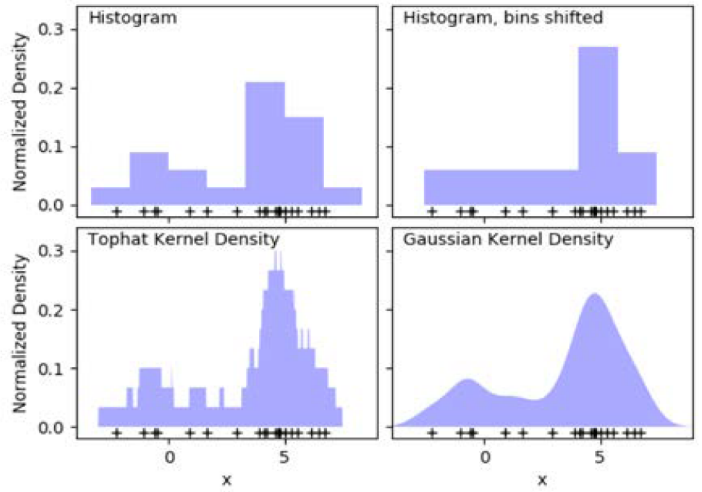


Image: scikit-learn.org

Parametric Bayesian Inference

- A parametric likelihood: $\mathbf{x} \sim p(\cdot|\theta)$
- Prior on θ : $\pi(\theta)$
- Posterior distribution

$$p(\theta|\mathbf{x}) = \frac{p(\mathbf{x}|\theta)\pi(\theta)}{\int p(\mathbf{x}|\theta)\pi(\theta)d\theta} \propto p(\mathbf{x}|\theta)\pi(\theta)$$

Examples:

- **Gaussian distribution prior + 2D Gaussian likelihood → Gaussian posterior distribution**
- **Dirichilet distribution prior + 2D Multinomial likelihood → Dirichlet posterior distribution**

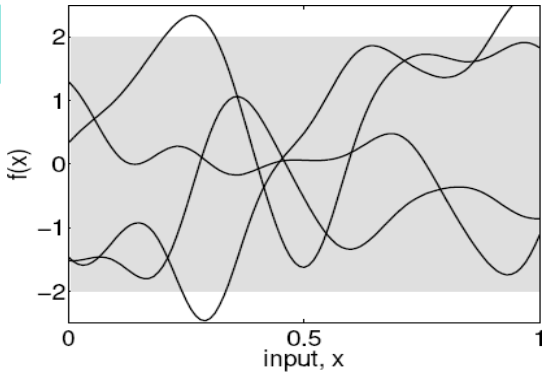
Nonparametric Bayesian Inference

$\mathcal{M}$ is a richer model, e.g., with an infinite set of parameters

- A nonparametric likelihood: $\mathbf{x} \sim p(\cdot|\mathcal{M})$
- Prior on $\mathcal{M}$: $\pi(\mathcal{M})$
- Posterior distribution

$$p(\mathcal{M}|\mathbf{x}) = \frac{p(\mathbf{x}|\mathcal{M})\pi(\mathcal{M})}{\int p(\mathbf{x}|\mathcal{M})\pi(\mathcal{M})d\mathcal{M}} \propto p(\mathbf{x}|\mathcal{M})\pi(\mathcal{M})$$

Examples: function

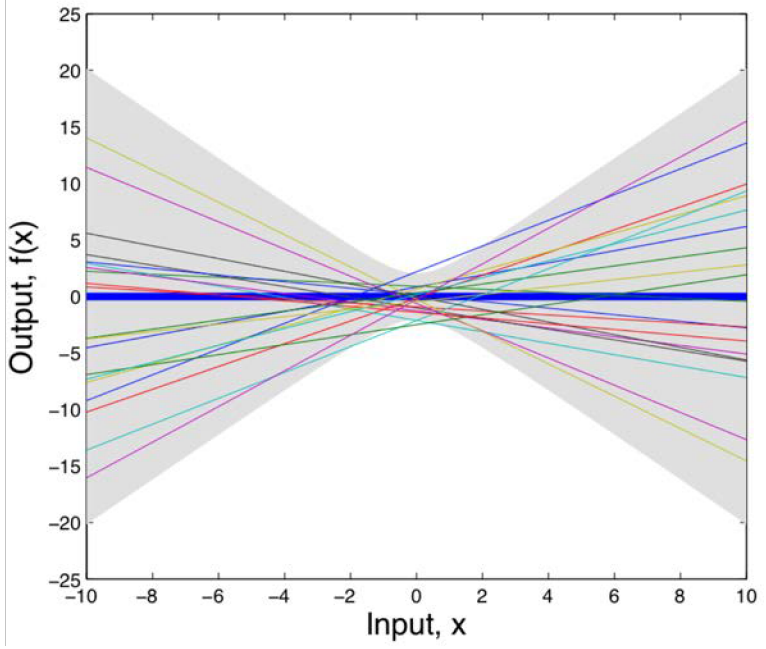


Gaussian Process Prior [Doob, 1944; Rasmussen & Williams, 2006]
+ Gaussian/Sigmoid/Softmax likelihood

Weight-space View

- Consider a simple linear model that result in a family of functions

$$f(x) = a_0 + a_1 x\,,$$
$$a_0, a_1 \sim \mathcal{N}(0,1)\,.$$



Function-space View

- We are interested in the **distribution over functions** induced by the distribution over parameters…
- In fact, we can characterize the properties of these functions directly:

$$f(x|a_0, a_1) = a_0 + a_1 x\,, \qquad a_0, a_1 \sim \mathcal{N}(0, 1)\,.$$

Function-space View

- We are interested in the **distribution over functions** induced by the distribution over parameters…
- In fact, we can characterize the properties of these functions directly:

$$f(x|a_0, a_1) = a_0 + a_1 x\,, \qquad a_0, a_1 \sim \mathcal{N}(0,1)\,.$$
$$\mathbb{E}[f(x)] = \mathbb{E}[a_0] + \mathbb{E}[a_1]x = 0\,.$$

Function-space View

- We are interested in the **distribution over functions** induced by the distribution over parameters…
- In fact, we can characterize the properties of these functions directly:

$$f(x|a_0, a_1) = a_0 + a_1 x\,, \qquad a_0, a_1 \sim \mathcal{N}(0,1)\,.$$
$$\mathbb{E}[f(x)] = \mathbb{E}[a_0] + \mathbb{E}[a_1]x = 0\,.$$
$$\mathrm{cov}[f(x_b), f(x_c)] = \mathbb{E}[f(x_b)f(x_c)] - \mathbb{E}[f(x_b)]\mathbb{E}[f(x_c)]$$

Function-space View

- We are interested in the **distribution over functions** induced by the distribution over parameters…
- In fact, we can characterize the properties of these functions directly:

$$
\begin{aligned}
f(x|a_0,a_1) &= a_0 + a_1 x\,, \qquad a_0, a_1 \sim \mathcal{N}(0,1)\,. \\
\mathbb{E}[f(x)] &= \mathbb{E}[a_0] + \mathbb{E}[a_1]x = 0\,. \\
\mathrm{cov}[f(x_b), f(x_c)] &= \mathbb{E}[f(x_b)f(x_c)] - \mathbb{E}[f(x_b)]\mathbb{E}[f(x_c)] \\
&= \mathbb{E}[a_0^2 + a_0 a_1 (x_b + x_c) + a_1^2 x_b x_c] - 0 \\
&= \mathbb{E}[a_0^2] + \mathbb{E}[a_1^2 x_b x_c] + \mathbb{E}[a_0 a_1 (x_b + x_c)] \\
&= 1 + x_b x_c + 0 \\
&= 1 + x_b x_c\,.
\end{aligned}
$$

Function-space View

- Any collection of f(x) values has a joint Gaussian distribution

  NOTE: here we have fixed x instead of X, and the randomness comes from the function f

$$[f(x_1), \ldots, f(x_N)] \sim \mathcal{N}(0, K) ,$$
$$K_{ij} = \mathrm{cov}(f(x_i), f(x_j)) = k(x_i, x_j) = 1 + x_b x_c .$$

Function-space View

- Any collection of f(x) values has a joint Gaussian distribution

  NOTE: here we have fixed x instead of X, and the randomness comes from the function f

$$[f(x_1), \ldots, f(x_N)] \sim \mathcal{N}(0, K) ,$$
$$K_{ij} = \mathrm{cov}(f(x_i), f(x_j)) = k(x_i, x_j) = 1 + x_b x_c .$$

- Definition: A Gaussian process (GP) is a collection of random variables, any finite number of which have a joint Gaussian distribution. We write $f(x) \sim \mathcal{GP}(m, k)$ to mean

$$[f(x_1), \ldots, f(x_N)] \sim \mathcal{N}(\boldsymbol{\mu}, K)$$
$$\boldsymbol{\mu}_i = m(x_i)$$
$$K_{ij} = k(x_i, x_j) ,$$

for any collection of input values $x_1, \ldots, x_N$. In other words, f is a GP with mean function $m(x)$ and *covariance kernel* $k(x_i, x_j)$.

Another GP Example: Linear Basis Function Models

- Model specification:

$$f(x, \boldsymbol{w}) = \boldsymbol{w}^{\mathrm{T}} \boldsymbol{\phi}(x)$$
$$p(\boldsymbol{w}) = \mathcal{N}(0, \Sigma_w)$$

- Moments of the the induced distribution over functions:

$$\begin{aligned}
\mathbb{E}[f(x, \boldsymbol{w})] &= m(x) = \mathbb{E}[\boldsymbol{w}^{\mathrm{T}}]\boldsymbol{\phi}(x) = 0 \\
\mathrm{cov}(f(x_i), f(x_j)) &= k(x_i, x_j) = \mathbb{E}[f(x_i) f(x_j)] - \mathbb{E}[f(x_i)]\mathbb{E}[f(x_j)] \\
&= \boldsymbol{\phi}(x_i)^{\mathrm{T}} \mathbb{E}[\boldsymbol{w}\boldsymbol{w}^{\mathrm{T}}] \boldsymbol{\phi}(x_j) - 0 \\
&= \boldsymbol{\phi}(x_i)^{\mathrm{T}} \Sigma_w \boldsymbol{\phi}(x_j)
\end{aligned}$$

- $f(x, \boldsymbol{w})$ is a Gaussian process, $f(x) \sim \mathcal{N}(m, k)$ with mean function $m(x) = 0$ and covariance kernel $k(x_i, x_j) = \boldsymbol{\phi}(x_i)^{\mathrm{T}} \Sigma_w \boldsymbol{\phi}(x_j)$.

Gaussian Processes

Interpretability:

- We are ultimately more interested in – and have stronger intuitions about – the **functions** that model our data than **weights** w in a parametric model. We can express these intuitions using a covariance kernel.

Generalization:

- The kernel controls the support and inductive biases of our model, and thus its ability to generalize to unseen.

Gaussian Process: Graphical Model

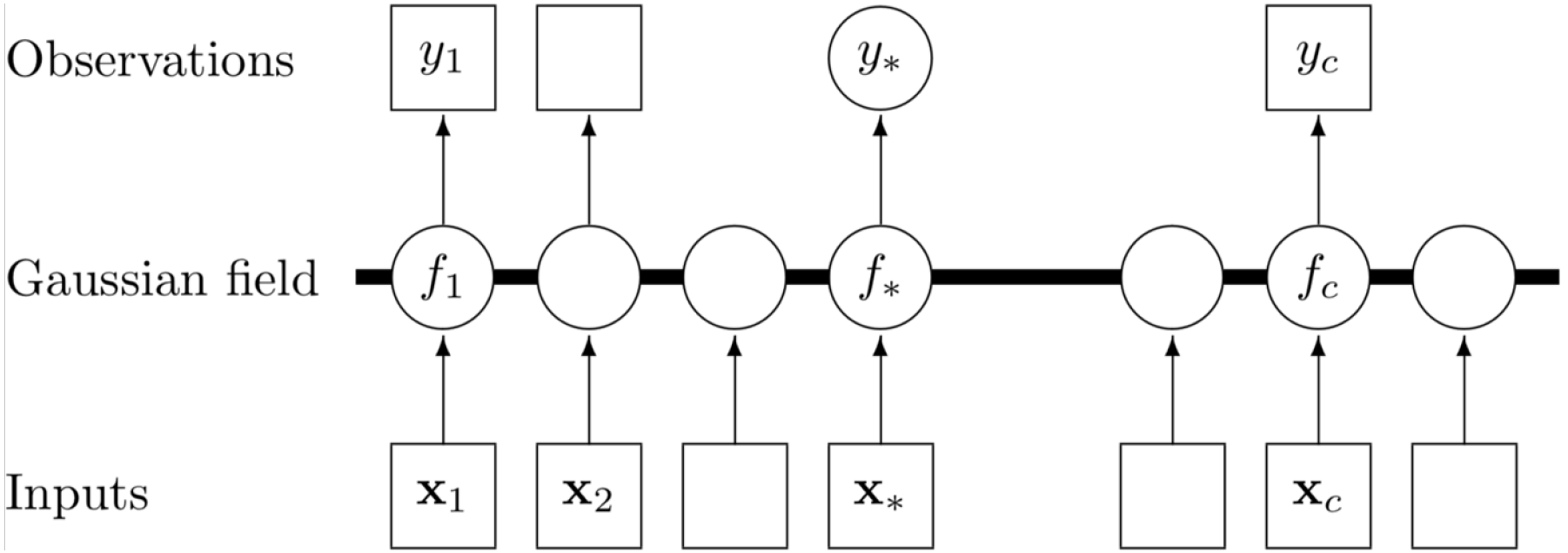


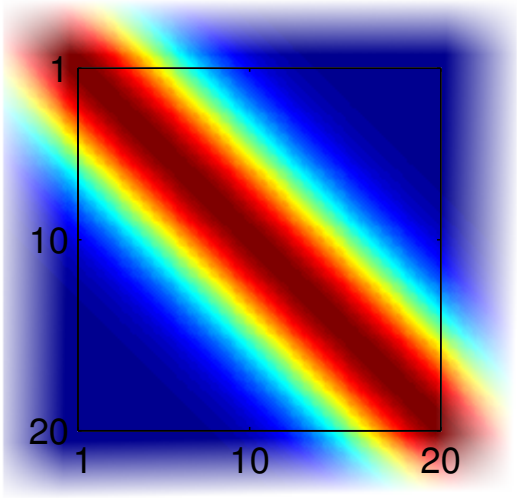
Image source: GPML book

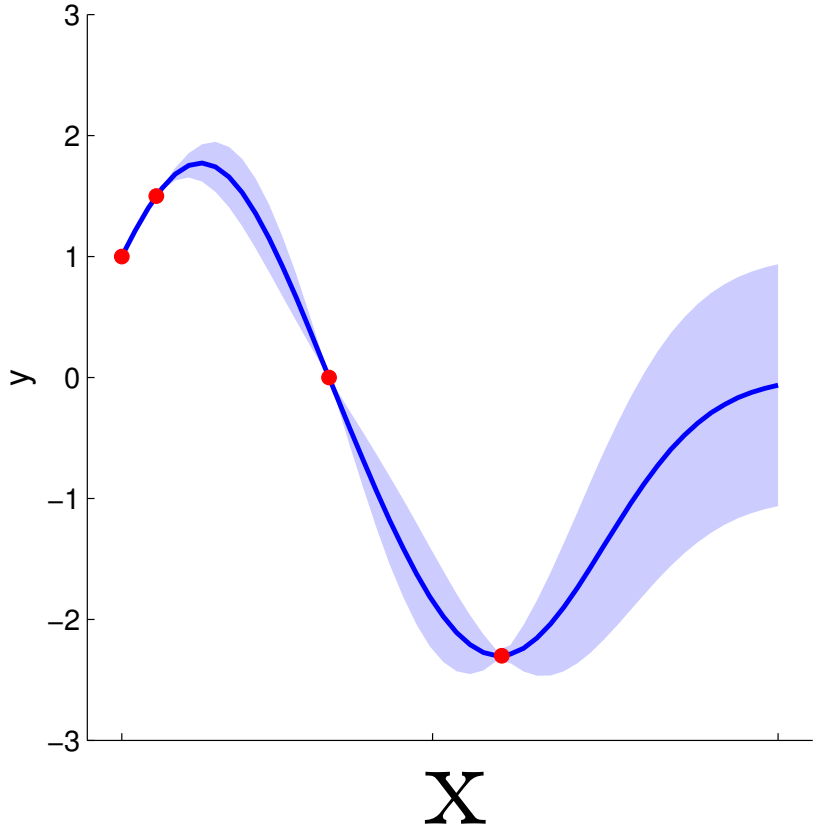
Regression: probabilistic inference in function space

