Lecture 4: Parametric and non-parametric regression

- Kernel regression
- RKHS
- Bayesian view
- Gaussian Processes (if we have time)

Recall: Linear regression

- The optimal solution (minimizing sum-squared-error) can be computed in polynomial time in the size of the data set.
- The solution is $\mathbf{w} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$, where $\mathbf{X}$ is the (normalized) data matrix, and $\mathbf{y}$ is the column vector of (centered) target outputs.
- A very rare case in which an analytical, exact solution is possible

Recall: Linear function approximation in general

- Given a set of examples $(\mathbf{x}_i, y_i)_{i=1\dots m}$, we fit a hypothesis

$$h_{\mathbf{w}}(\mathbf{x}) = \sum_{k=1}^{d} w_k \phi_k(\mathbf{x}) = \mathbf{w}^T \phi(\mathbf{x})$$

  where ϕ_k are called *basis functions*

- We define the $\mathbb{R}^{m \times d}$ matrix with one row per instance: $\mathbf{\Phi}_{m,:} = \phi(\mathbf{x}_m)^\top$

$$\mathbf{\Phi} = \begin{bmatrix} \phi_1(\mathbf{x}_1) & \phi_2(\mathbf{x}_1) & \dots & \phi_d(\mathbf{x}_1) \\ \phi_1(\mathbf{x}_2) & \phi_2(\mathbf{x}_2) & \dots & \phi_d(\mathbf{x}_2) \\ \vdots & & \vdots & \vdots \\ \phi_1(\mathbf{x}_m) & \phi_2(\mathbf{x}_m) & \dots & \phi_d(\mathbf{x}_m) \end{bmatrix}$$

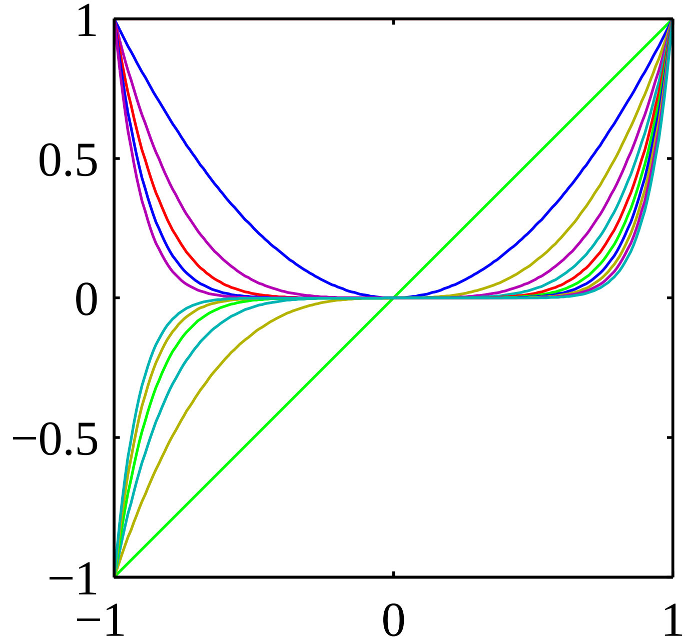
- The hypothesis can alternatively be written as:

$$h_{\mathbf{w}}(\mathbf{x}) = \mathbf{\Phi}\mathbf{w}$$

Basis functions

- Basis functions are *fixed*
- Assumption: $f(\mathbf{x})$ can be modelled by the set of weighted basis function
- Basis functions implement a form of prior knowledge

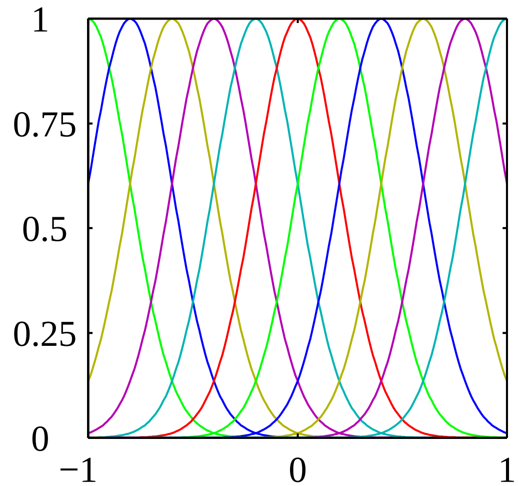
Example basis functions: Polynomials



$$\phi_k(x) = x^k$$

"Global" functions: a small change in x may cause large change in the output of many basis functions.