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

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

$\Sigma =$





Regression: probabilistic inference in function space

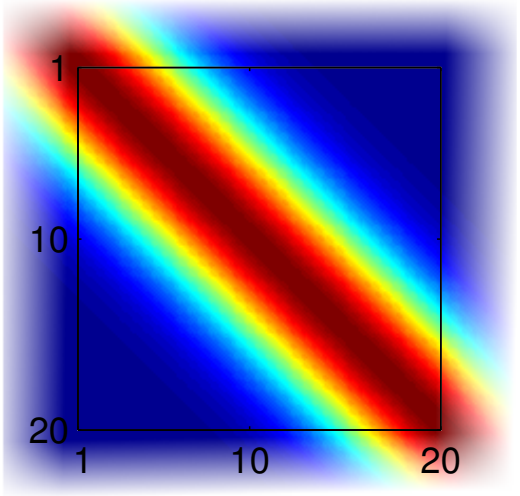
Non-parametric (∞-parametric)

$$p(\mathsf{y}|\theta) = \mathcal{N}(0, \Sigma)$$

$$\Sigma(\mathsf{x}_1, \mathsf{x}_2) = \mathsf{K}(\mathsf{x}_1, \mathsf{x}_2) + \mathsf{I}\sigma_{\mathsf{y}}^2$$

$$\mathsf{K}(\mathsf{x}_1, \mathsf{x}_2) = \sigma^2 \mathrm{e}^{-\frac{1}{2l^2}(\mathsf{x}_1 - \mathsf{x}_2)^2}$$

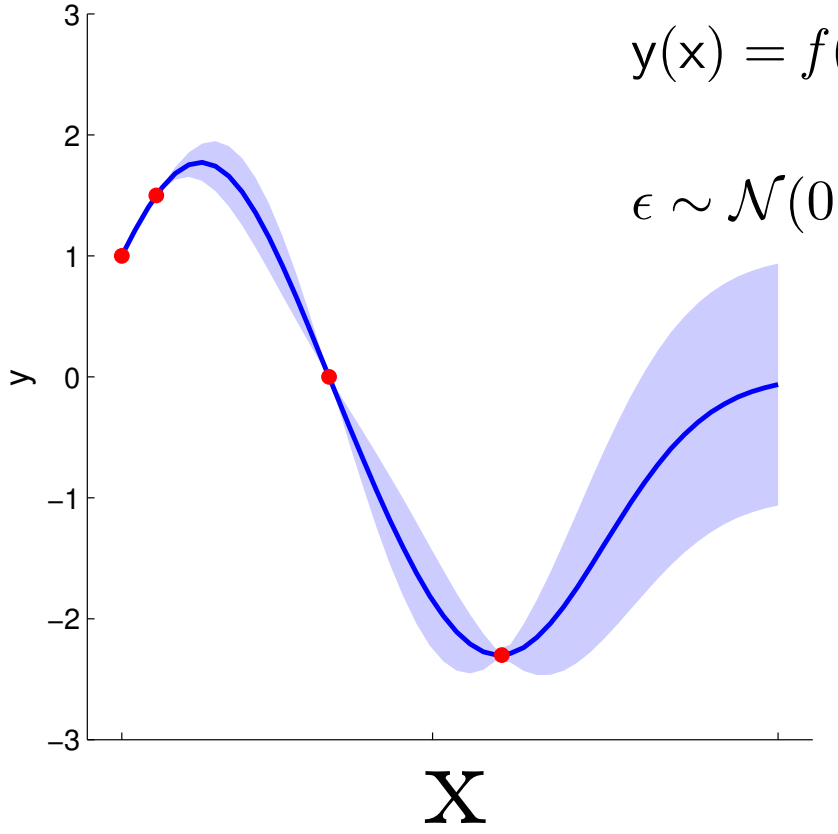
$\Sigma =$



Parametric model

$$\mathsf{y}(\mathsf{x}) = f(\mathsf{x}; \theta) + \sigma_{\mathsf{y}}\epsilon$$

$$\epsilon \sim \mathcal{N}(0, 1)$$



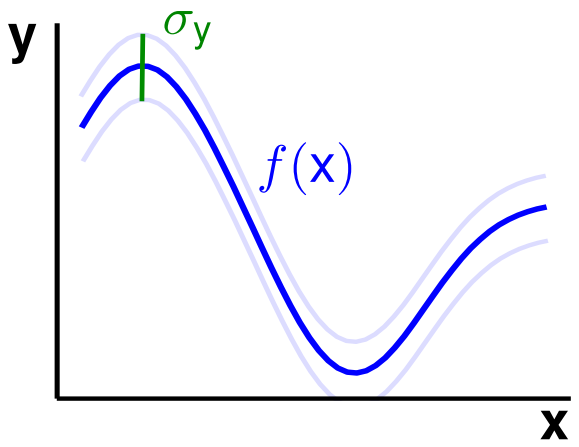
Mathematical Foundations: Regression

Q1. What's the formal justification for how we were using GPs for regression?

generative model (like non-linear regression)

$$\mathsf{y}(\mathsf{x}) = f(\mathsf{x}) + \epsilon\sigma_{\mathsf{y}}$$

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



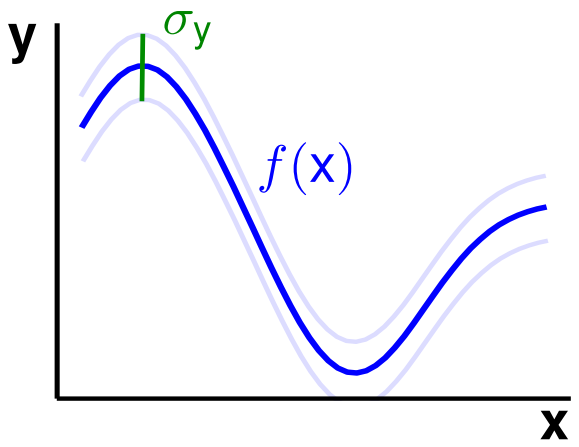
Mathematical Foundations: Regression

Q1. What's the formal justification for how we were using GPs for regression?

generative model (like non-linear regression)

$$\mathsf{y}(\mathsf{x}) = f(\mathsf{x}) + \epsilon\sigma_{\mathsf{y}}$$

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



place GP prior over the non-linear function

$$p(f(\mathsf{x})|\theta) = \mathcal{GP}(0, \mathsf{K}(\mathsf{x}, \mathsf{x}'))$$

$$\mathsf{K}(\mathsf{x}, \mathsf{x}') = \sigma^2 \exp\left(-\frac{1}{2l^2}(\mathsf{x} - \mathsf{x}')^2\right)$$ (smoothly wiggling functions expected)

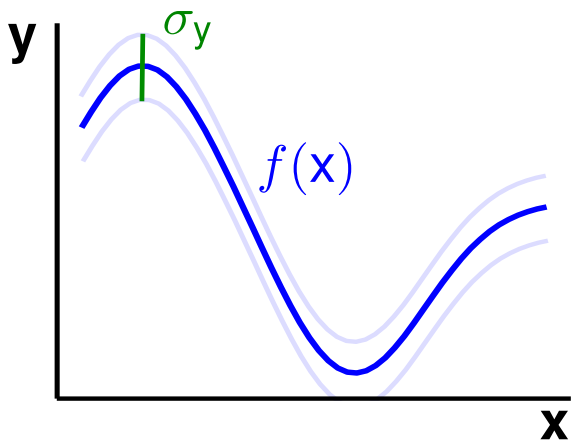
Mathematical Foundations: Regression

Q1. What's the formal justification for how we were using GPs for regression?

generative model (like non-linear regression)

$$\mathsf{y}(\mathsf{x}) = f(\mathsf{x}) + \epsilon\sigma_{\mathsf{y}}$$

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



place GP prior over the non-linear function

$$p(f(\mathsf{x})|\theta) = \mathcal{GP}(0, \mathsf{K}(\mathsf{x}, \mathsf{x}'))$$

$$\mathsf{K}(\mathsf{x}, \mathsf{x}') = \sigma^2 \exp\left(-\frac{1}{2l^2}(\mathsf{x} - \mathsf{x}')^2\right)$$ (smoothly wiggling functions expected)

sum of Gaussian variables = Gaussian: induces a GP over $\mathsf{y}(\mathsf{x})$

$$p(\mathsf{y}(\mathsf{x})|\theta) = \mathcal{GP}(0, \mathsf{K}(\mathsf{x}, \mathsf{x}') + \mathrm{I}\sigma_{\mathsf{y}}^2)$$

Mathematical Foundations: Marginalisation

Q2. A GP is "like" a Gaussian distribution with an infinitely long mean vector and an "infinite by infinite" covariance matrix, so how do we represent it on a computer?

Mathematical Foundations: Marginalisation

Q2. A GP is "like" a Gaussian distribution with an infinitely long mean vector and an "infinite by infinite" covariance matrix, so how do we represent it on a computer?

We are saved by the marginalisation property:

$$p(\mathbf{y}_1) = \int p(\mathbf{y}_1, \mathbf{y}_2) \mathrm{d}\mathbf{y}_2$$

$$p(\mathbf{y}_1, \mathbf{y}_2) = \mathcal{N}\left(\begin{bmatrix} \mathbf{a} \\ \mathbf{b} \end{bmatrix}, \begin{bmatrix} \mathrm{A} & \mathrm{B} \\ \mathrm{B}^\mathsf{T} & \mathrm{C} \end{bmatrix}\right)$$

Mathematical Foundations: Marginalisation

Q2. A GP is "like" a Gaussian distribution with an infinitely long mean vector and an "infinite by infinite" covariance matrix, so how do we represent it on a computer?

We are saved by the marginalisation property:

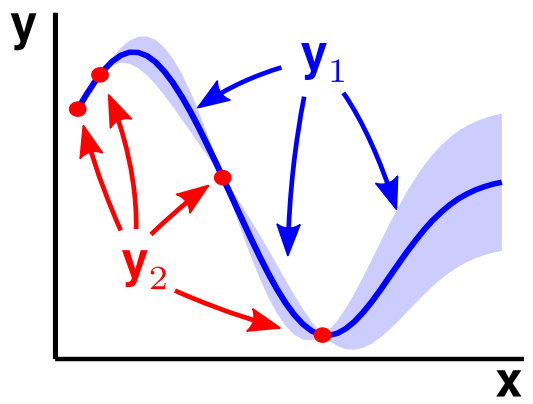
$$p(\mathbf{y}_1) = \int p(\mathbf{y}_1, \mathbf{y}_2) \mathrm{d}\mathbf{y}_2$$

$$p(\mathbf{y}_1, \mathbf{y}_2) = \mathcal{N}\left(\begin{bmatrix} \mathbf{a} \\ \mathbf{b} \end{bmatrix}, \begin{bmatrix} \mathsf{A} & \mathsf{B} \\ \mathsf{B}^\mathsf{T} & \mathsf{C} \end{bmatrix}\right) \quad \Longrightarrow \quad p(\mathbf{y}_1) = \mathcal{N}(\mathbf{a}, \mathsf{A})$$

$\Longrightarrow$ Only need to represent finite dimensional projections of GPs on computer.

Mathematical Foundations: Prediction

Q3. How do we make predictions?



Mathematical Foundations: Prediction

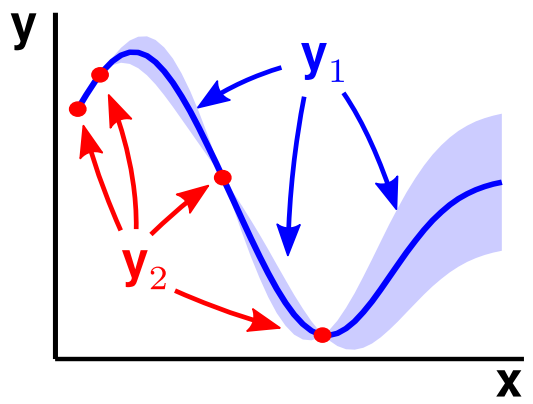
Q3. How do we make predictions?

A=K(x1, x1)
B=K(x1, x2)
C=K(x2, x2)

$$p(\mathbf{y}_1, \mathbf{y}_2) = \mathcal{N}\left(\begin{bmatrix} \mathbf{a} \\ \mathbf{b} \end{bmatrix}, \begin{bmatrix} \mathsf{A} & \mathsf{B} \\ \mathsf{B}^\intercal & \mathsf{C} \end{bmatrix}\right)$$

$$p(\mathbf{y}_1|\mathbf{y}_2) = \frac{p(\mathbf{y}_1, \mathbf{y}_2)}{p(\mathbf{y}_2)}$$

$$\implies p(\mathbf{y}_1|\mathbf{y}_2) = \mathcal{N}(\mathbf{a} + \mathsf{B}\mathsf{C}^{-1}(\mathbf{y}_2 - \mathbf{b}), \mathsf{A} - \mathsf{B}\mathsf{C}^{-1}\mathsf{B}^\intercal)$$



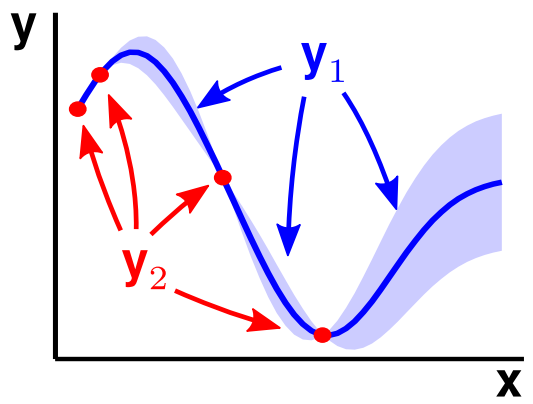
Mathematical Foundations: Prediction

Q3. How do we make predictions?

A=K(x1, x1)
B=K(x1, x2)
C=K(x2, x2)

$$p(\mathbf{y}_1, \mathbf{y}_2) = \mathcal{N}\left(\begin{bmatrix} \mathbf{a} \\ \mathbf{b} \end{bmatrix}, \begin{bmatrix} \mathrm{A} & \mathrm{B} \\ \mathrm{B}^\intercal & \mathrm{C} \end{bmatrix}\right)$$

$$p(\mathbf{y}_1|\mathbf{y}_2) = \frac{p(\mathbf{y}_1, \mathbf{y}_2)}{p(\mathbf{y}_2)}$$



$$\implies p(\mathbf{y}_1|\mathbf{y}_2) = \mathcal{N}(\mathbf{a} + \mathrm{BC}^{-1}(\mathbf{y}_2 - \mathbf{b}), \mathrm{A} - \mathrm{BC}^{-1}\mathrm{B}^\intercal)$$

predictive mean

$$\mu_{\mathbf{y}_1|\mathbf{y}_2} = \mathbf{a} + \mathrm{BC}^{-1}(\mathbf{y}_2 - \mathbf{b})$$
$$= \mathrm{BC}^{-1}\mathbf{y}_2$$
$$= \mathrm{W}\mathbf{y}_2$$

linear in the data

predictive covariance

$$\Sigma_{\mathbf{y}_1|\mathbf{y}_2} = \mathrm{A} - \mathrm{BC}^{-1}\mathrm{B}^\intercal$$

predictive uncertainty = prior uncertainty − reduction in uncertainty

predictions more confident than prior

Gaussian Process and Determinantal Point Process

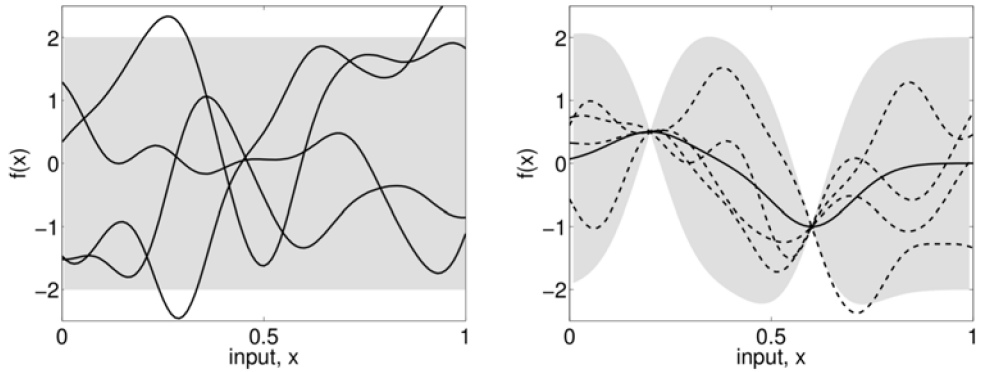
Shichang Zhang

05/06/2021

Slides adapted from Eric Xing, Richard Turner, and Pengtao Xie

Outline

- Part 1: Gaussian Processes (GPs)
  - Motivating example of GPs
  - Parametric model and non-parametric model
  - Definition of GPs
  - Inference and learning with GPs
  - Review
  - Comparing different kernel functions (IOU)
  - Deep kernel learning
  - Graph Convolutional Gaussian Processes (ICML 2019)
- Part 2: Determinantal Point Process (DPP)



(a), prior (b), posterior

Walker, I., & Glocker, B. (2019, May). Graph convolutional Gaussian processes. In *International Conference on Machine Learning* (pp. 6495-6504). PMLR.

Review: Definition of Gaussian Process

A Gaussian process (GP) is a collection of random variables, any finite number of which have a joint Gaussian distribution. We write $f(x) \sim \mathcal{GP}(m, k)$ to mean

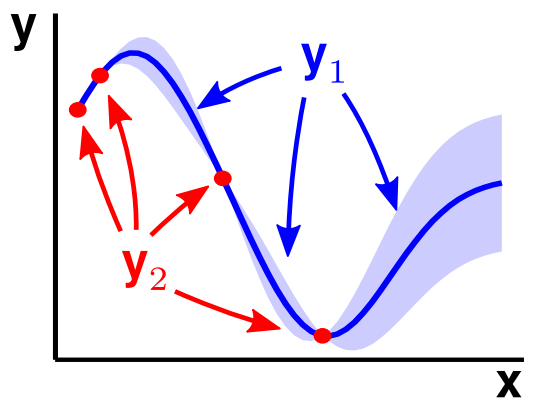
$$
\begin{aligned}
[f(x_1), \ldots, f(x_N)] &\sim \mathcal{N}(\boldsymbol{\mu}, K) \\
\boldsymbol{\mu}_i &= m(x_i) \\
K_{ij} &= k(x_i, x_j) \,,
\end{aligned}
$$

for any collection of input values $x_1, \ldots, x_N$. In other words, f is a GP with mean function $m(x)$ and *covariance kernel* $k(x_i, x_j)$.

Note: GP is a nonparameteric model, meaning the number of parameters can grow with sample size.

Mathematical Foundations: Prediction

Q3. How do we make predictions?



Mathematical Foundations: Prediction

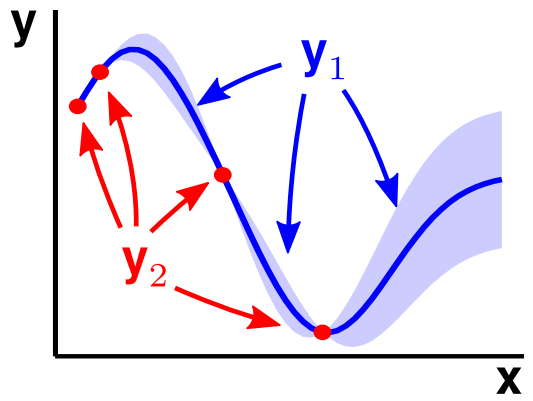
Q3. How do we make predictions?

A=K(x1, x1)
B=K(x1, x2)
C=K(x2, x2)

$$p(\mathbf{y}_1, \mathbf{y}_2) = \mathcal{N}\left(\begin{bmatrix} \mathbf{a} \\ \mathbf{b} \end{bmatrix}, \begin{bmatrix} \mathsf{A} & \mathsf{B} \\ \mathsf{B}^\intercal & \mathsf{C} \end{bmatrix}\right)$$

$$p(\mathbf{y}_1|\mathbf{y}_2) = \frac{p(\mathbf{y}_1, \mathbf{y}_2)}{p(\mathbf{y}_2)}$$

$$\implies p(\mathbf{y}_1|\mathbf{y}_2) = \mathcal{N}(\mathbf{a} + \mathsf{B}\mathsf{C}^{-1}(\mathbf{y}_2 - \mathbf{b}), \mathsf{A} - \mathsf{B}\mathsf{C}^{-1}\mathsf{B}^\intercal)$$



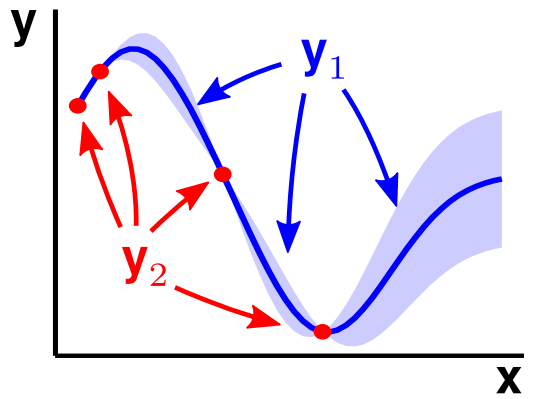
Mathematical Foundations: Prediction

Q3. How do we make predictions?

A=K(x1, x1)
B=K(x1, x2)
C=K(x2, x2)

$$p(\mathbf{y}_1, \mathbf{y}_2) = \mathcal{N}\left(\begin{bmatrix} \mathbf{a} \\ \mathbf{b} \end{bmatrix}, \begin{bmatrix} \mathrm{A} & \mathrm{B} \\ \mathrm{B}^\intercal & \mathrm{C} \end{bmatrix}\right)$$

$$p(\mathbf{y}_1|\mathbf{y}_2) = \frac{p(\mathbf{y}_1, \mathbf{y}_2)}{p(\mathbf{y}_2)}$$



$$\implies p(\mathbf{y}_1|\mathbf{y}_2) = \mathcal{N}(\mathbf{a} + \mathrm{B}\mathrm{C}^{-1}(\mathbf{y}_2 - \mathbf{b}), \mathrm{A} - \mathrm{B}\mathrm{C}^{-1}\mathrm{B}^\intercal)$$

predictive mean

$$\mu_{\mathbf{y}_1|\mathbf{y}_2} = \mathbf{a} + \mathrm{B}\mathrm{C}^{-1}(\mathbf{y}_2 - \mathbf{b})$$
$$= \mathrm{B}\mathrm{C}^{-1}\mathbf{y}_2$$
$$= \mathrm{W}\mathbf{y}_2$$

linear in the data

predictive covariance

$$\Sigma_{\mathbf{y}_1|\mathbf{y}_2} = \mathrm{A} - \mathrm{B}\mathrm{C}^{-1}\mathrm{B}^\intercal$$

predictive uncertainty = prior uncertainty − reduction in uncertainty

predictions more confident than prior

Exponentiated Quadratic Kernel

The exponentiated quadratic kernel (also known as squared exponential kernel, Gaussian kernel or radial basis function kernel) is one of the most popular kernels used in Gaussian process modelling. It can be computed as:

$$k(x_a, x_b) = \sigma^2 \exp\left(-\frac{\|x_a - x_b\|^2}{2\ell^2}\right)$$

With:

- σ^2 the overall variance (σ is also known as amplitude).
- ℓ the lengthscale.

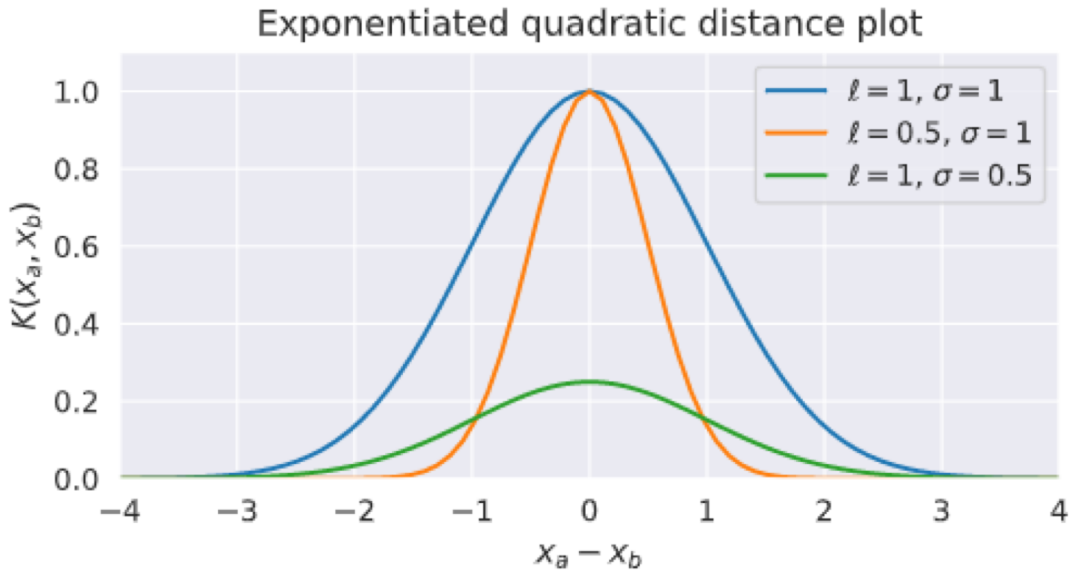
Exponentiated Quadratic Kernel

The exponentiated quadratic kernel (also known as squared exponential kernel, Gaussian kernel or radial basis function kernel) is one of the most popular kernels used in Gaussian process modelling. It can be computed as:

$$k(x_a, x_b) = \sigma^2 \exp\left(-\frac{\|x_a - x_b\|^2}{2\ell^2}\right)$$

With:

- σ^2 the overall variance (σ is also known as amplitude).
- ℓ the lengthscale.



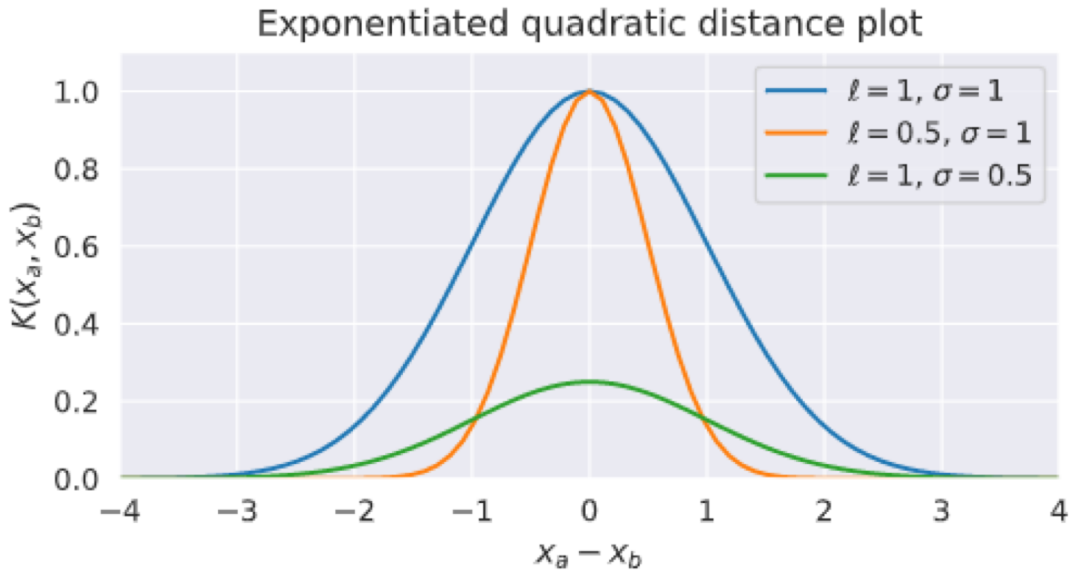
Exponentiated Quadratic Kernel

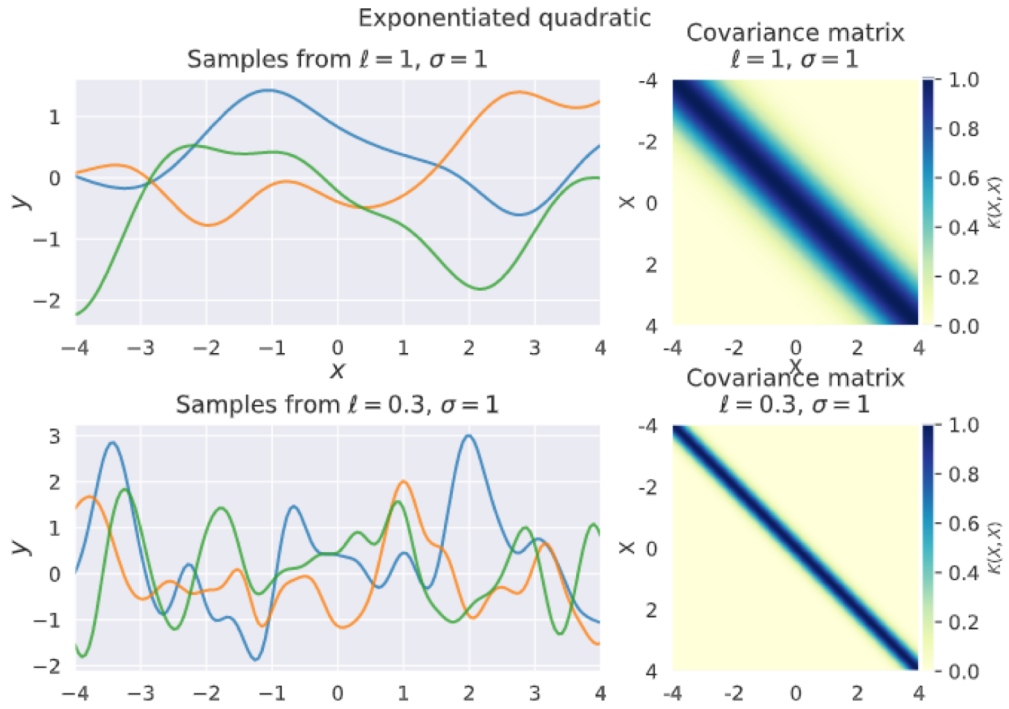
The exponentiated quadratic kernel (also known as squared exponential kernel, Gaussian kernel or radial basis function kernel) is one of the most popular kernels used in Gaussian process modelling. It can be computed as:

$$k(x_a, x_b) = \sigma^2 \exp\left(-\frac{\|x_a - x_b\|^2}{2\ell^2}\right)$$

With:

- σ^2 the overall variance (σ is also known as amplitude).
- ℓ the lengthscale.





Periodic Kernel

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

With:

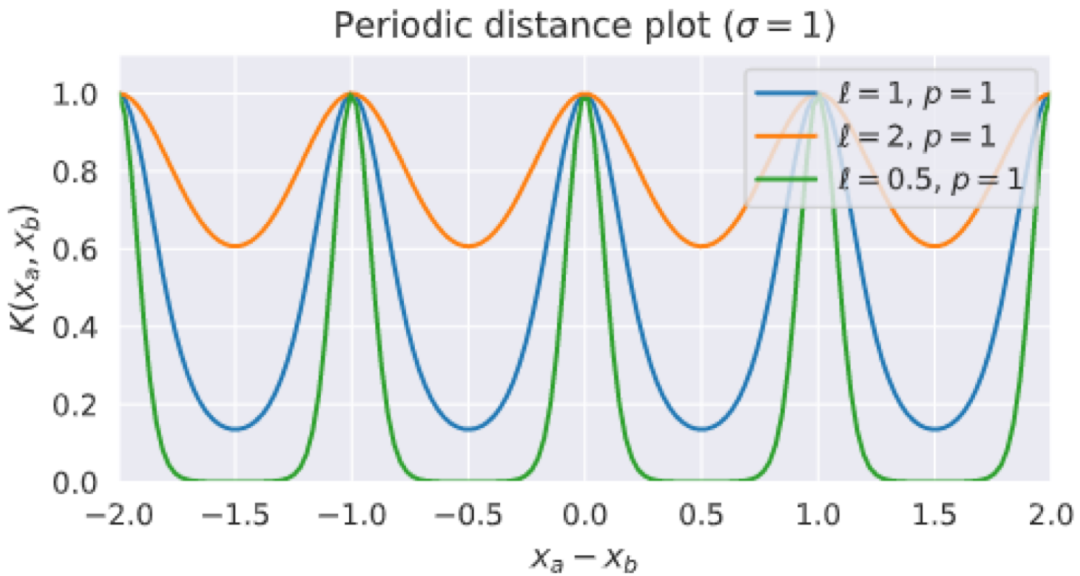
- σ^2 the overall variance (σ is also known as amplitude).
- ℓ the lengthscale.
- p the period, which is the distance between repetitions.