Example basis functions: Gaussian



$$\phi_k(x) = \exp\left(-\frac{(x-\mu_k)^2}{2\sigma^2}\right)$$

- μ_k controls the position along the x-axis
- σ controls the width (activation radius)
- Usually thought as "local" functions: if σ is relatively small, a small change in x only causes a change in the output of a few basis functions (the ones with means close to x)

Recall: Solving linear models

- By linear models, we mean that the hypothesis function $h_{\mathbf{w}}(\mathbf{x})$ is a *linear function of the parameters* $\mathbf{w}$
- The best $\mathbf{w}$ is considered the one which minimizes the sum-squared error over the training data:

$$\sum_{i=1}^{m}(y_i - h_{\mathbf{w}}(\mathbf{x}_i))^2$$

- We can find the best $\mathbf{w}$ in closed form:

$$\mathbf{w} = (\mathbf{\Phi}^T\mathbf{\Phi})^{-1}\mathbf{\Phi}^T\mathbf{y}$$

- This solution may overfit
- $\mathbf{\Phi}^T\mathbf{\Phi}$ may not be invertible

Regularized solution (Ridge)

- Regularization parameter $\lambda \geq 0$
- Minimize
$$J_\lambda(\mathbf{w}) = \frac{1}{2}(\mathbf{\Phi w} - \mathbf{y})^\top(\mathbf{\Phi w} - \mathbf{y}) + \frac{\lambda}{2}\mathbf{w}^\top\mathbf{w}$$
- Optimal solution (obtained by solving $\nabla J_\lambda(\mathbf{w}) = 0$)
$$\mathbf{w} = (\mathbf{\Phi}^\top\mathbf{\Phi} + \lambda\mathbf{I})^{-1}\mathbf{\Phi}^\top\mathbf{y}$$
- $\mathbf{\Phi}^\top\mathbf{\Phi} + \lambda\mathbf{I}$ is now invertible

Parametric regression

- Compute
$$\mathbf{w} = (\mathbf{\Phi}^\top \mathbf{\Phi} + \lambda \mathbf{I})^{-1} \mathbf{\Phi}^\top \mathbf{y}$$
- Make prediction at new point $\mathbf{x}$: $\hat{f}(\mathbf{x}) = \phi(\mathbf{x})^\top \mathbf{w}$
- Requires to explicit the matrix $\mathbf{\Phi}$ of size $m \times d$
- Requires to compute a matrix $\mathbf{\Phi}^\top \mathbf{\Phi}$ of size $d \times d$

$\Rightarrow$ Parametric regression scales with the number of parameters d

$\Rightarrow$ What if $d \to \infty$?

Non-parametric regression

- Using the identity $(\mathbf{M}^\top \mathbf{M} + \alpha \mathbf{I})^{-1}\mathbf{M}^\top = \mathbf{M}^\top(\mathbf{M}\mathbf{M}^\top + \alpha \mathbf{I})^{-1}$, the solution can be rewritten as

$$\mathbf{w} = (\mathbf{\Phi}^\top \mathbf{\Phi} + \lambda \mathbf{I}_d)^{-1}\mathbf{\Phi}^\top \mathbf{y} = \mathbf{\Phi}^\top(\mathbf{\Phi}\mathbf{\Phi}^\top + \lambda \mathbf{I}_m)^{-1}\mathbf{y}$$

$\Rightarrow$ The solution $\mathbf{w}$ is a linear combination of input points!