Periodic Kernel

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

With:

- σ^2 the overall variance (σ is also known as amplitude).
- ℓ the lengthscale.
- p the period, which is the distance between repetitions.

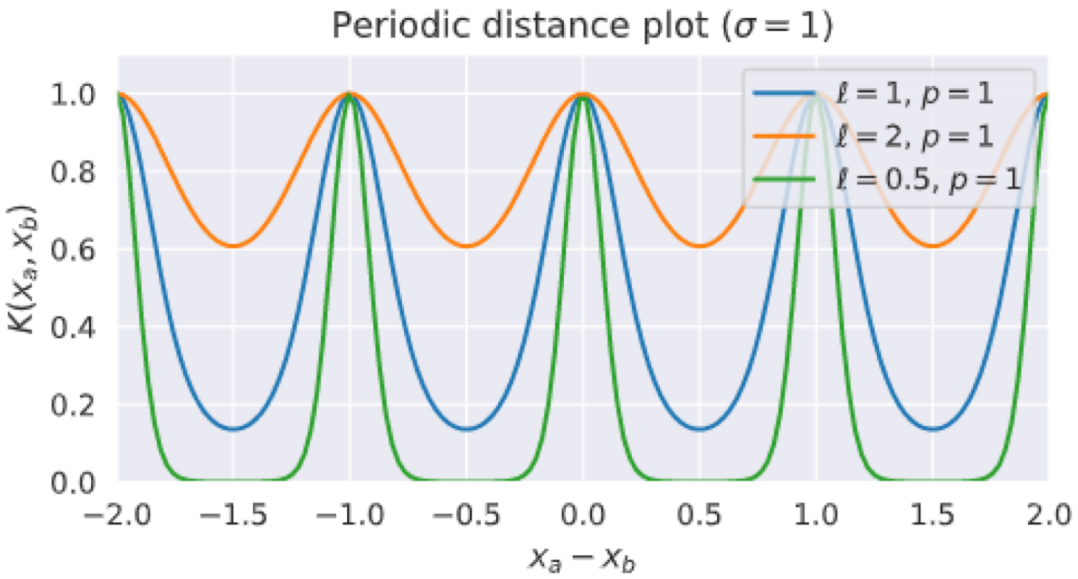


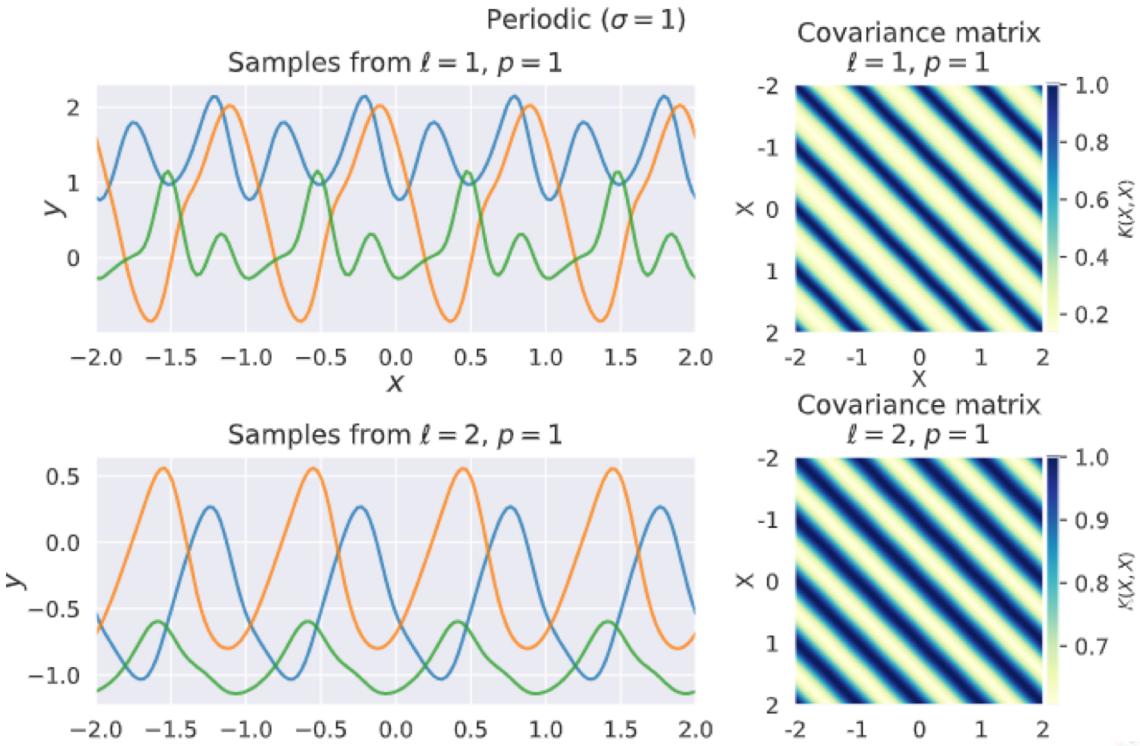
Periodic Kernel

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

With:

- σ^2 the overall variance (σ is also known as amplitude).
- ℓ the lengthscale.
- p the period, which is the distance between repetitions.





Combine Kernels by Multiplication: Local Periodic

The local periodic kernel is a multiplication of the periodic kernel with the exponentiated quadratic kernel to allow the periods to vary over longer distances. Note that the variance parameters σ^2 are combined into one.

$$k(x_a, x_b) = \sigma^2 \exp\left(-\frac{2}{\ell_p^2}\sin^2\left(\pi\frac{|x_a - x_b|}{p}\right)\right)\exp\left(-\frac{\|x_a - x_b\|^2}{2\ell_{eq}^2}\right)$$

With:

- σ^2 the overall variance (σ is also known as amplitude).
- ℓ_p lengthscale of the periodic function.
- p the period.
- ℓ_{eq} the lengthscale of the exponentiated quadratic.

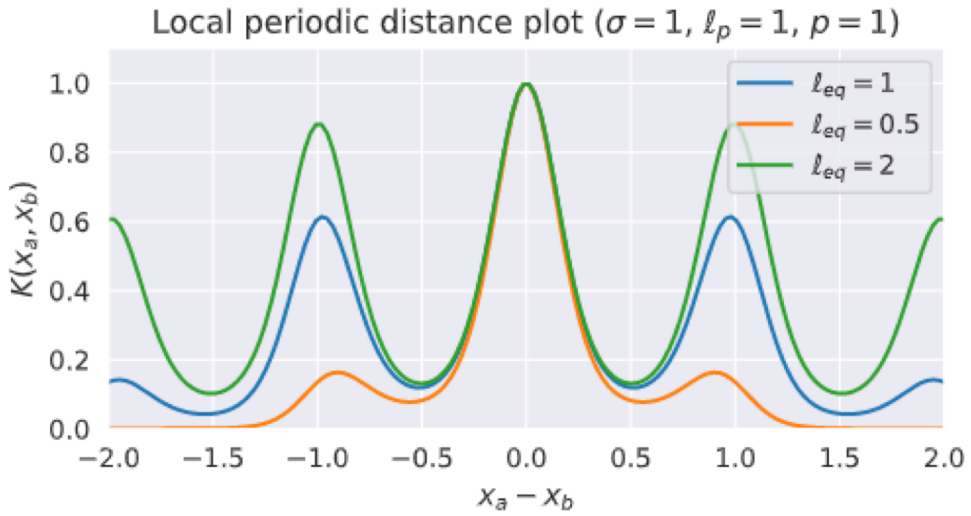
Combine Kernels by Multiplication: Local Periodic

The local periodic kernel is a multiplication of the periodic kernel with the exponentiated quadratic kernel to allow the periods to vary over longer distances. Note that the variance parameters σ^2 are combined into one.

$$k(x_a, x_b) = \sigma^2 \exp\left(-\frac{2}{\ell_p^2}\sin^2\left(\pi\frac{|x_a - x_b|}{p}\right)\right)\exp\left(-\frac{\|x_a - x_b\|^2}{2\ell_{eq}^2}\right)$$

With:

- σ^2 the overall variance (σ is also known as amplitude).
- ℓ_p lengthscale of the periodic function.
- p the period.
- ℓ_{eq} the lengthscale of the exponentiated quadratic.



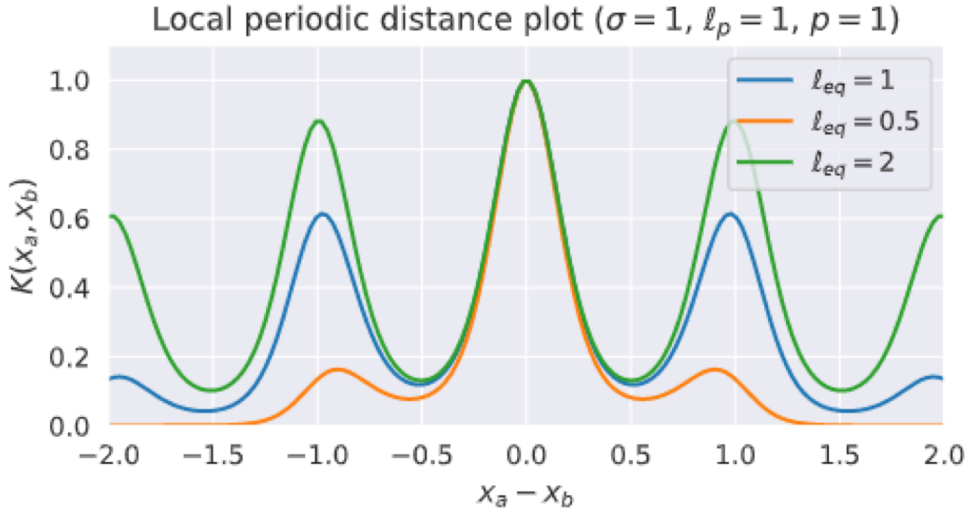
Combine Kernels by Multiplication: Local Periodic

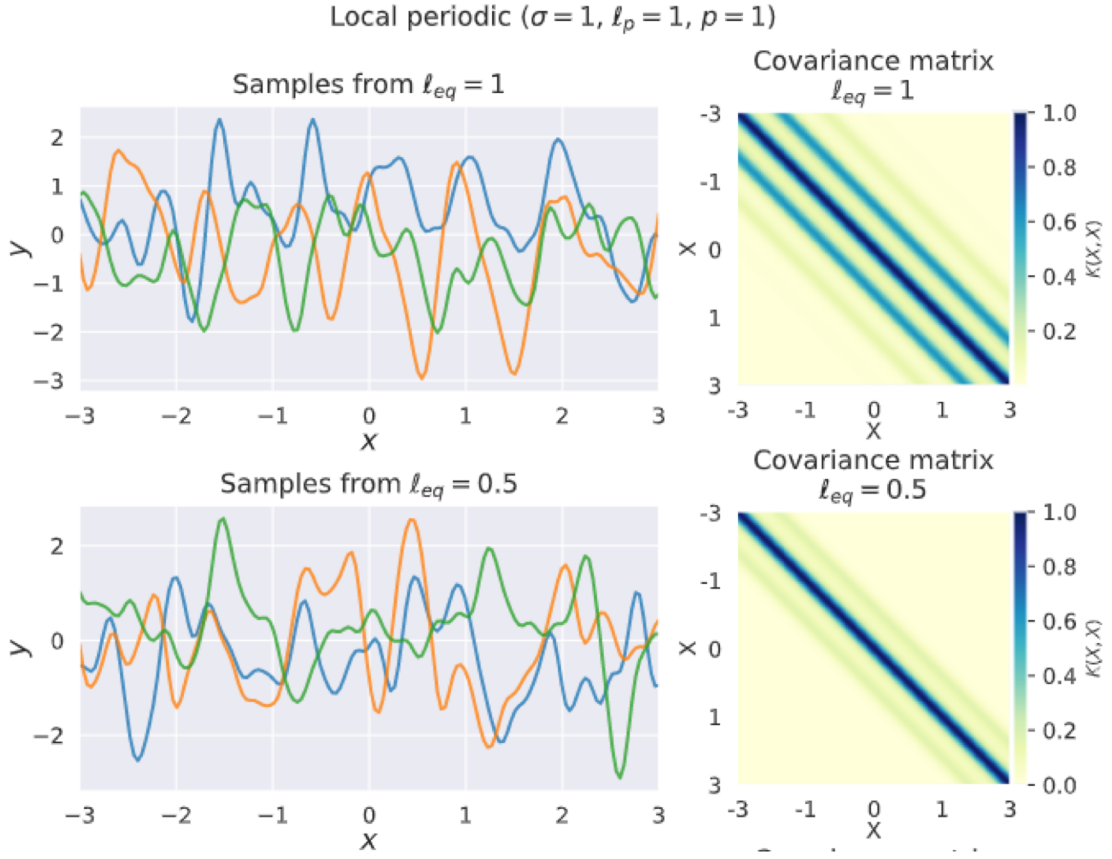
The local periodic kernel is a multiplication of the periodic kernel with the exponentiated quadratic kernel to allow the periods to vary over longer distances. Note that the variance parameters σ^2 are combined into one.

$$k(x_a, x_b) = \sigma^2 \exp\left(-\frac{2}{\ell_p^2}\sin^2\left(\pi\frac{|x_a - x_b|}{p}\right)\right)\exp\left(-\frac{\|x_a - x_b\|^2}{2\ell_{eq}^2}\right)$$

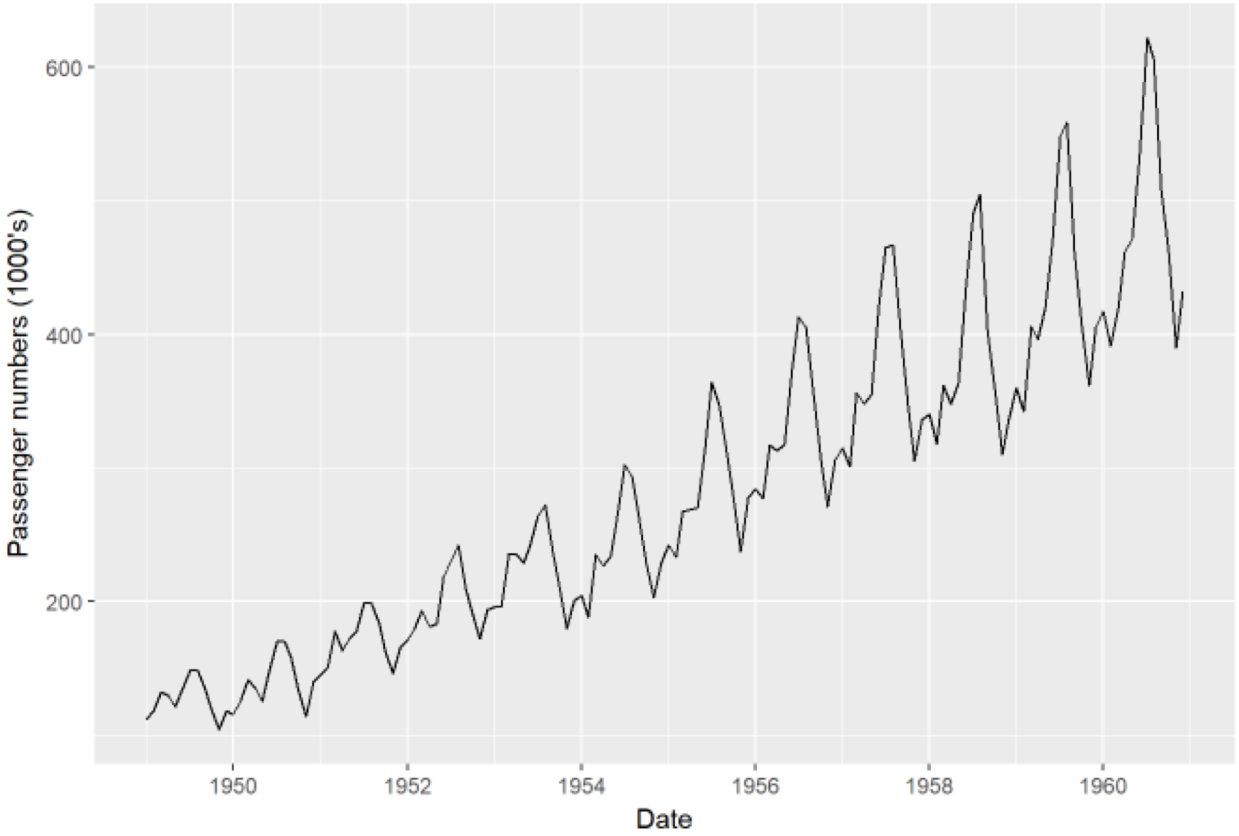
With:

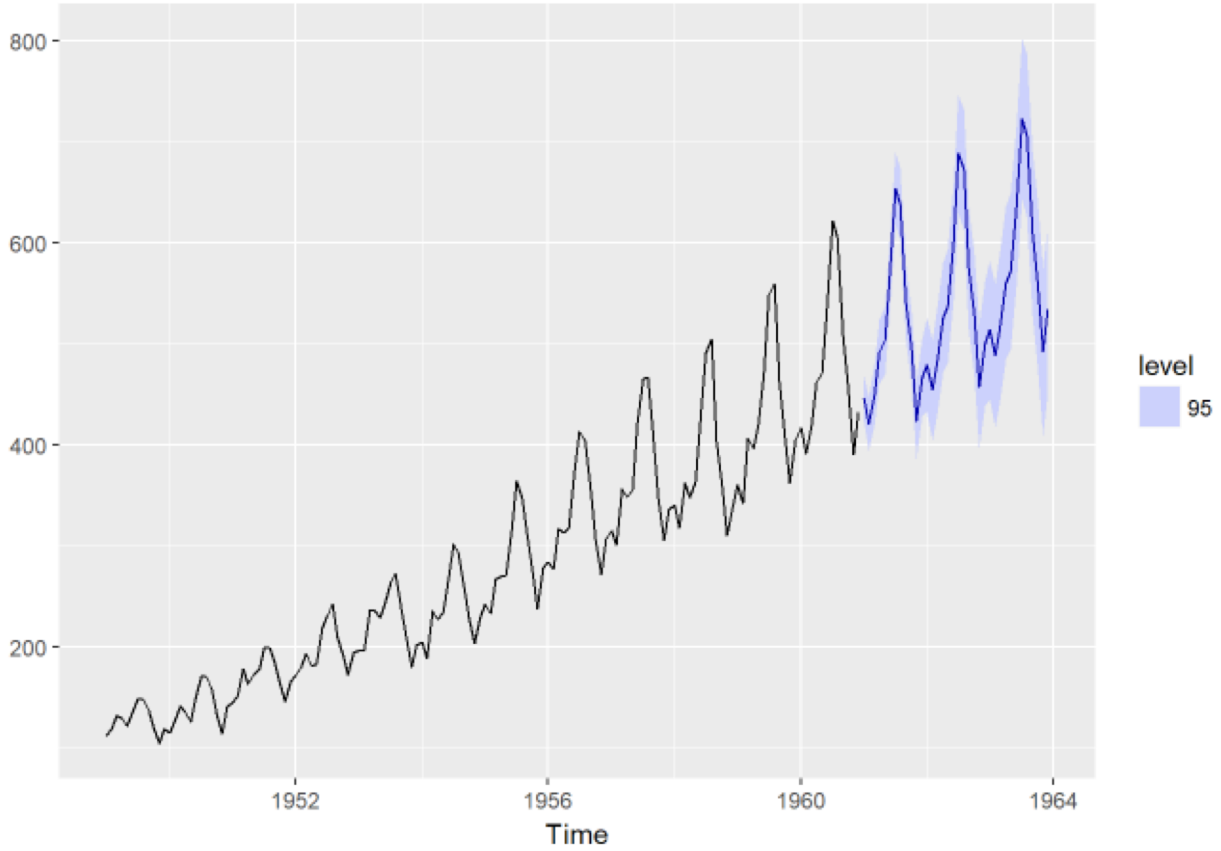
- σ^2 the overall variance (σ is also known as amplitude).
- ℓ_p lengthscale of the periodic function.
- p the period.
- ℓ_{eq} the lengthscale of the exponentiated quadratic.





Example: Airline Passenger Prediction

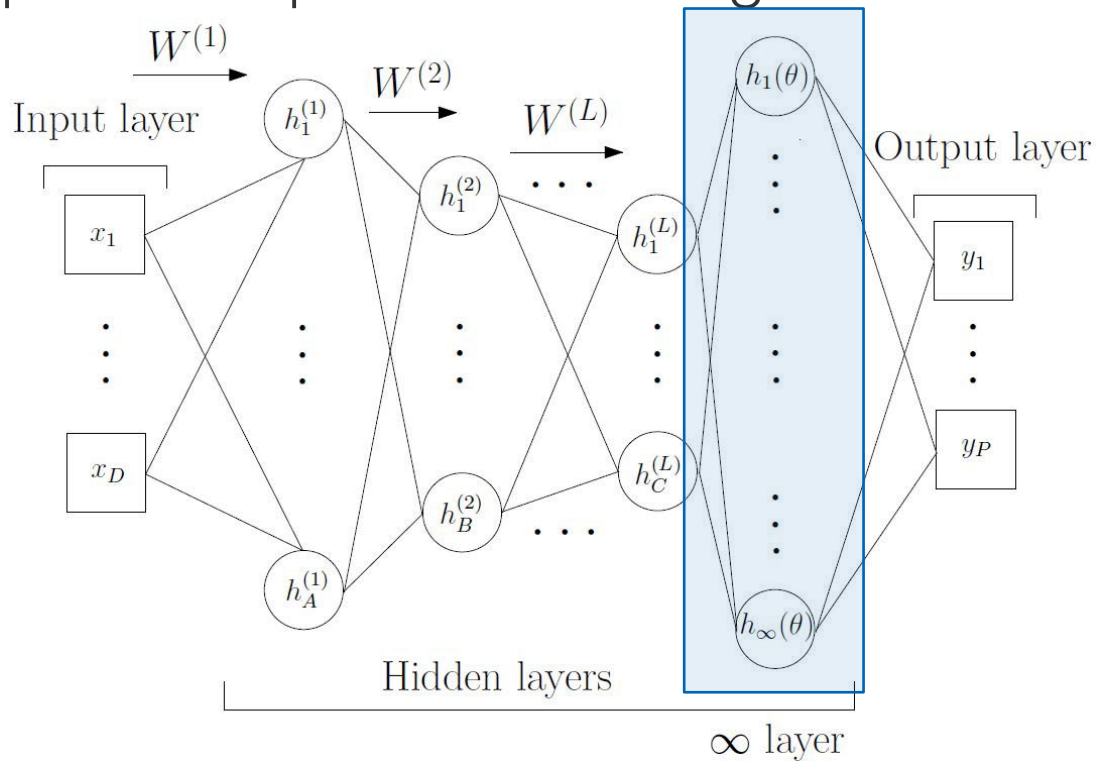




Picture credit: *kimnewzealand (*https://rpubs.com/kimnewzealand/311446)

Gaussian Process and Deep Kernel Learning

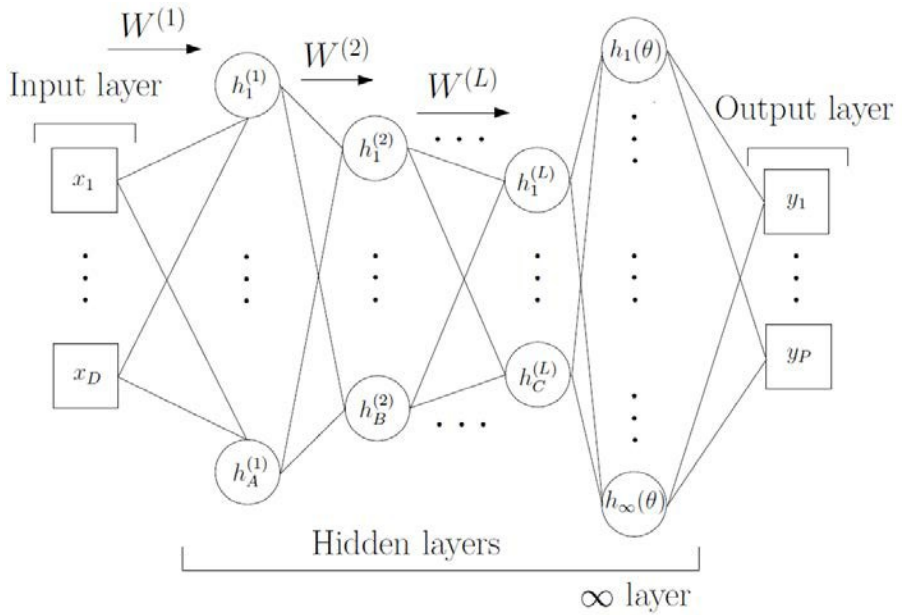
- By adding GP as a layer to a deep neural net, we can think of it as adding an infinite hidden layer with a particular prior on the weights
- Deep kernel learning [Wilson et al., 2016]
  - Combines the inductive biases of deep models with the non-parametric flexibility of Gaussian processes
  - GPs add powerful regularization to the network
  - Additionally, they provide predictive uncertainty estimates



Deep Kernel Learning

- Combines inductive biases of deep learning architectures with the nonparametric flexibility of Gaussian processes.
- Starting from some base kernel, we can get a deep kernel using functional composition:

$$\kappa(x, x') = k(h(x), h(x'))$$



Wilson et al., NIPS 2016

Graph Convolutional Gaussian Processes (ICML 2019)

Given a graph. $G = \langle \mathcal{V}, \mathcal{E} \rangle$ with features in $\mathbb{R}^{|\mathcal{V}| \times d}$, define GP $f : \mathbb{R}^{|\mathcal{V}| \times d} \to \mathbb{R}$

f is a function on graphs, but it ignores graph structure.

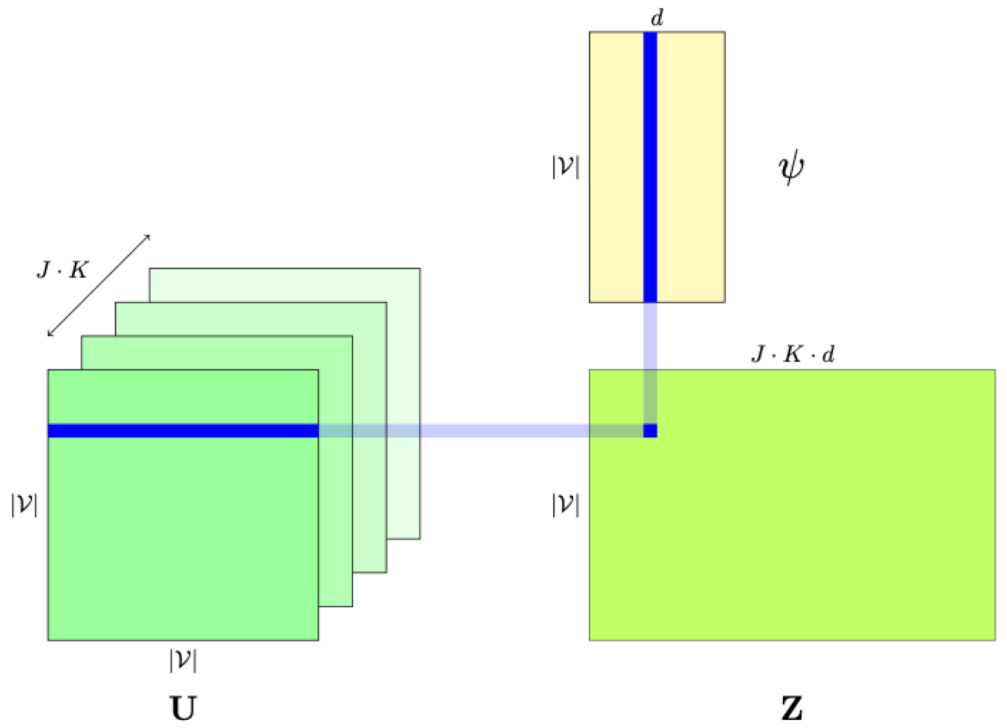
Graph Convolutional Gaussian Processes (ICML 2019)

Given a graph. $G = \langle \mathcal{V}, \mathcal{E} \rangle$ with features in $\mathbb{R}^{|\mathcal{V}| \times d}$, define GP $f : \mathbb{R}^{|\mathcal{V}| \times d} \to \mathbb{R}$

f is a function on graphs, but it ignores graph structure.

$$f(\psi) = h(\mathcal{Z}(\psi)) = h(Z)$$

$$h : \mathbb{R}^{|\mathcal{V}| \times (JKd)} \to \mathbb{R} \qquad \mathcal{Z} : \mathbb{R}^{|\mathcal{V}| \times d} \to \mathbb{R}^{|\mathcal{V}| \times (JKd)}$$



Graph Convolutional Gaussian Processes (ICML 2019)

Given a graph. $G = \langle \mathcal{V}, \mathcal{E} \rangle$ with features in $\mathbb{R}^{|\mathcal{V}| \times d}$, define GP $f : \mathbb{R}^{|\mathcal{V}| \times d} \to \mathbb{R}$

f is a function on graphs, but it ignores graph structure.

$$f(\psi) = h(\mathcal{Z}(\psi)) = h(Z)$$

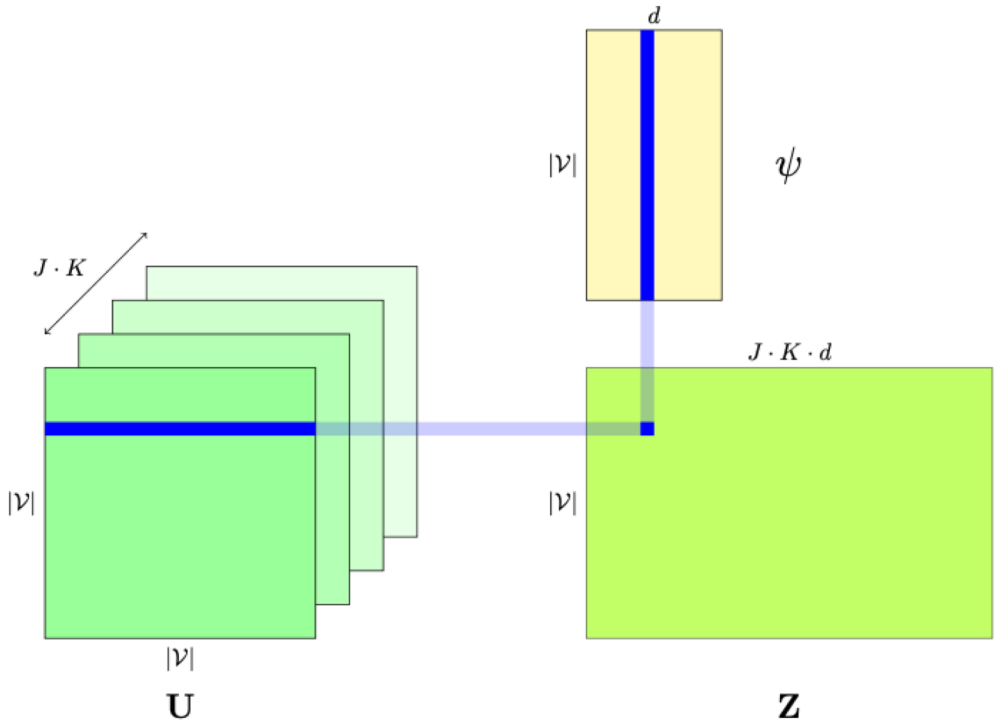
$$h : \mathbb{R}^{|\mathcal{V}| \times (JKd)} \to \mathbb{R} \qquad \mathcal{Z} : \mathbb{R}^{|\mathcal{V}| \times d} \to \mathbb{R}^{|\mathcal{V}| \times (JKd)}$$

Different $\mathbf{u}_{\mathbf{j},\mathbf{k}}(\mathbf{x}, \mathbf{x}')$ can be used for Z

- Manifold:

$$\mathbf{u}_{\mathbf{j},\mathbf{k}}(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{(\rho(\mathbf{x}, \mathbf{x}') - \rho_{\mathbf{k}})^2}{2\sigma_\rho^2}\right)\exp\left(-\frac{(\theta(\mathbf{x}, \mathbf{x}') - \theta_{\mathbf{j}})^2}{2\sigma_\theta^2}\right)$$

where $\rho(\cdot)$ measures the intrinsic radial distance and $\theta(\cdot)$ measures the intrinsic angular distance from x to x'.



Graph Convolutional Gaussian Processes (ICML 2019)

Given a graph. $G = \langle \mathcal{V}, \mathcal{E} \rangle$ with features in $\mathbb{R}^{|\mathcal{V}| \times d}$, define GP $f : \mathbb{R}^{|\mathcal{V}| \times d} \to \mathbb{R}$

f is a function on graphs, but it ignores graph structure.

$$f(\psi) = h(\mathcal{Z}(\psi)) = h(Z)$$

$$h : \mathbb{R}^{|\mathcal{V}| \times (JKd)} \to \mathbb{R} \qquad \mathcal{Z} : \mathbb{R}^{|\mathcal{V}| \times d} \to \mathbb{R}^{|\mathcal{V}| \times (JKd)}$$

Different $\mathbf{u}_{\mathbf{j},\mathbf{k}}(\mathbf{x}, \mathbf{x}')$ can be used for Z

- Manifold:

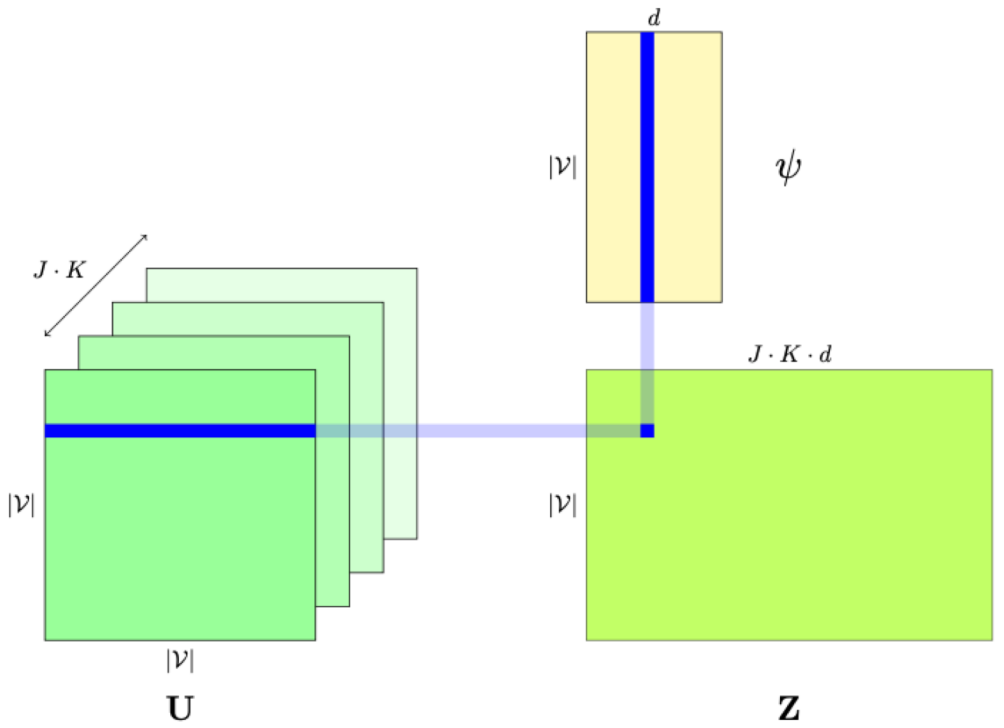
$$\mathbf{u}_{\mathbf{j},\mathbf{k}}(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{(\rho(\mathbf{x}, \mathbf{x}') - \rho_{\mathbf{k}})^2}{2\sigma_\rho^2}\right)\exp\left(-\frac{(\theta(\mathbf{x}, \mathbf{x}') - \theta_{\mathbf{j}})^2}{2\sigma_\theta^2}\right)$$

where $\rho(\cdot)$ measures the intrinsic radial distance and $\theta(\cdot)$ measures the intrinsic angular distance from x to x'.

- General graphs:

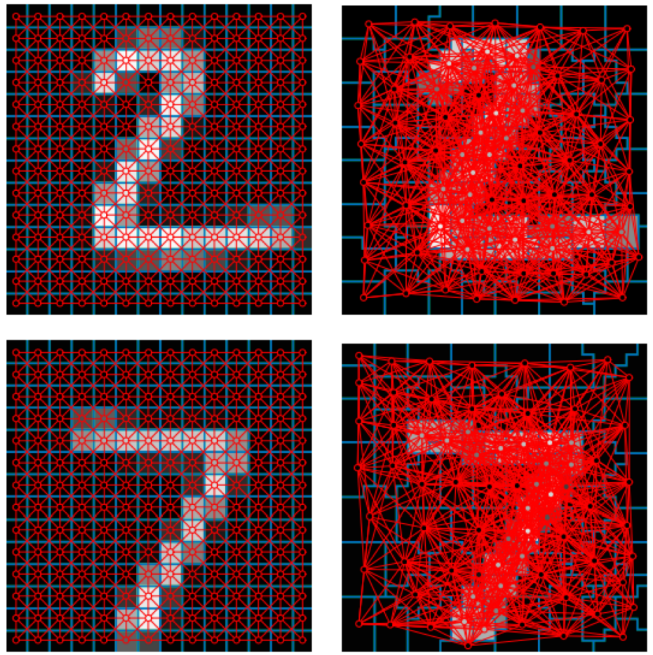
$$\mathbf{u}_{\mathbf{j}}(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{1}{2}(\rho(\mathbf{x}, \mathbf{x}') - \rho_{\mathbf{j}})\boldsymbol{\Sigma}_{\mathbf{j}}^{-1}(\rho(\mathbf{x}, \mathbf{x}') - \rho_{\mathbf{j}})\right)$$

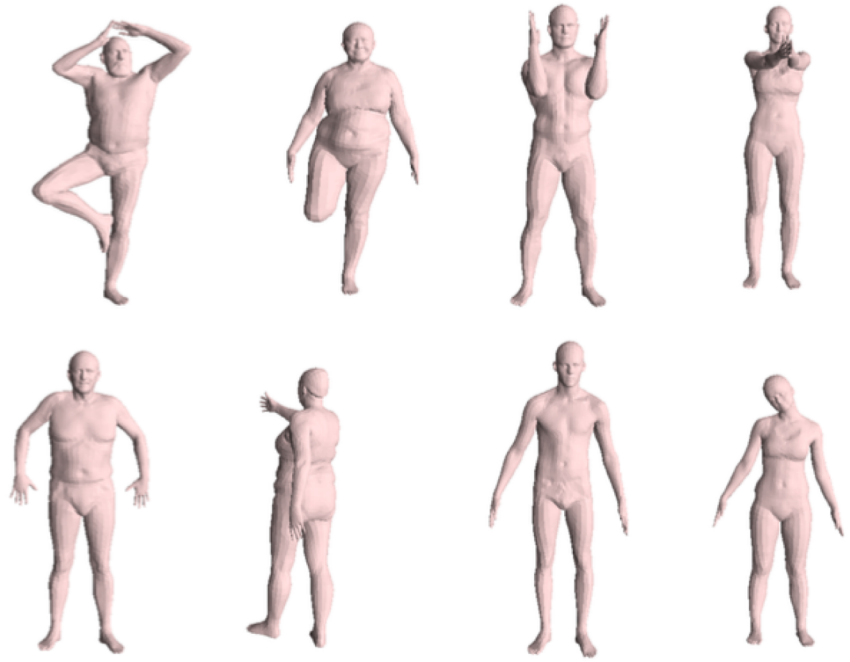
$$\rho(\mathbf{x}, \mathbf{x}') = (\mathbf{deg}(\mathbf{x})^{-1/2}, \mathbf{deg}(\mathbf{x}')^{-1/2})$$



Graph Convolutional Gaussian Processes (ICML 2019)

Application: superpixel digit classification and pose classification





MNIST Superpixel 75

ChebNet (Defferrard et al., 2016)	24.4%
MoNet (Monti et al., 2017)	8.9%
GCGP	**4.2%**

Table 3. Error rates on MPI Faust mesh classification

Number of vertices	500	1000	2500
MoNet	40.00%	33.33%	33.33%
GCGP	**23.33%**	**10.00%**	**3.33%**

Summary

- Gaussian process are Bayesian nonparametric models that can represent distributions over smooth functions.
- Inference can be done fully analytically (in case of Gaussian likelihood).
- Gaussian process can be combined with different kernel functions and even deep neural networks.

Determinantal Point Process (DPP)

Subset Selection

- Given a set of K items, select a subset from them
- Extractive document summarization: select a subset of sentences to summarize a document
- Feature selection: select a subset of features to speed up computation
- Product recommendation: recommend a subset of products to users

Diverse Subset Selection

- We prefer selected items to be mutually dissimilar from each other
- **Diverse** extractive document summarization: a subset of sentences with different meanings
- **Diverse** feature selection: a subset of features represent different perspectives of the data
- **Diverse** product recommendation: no redundant products from the same category

Determinantal Point Process (DPP)

- A distribution over subsets and favors "diverse" subsets
- Given a set of K items $\mathcal{Y} = \{a_i\}_{i=1}^{K}$, where a_i is the feature representation of item i
- Calculate a K-by-K Gram matrix L, where $L_{ij} =< a_i, a_j >$.
- The probability of a subset of items $\mathbf{Y} \subseteq \mathcal{Y}$ is defined as

$$p(Y) = \frac{\det(L_Y)}{\det(L + I)}$$

  - L_Y is a sub-matrix of L where the rows and columns are indexed by elements in Y.
  - $\det(\cdot)$ denotes the determinant of a matrix
  - I is an identity matrix

Determinantal Point Process (DPP)

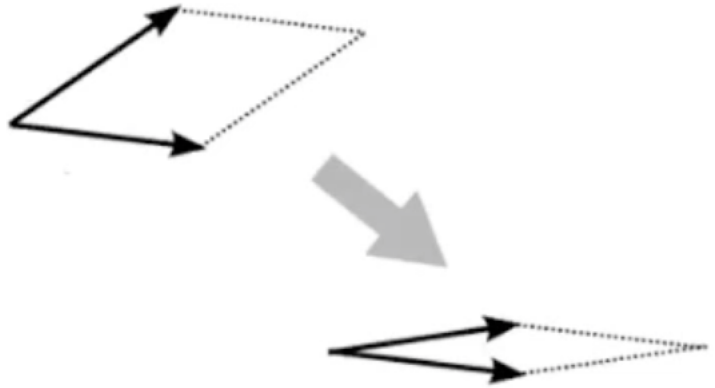
$$Y = \{2,3\} \subseteq \{1,2,3,4\}$$

$$L = \begin{bmatrix} L_{11} & L_{12} & L_{13} & L_{14} \\ L_{21} & L_{22} & L_{23} & L_{24} \\ L_{31} & L_{32} & L_{33} & L_{34} \\ L_{41} & L_{42} & L_{43} & L_{44} \end{bmatrix}$$

$$p(Y) = \frac{\det(L_Y)}{\det(L + I)}$$

Diversity and DPP

- DPP has an induction bias of diversity: it has higher probability mass on a subset where the items are more diverse。
- Why?
  - $\det(L_Y)$ is the volume of the parallelepiped formed by vectors in the subset indexed by Y



Partition Function (Normalizing Factor) of DPP

- Has a closed-form $\det(L + I)$
- How to prove $\det(L + I)$ is the partition function?
  - Namely $\det(L + I) = \sum_{Y \subseteq \mathcal{Y}} \det(L_Y)$
    - We first prove the following theorem

Theorem 1 For any $A \subseteq \mathcal{Y}$,

$$\sum_{A \subseteq Y \subseteq \mathcal{Y}} \det(L_Y) = \det(L + I_{\bar{A}}),$$

where $I_{\bar{A}}$ is the diagonal matrix with ones in the diagonal positions corresponding to elements of $\bar{A} = \mathcal{Y} - A$, and zeros everywhere else.

$\mathcal{Y} = \{1,2,3,4\}$
A={2,4}

$$\begin{matrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{matrix}$$

- Let A be an empty set, then $\det(L + I) = \sum_{Y \subseteq \mathcal{Y}} \det(L_Y)$

Kernel Version of DPP

~~Calculate a K-by-K Gram matrix L, where $L_{ij} =< a_i, a_j >$.~~

Calculate a K-by-K kernel matrix L, where $L_{ij} = l(a_i, a_j) =< \phi(a_i), \phi(a_j) >$

- $l(\cdot,\cdot)$ is a kernel function.

$$p(Y) = \frac{\det(L_Y)}{\det(L + I)}$$

Deep DPP

- Calculate a K-by-K kernel matrix L, where $L_{ij} = l(a_i, a_j) = < \phi(a_i), \phi(a_j) >$
  - $l(\cdot,\cdot)$ is a deep kernel.

$$p(Y) = \frac{\det(L_Y)}{\det(L + I)}$$

A deep neural network

Inference: Find the Mode of DPP

- $Y^* = \text{argmax}_Y \text{ logdet}(L_Y)$
- Exact solution is NP hard.
- Approximate algorithm (Gillenwater et al., 2012)
  - Continuous relaxation of item selection variables: {0,1} → [0,1]
  - Continuous relaxation of logdet(L_Y): F(y)=logdet(diag(y)(L-I)+I)
  - Find $y^* = \text{argmax}_y$ F(y)
  - Rounding elements in y* to {0,1}

Learning: Hyperparameters in the Kernel

- Learn the hyperparameters in the kernel function
- Observations: (universal set $\mathcal{Y}$, selected subset Y)
  - E.g., in the training data of extraction summarization, the groundtruth sentences are given
  - Maximum log-likelihood

$$\log \prod_{i=1}^{N} p(Yi|\mathcal{Y}_i) = \sum_{i=1}^{N} \log \det(L_{Y_i}) - \log\det(L_{\mathcal{Y}_i} + I_{\mathcal{Y}_i})$$

Conditional DPP

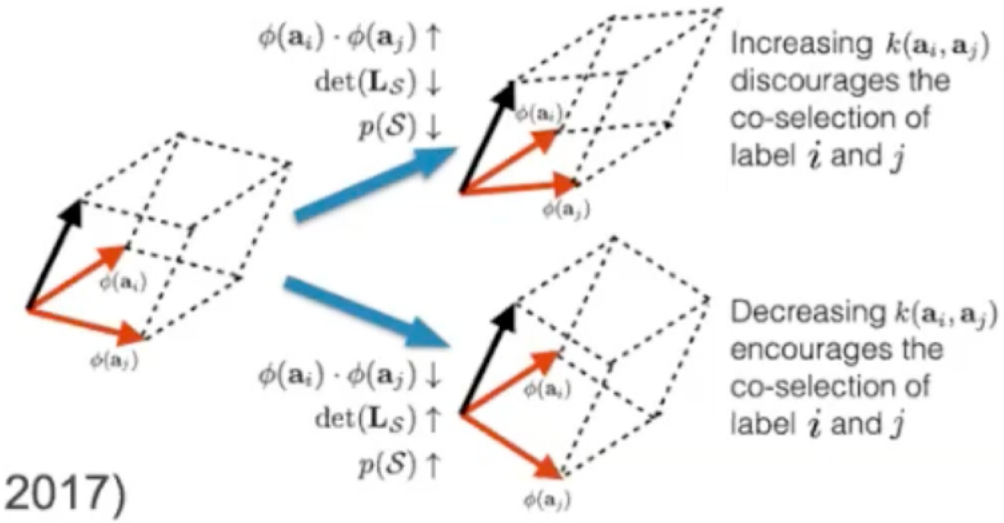
- Personalized product recommendation: given a user x and a set of products $\mathcal{Y}$, select a subset of products Y that are not only diverse, but also relevant to the user

- Calculate a K-by-K kernel matrix L(x), where $L_{ij}(x) = l(a_i, a_j|x)$
  - E.g. $l(a_i, a_j|x) = \mathrm{g}(a_i, x)s(a_i, a_j)\mathrm{g}(a_j, x)$

$$p(Y|x) = \frac{\det(L_Y(x))}{\det(L(x) + I)}$$

Case Study: Deep DPP for Multi-label Classification

- Multi-label classification (MLC): given an input, select a subset of labels
- Deep DPP for MLC $p(\mathcal{S}|x) = \frac{\det(\mathbf{L}_{\mathcal{S}}(x))}{\det(\mathbf{L}(x)+\mathbf{I})}$
- Incorporate prior knowledge
  - Cannot links and must links



(Xie et al., 2017)

$$\max_{\Theta} \quad \mathcal{L}(\{(x_n,\mathcal{S}_n)\}_{n=1}^{N}) + \lambda(-\sum_{(i,j)\in\mathcal{M}} k(\mathbf{a}_i,\mathbf{a}_j) + \sum_{(i,j)\in\mathcal{C}} k(\mathbf{a}_i,\mathbf{a}_j))$$

Reference

https://www.cs.cmu.edu/~epxing/Class/10708-20/lectures.html

http://cbl.eng.cam.ac.uk/pub/Public/Turner/News/slides-imperial.pdf

Walker, I., & Glocker, B. (2019, May). Graph convolutional Gaussian processes. In *International Conference on Machine Learning* (pp. 6495-6504). PMLR.

Xie, P., Salakhutdinov, R., Mou, L., & Xing, E. P. (2017). Deep determinantal point process for large-scale multi-label classification. In *Proceedings of the IEEE International Conference on Computer Vision* (pp. 473-482).

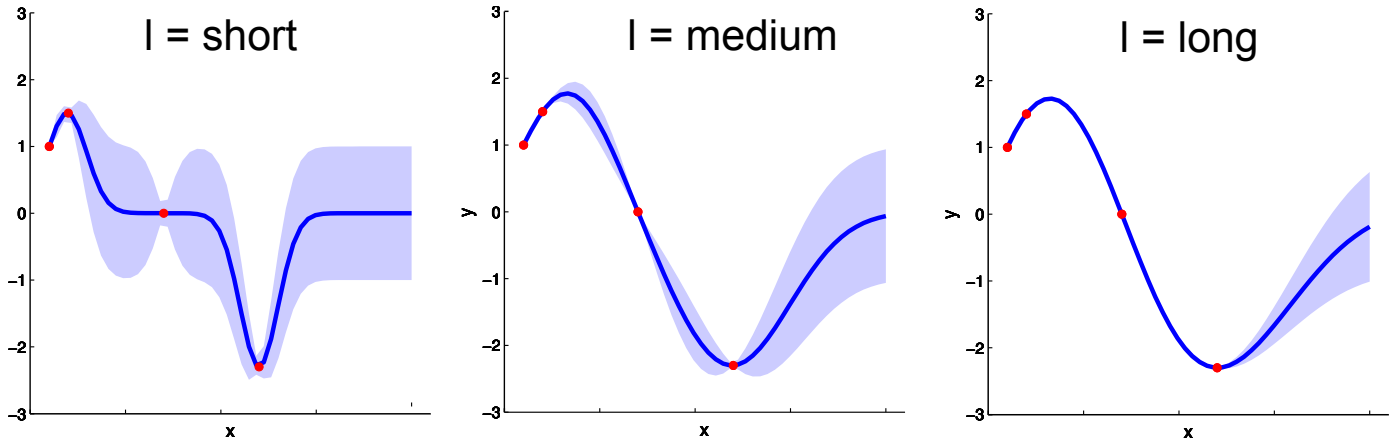
Appendix

What effect do the hyper-parameters have?

$$\mathsf{K}(\mathsf{x}_1, \mathsf{x}_2) = \sigma^2 \exp\left(-\frac{1}{2\mathsf{l}^2}(\mathsf{x}_1 - \mathsf{x}_2)^2\right)$$

- Hyper-parameters have a strong effect
  - l controls the horizontal length-scale
  - σ^2 controls the vertical scale of the data
- $\Longrightarrow$ need automatic learning of hyper-parameters from data e.g.

$$\arg\max_{l,\sigma^2} \log p(\mathbf{y}\,|\theta)$$



Regression: probabilistic inference in function space

Non-parametric (∞-parametric)

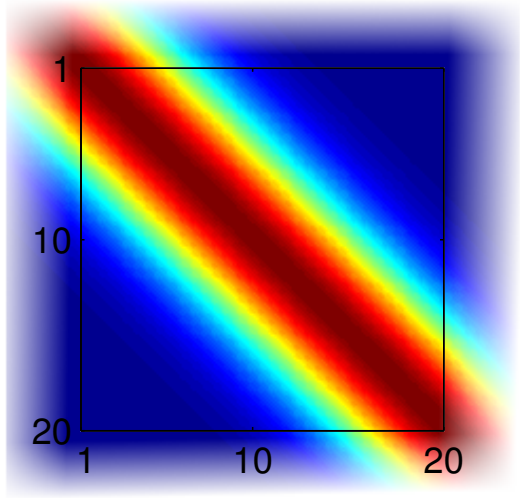
$$p(\mathsf{y}|\theta) = \mathcal{N}(0, \Sigma)$$

$$\Sigma(\mathsf{x}_1, \mathsf{x}_2) = \mathsf{K}(\mathsf{x}_1, \mathsf{x}_2) + \mathsf{I}\sigma_{\mathsf{y}}^2$$

$$\mathsf{K}(\mathsf{x}_1, \mathsf{x}_2) = \sigma^2 \mathrm{e}^{-\frac{1}{2l^2}(\mathsf{x}_1 - \mathsf{x}_2)^2}$$

vertical-scale horizontal-scale

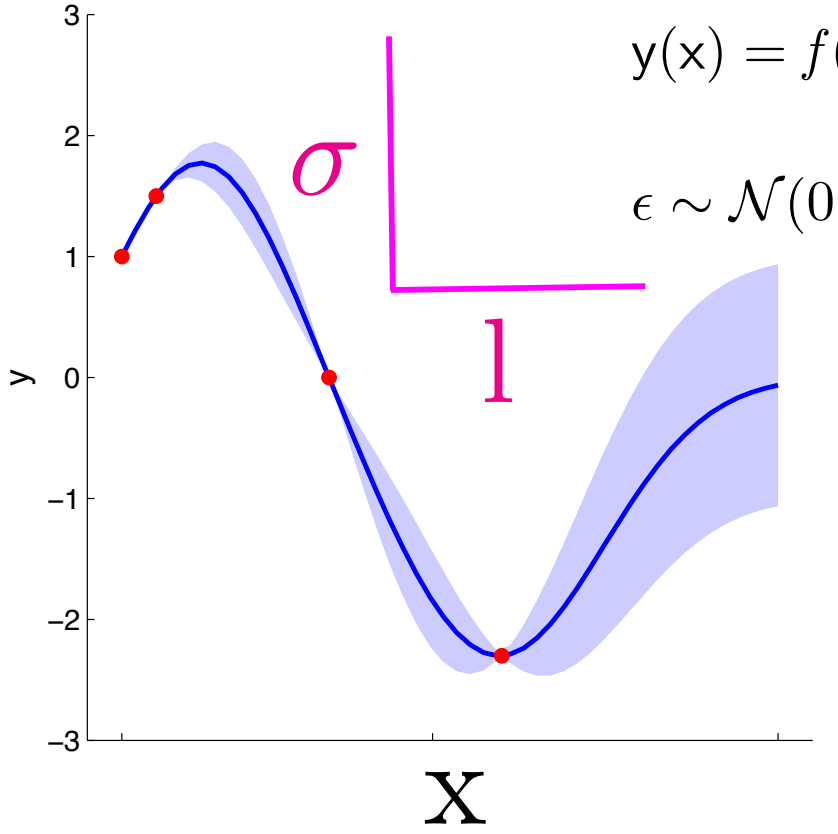
$\Sigma =$



Parametric model

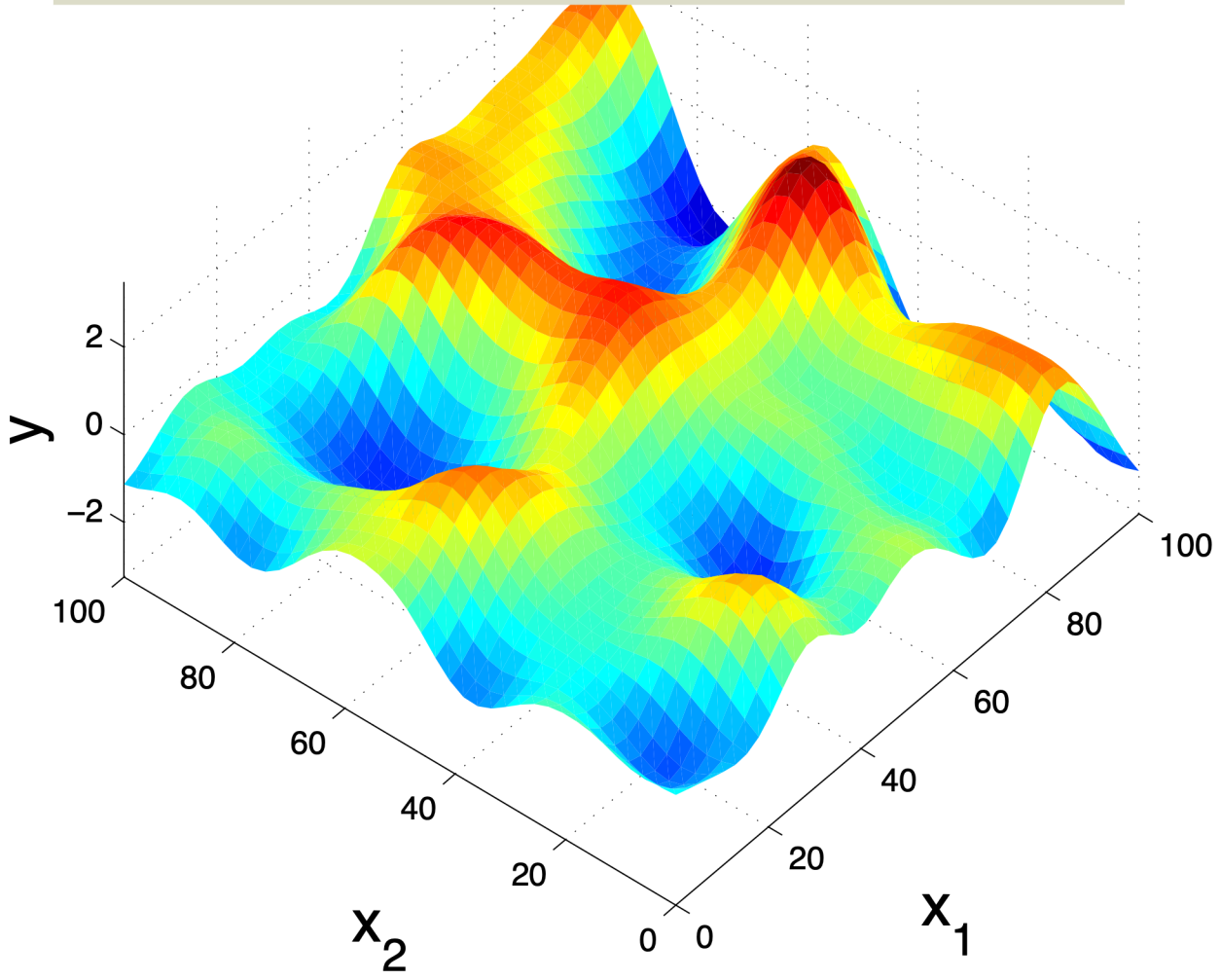
$$\mathsf{y}(\mathsf{x}) = f(\mathsf{x}; \theta) + \sigma_{\mathsf{y}}\epsilon$$

$$\epsilon \sim \mathcal{N}(0, 1)$$

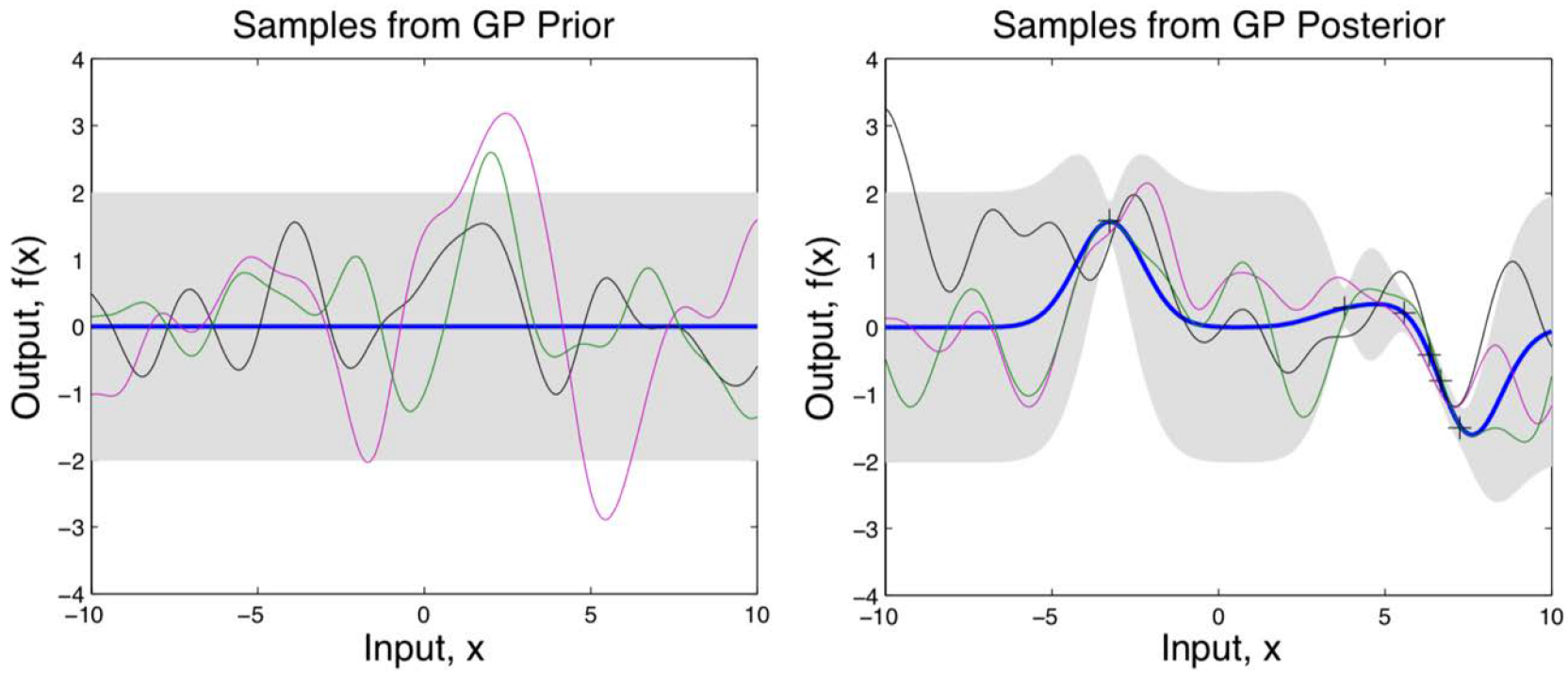


Higher dimensional input spaces

$$K(\mathbf{x}, \mathbf{x}') = \sigma^2 \exp\left(-\frac{1}{2l_1^2}(\mathsf{x}_1 - \mathsf{x}_1')^2 - \frac{1}{2l_2^2}(\mathsf{x}_2 - \mathsf{x}_2')^2\right)$$



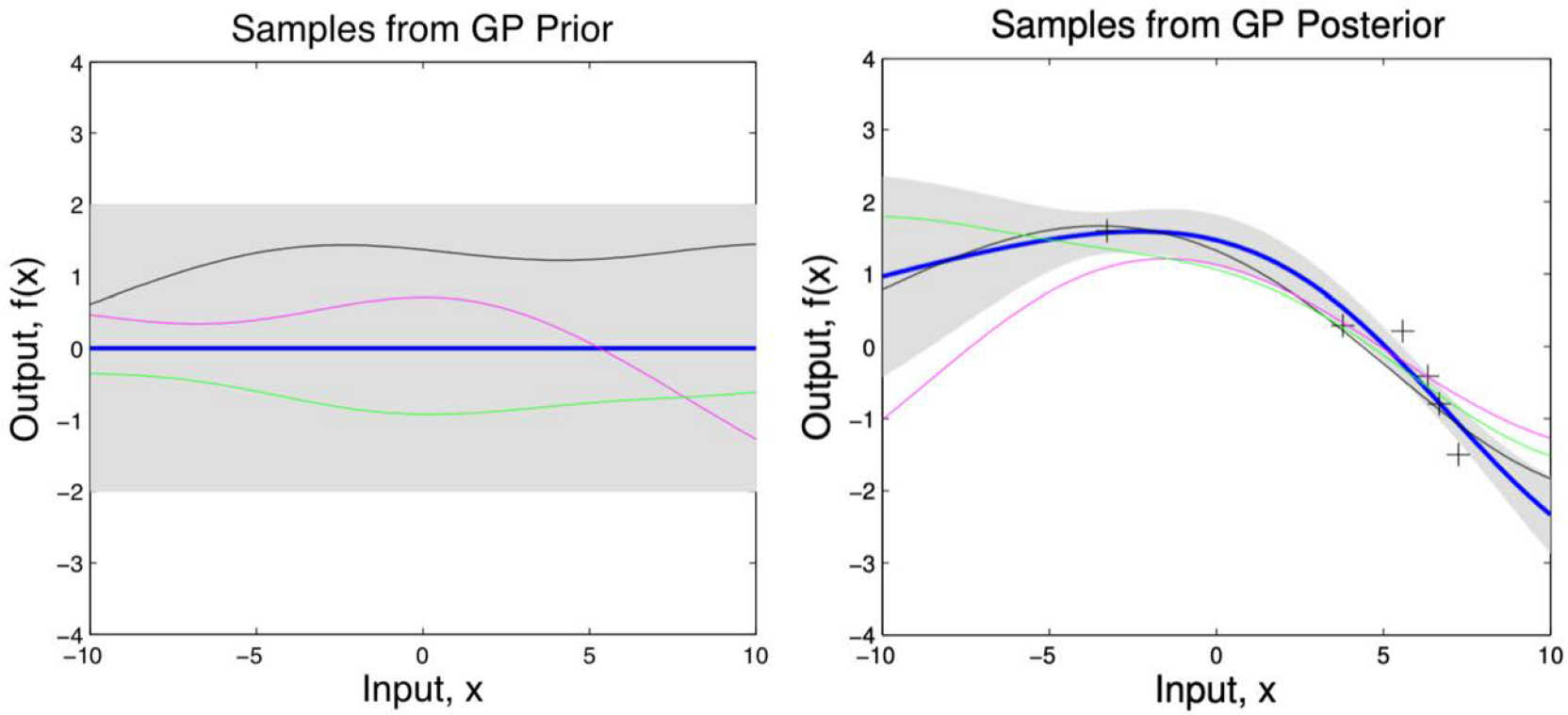
Gaussian Process Inference



Slide credit: Andrew G. Wilson

Gaussian Process Inference

Increase the length-scale ℓ.



Slide credit: Andrew G. Wilson

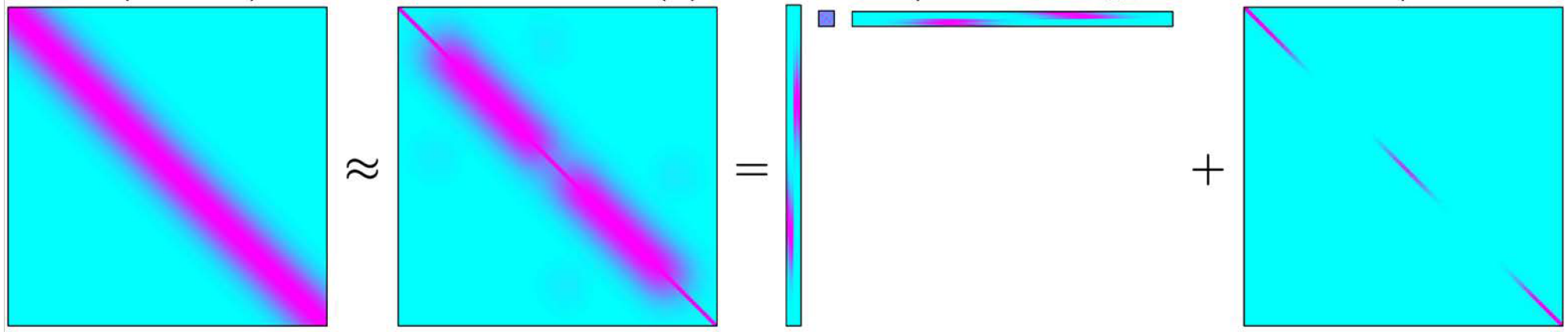
The Scalability Issue

- Computational bottlenecks for GPs:
  - Inference: $(K_\theta + \sigma^2 I)^{-1}\mathbf{y}$ for $n \times n$ matrix K.
  - Learning: $\log |K_\theta + \sigma^2 I|$, for marginal likelihood evaluations needed to learn θ.
- Both inference and learning naively require $\mathcal{O}(n^3)$ operations and $\mathcal{O}(n^2)$ storage (typically from computing a Cholesky decomposition of K). Afterwards, the predictive mean and variance cost $\mathcal{O}(n)$ and $\mathcal{O}(n^2)$ per test point.

Inducing Point Methods

We can approximate GP through $M < N$ inducing points $\bar{\mathbf{f}}$ to obtain this Sparse Pseudo-input Gaussian process (SPGP) prior: $p(\mathbf{f}) = \int \mathrm{d}\bar{\mathbf{f}} \prod_n p(f_n|\bar{\mathbf{f}})\, p(\bar{\mathbf{f}})$

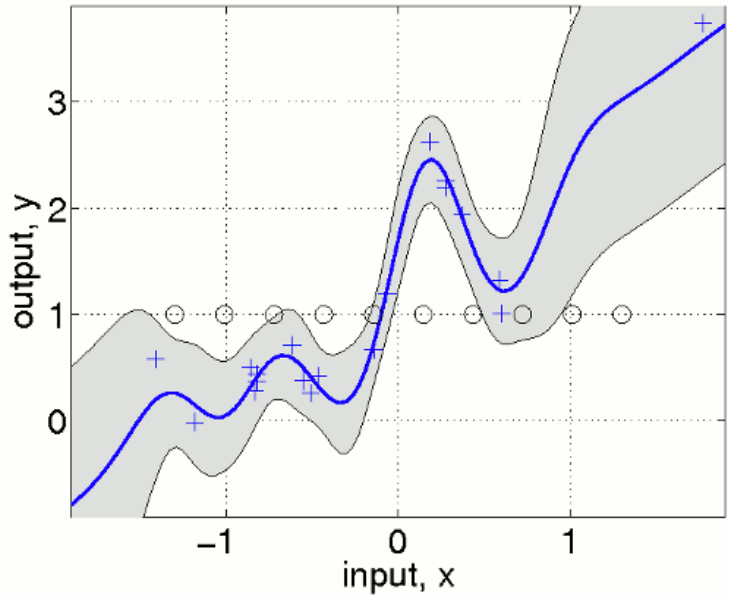
$\mathcal{N}(\mathbf{0}, \mathbf{K}_N) \approx p(\mathbf{f}) = \mathcal{N}(\mathbf{0}, \mathbf{K}_{NM}\mathbf{K}_M^{-1}\mathbf{K}_{MN} + \mathbf{\Lambda})$

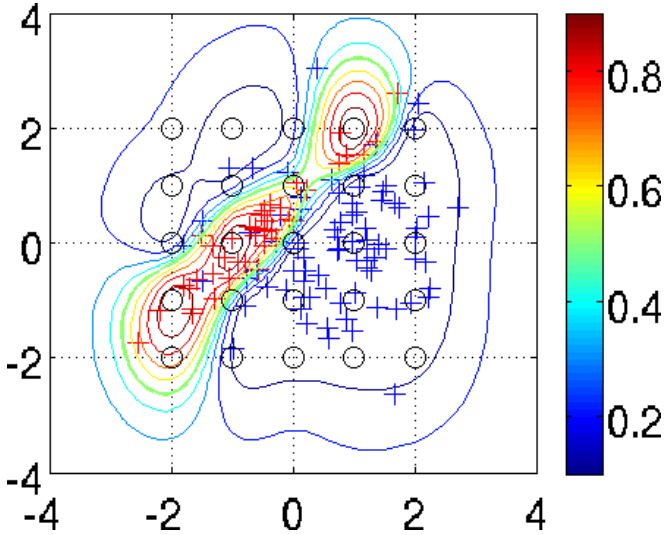


- SPGP covariance inverted in $\mathcal{O}(M^2N) \ll \mathcal{O}(N^3) \Rightarrow$ much faster

Slide credit: Zoubin Ghahramani (MLSS, 2011)

Inducing Point Methods





Grids are tricky:
In high dimensions, one would need a LOT of inducing points to build a high-dimensional grid.
This might drastically affect efficiency.

Further reading:
Wilson, Dann, Nickisch (2015). Thoughts on Massively Scalable Gaussian Processes
Bauer, van der Wilk, Rasmussen (2016). Understanding Probabilistic Sparse Gaussian Process Approximations.

Images: GPML library docs

Proof of Theorem 1

- Proof by induction
- Base case: A= y. The proof is trivial.
- Induction step.

Induction Step

where $I_{\bar{A}}$ is the diagonal matrix with ones in the diagonal positions corresponding to elements of $\bar{A} = \mathcal{Y} - A$, and zeros everywhere else.

suppose inductively that the theorem holds whenever $\bar{A}$ has cardinality less than k. Given A such that $|\bar{A}| = k > 0$, let i be an element of $\mathcal{Y}$ where $i \in \bar{A}$. Splitting blockwise according to the partition $\mathcal{Y} = \{i\} \cup \mathcal{Y} - \{i\}$, we can write

$$L + I_{\bar{A}} = \begin{pmatrix} L_{ii} + 1 & L_{i\bar{i}} \\ L_{\bar{i}i} & L_{\mathcal{Y}-\{i\}} + I_{\mathcal{Y}-\{i\}-A} \end{pmatrix}, \tag{2.8}$$

$$L = \begin{bmatrix} L_{ii} & L_{i\bar{i}} \\ L_{\bar{i}i} & L_{\mathcal{Y}-\{i\}} \end{bmatrix}$$

$$I_{\bar{A}} = \begin{bmatrix} 1 & 0 \\ 0 & I_{\mathcal{Y}-A-\{i\}} \end{bmatrix}$$

where $L_{\bar{i}i}$ is the subcolumn of the i-th column of L whose rows correspond to $\bar{i}$, and similarly for $L_{i\bar{i}}$. By multilinearity of the determinant, then,

$$\det(L + I_{\bar{A}}) = \begin{vmatrix} L_{ii} & L_{i\bar{i}} \\ L_{\bar{i}i} & L_{\mathcal{Y}-\{i\}} + I_{\mathcal{Y}-\{i\}-A} \end{vmatrix} + \begin{vmatrix} 1 & \mathbf{0} \\ L_{\bar{i}i} & L_{\mathcal{Y}-\{i\}} + I_{\mathcal{Y}-\{i\}-A} \end{vmatrix} \tag{2.9}$$

$$= \det(L + I_{\overline{A \cup \{i\}}}) + \det(L_{\mathcal{Y}-\{i\}} + I_{\mathcal{Y}-\{i\}-A}). \tag{2.10}$$

Induction Step

$$\det(r_1, \ldots, r_{i-1}, s\,r_i + r_i', r_{i+1}, \ldots, r_n) = s\,\det(r_1, \ldots, r_{i-1}, r_i, r_{i+1}, \ldots, r_n) + \det(r_1, \ldots, r_{i-1}, r_i', r_{i+1}, \ldots, r_n).$$

suppose inductively that the theorem holds whenever $\bar{A}$ has cardinality less than k. Given A such that $|\bar{A}| = k > 0$, let i be an element of $\mathcal{Y}$ where $i \in \bar{A}$. Splitting blockwise according to the partition $\mathcal{Y} = \{i\} \cup \mathcal{Y} - \{i\}$, we can write

$$L = \begin{bmatrix} L_{ii} & L_{i\bar{i}} \\ L_{\bar{i}i} & L_{\mathcal{Y}-\{i\}} \end{bmatrix}$$

$$L + I_{\bar{A}} = \begin{pmatrix} L_{ii} + 1 & L_{i\bar{i}} \\ L_{\bar{i}i} & L_{\mathcal{Y}-\{i\}} + I_{\mathcal{Y}-\{i\}-A} \end{pmatrix}, \tag{2.8}$$

$$I_{\bar{A}} = \begin{bmatrix} 1 & 0 \\ 0 & I_{\mathcal{Y}-A-\{i\}} \end{bmatrix}$$

where $L_{\bar{i}i}$ is the subcolumn of the i-th column of L whose rows correspond to $\bar{i}$, and similarly for $L_{i\bar{i}}$. By multilinearity of the determinant, then,

$$\det(L + I_{\bar{A}}) = \begin{vmatrix} L_{ii} & L_{i\bar{i}} \\ L_{\bar{i}i} & L_{\mathcal{Y}-\{i\}} + I_{\mathcal{Y}-\{i\}-A} \end{vmatrix} + \begin{vmatrix} 1 & \mathbf{0} \\ L_{\bar{i}i} & L_{\mathcal{Y}-\{i\}} + I_{\mathcal{Y}-\{i\}-A} \end{vmatrix} \tag{2.9}$$

$$= \det(L + I_{\overline{A \cup \{i\}}}) + \det(L_{\mathcal{Y}-\{i\}} + I_{\mathcal{Y}-\{i\}-A}). \tag{2.10}$$

Induction Step

suppose inductively that the theorem holds whenever $\bar{A}$ has cardinality less than k. Given A such that $|\bar{A}| = k > 0$, let i be an element of $\mathcal{Y}$ where $i \in \bar{A}$.

$$\sum_{A \subseteq Y \subseteq \mathcal{Y}} \det(L_Y) = \det(L + I_{\bar{A}}),$$

$$\det(L + I_{\bar{A}}) = \begin{vmatrix} L_{ii} & L_{i\bar{i}} \\ L_{\bar{i}i} & L_{\mathcal{Y}-\{i\}} + I_{\mathcal{Y}-\{i\}-A} \end{vmatrix} + \begin{vmatrix} 1 & \mathbf{0} \\ L_{\bar{i}i} & L_{\mathcal{Y}-\{i\}} + I_{\mathcal{Y}-\{i\}-A} \end{vmatrix} \tag{2.9}$$

$$= \det(L + I_{\overline{A \cup \{i\}}}) + \det(L_{\mathcal{Y}-\{i\}} + I_{\mathcal{Y}-\{i\}-A}). \tag{2.10}$$

$|\bar{A}| = k \qquad |A| = K - k$

$|A \cup \{i\}| = K - k + 1$

$|\overline{A \cup \{i\}}| = k - 1$

We can now apply the inductive hypothesis separately to each term, giving

$$\det(L + I_{\bar{A}}) = \sum_{A \cup \{i\} \subseteq Y \subseteq \mathcal{Y}} \det(L_Y) + \sum_{A \subseteq Y \subseteq \mathcal{Y}-\{i\}} \det(L_Y) \tag{2.11}$$

$$= \sum_{A \subseteq Y \subseteq \mathcal{Y}} \det(L_Y), \tag{2.12}$$

where we observe that every Y either contains i and is included only in the first sum, or does not contain i and is included only in the second sum. □