$\Rightarrow$ Exercise: Prove that $(\mathbf{M}^\top \mathbf{M} + \alpha \mathbf{I})^{-1}\mathbf{M}^\top = \mathbf{M}^\top(\mathbf{M}\mathbf{M}^\top + \alpha \mathbf{I})^{-1}$

Non-parametric regression (cont'd)

- The predictions for the input data are given by

$$\hat{\mathbf{y}} = \mathbf{\Phi}\mathbf{w} = \mathbf{\Phi}\mathbf{\Phi}^\top(\mathbf{\Phi}\mathbf{\Phi}^\top + \lambda\mathbf{I}_m)^{-1}\mathbf{y}$$

- The prediction for a new input point $\mathbf{x}$ is given by

$$\hat{f}(\mathbf{x}) = \phi(\mathbf{x})^\top\mathbf{\Phi}^\top(\mathbf{\Phi}\mathbf{\Phi}^\top + \lambda\mathbf{I}_m)^{-1}\mathbf{y}$$

$\Rightarrow$ The matrix $\mathbf{\Phi}\mathbf{\Phi}^\top$ has size $m \times m$!

$\Rightarrow$ The vector $\phi(\mathbf{x})^\top\mathbf{\Phi}^\top$ has size $1 \times m$!

$\Rightarrow$ Non-parametric regression scales with the number of data m!

Kernel trick

- Avoid the explicit mapping to the feature space
- A *kernel* is any function $k : \mathbb{R}^n \times \mathbb{R}^n \mapsto \mathbb{R}$ which corresponds to a dot product for some feature mapping ϕ:

$$k(\mathbf{x}, \mathbf{x}') = \phi(\mathbf{x})^\top \phi(\mathbf{x}') \text{ for some } \phi.$$

- Conversely, by choosing a feature map $\phi : \mathbb{R}^n \to \mathbb{R}^d$, we implicitly choose a kernel function (d may even be infinite!)
- Recall that $\phi(\mathbf{x})^\top \phi(\mathbf{x}') = \cos \angle(\mathbf{x}, \mathbf{x}')$ where $\angle$ denotes the angle between the vectors, so a kernel function can be thought of as a notion of *similarity*.

Kernel regression

- Let $\mathbf{K} = \mathbf{\Phi}\mathbf{\Phi}^\top \in \mathbb{R}^{m \times m}$ be the so-called Gram matrix:

$$\mathbf{K} = \begin{bmatrix} k(\mathbf{x}_1, \mathbf{x}_1) & k(\mathbf{x}_1, \mathbf{x}_2) & \dots & k(\mathbf{x}_1, \mathbf{x}_m) \\ k(\mathbf{x}_2, \mathbf{x}_1) & k(\mathbf{x}_2, \mathbf{x}_2) & \dots & k(\mathbf{x}_2, \mathbf{x}_m) \\ \vdots & \vdots & & \vdots \\ k(\mathbf{x}_m, \mathbf{x}_1) & k(\mathbf{x}_m, \mathbf{x}_2) & \dots & k(\mathbf{x}_m, \mathbf{x}_m) \end{bmatrix}$$

- Solution of regularized least squares: $\mathbf{w} = \mathbf{\Phi}^\top(\mathbf{K} + \lambda \mathbf{I}_m)^{-1}\mathbf{y}$
- The predictions for the input data are given by

$$\hat{\mathbf{y}} = \mathbf{\Phi}\mathbf{w} = \mathbf{K}(\mathbf{K} + \lambda \mathbf{I})^{-1}\mathbf{y}$$

Kernel regression (cont'd)

- The prediction for a new input point $\mathbf{x}$ is given by

$$\hat{f}(\mathbf{x}) = \phi(\mathbf{x})^\top \mathbf{\Phi}^\top (\mathbf{K} + \lambda \mathbf{I}_m)^{-1}\mathbf{y} = \mathbf{k}(\mathbf{x})(\mathbf{K} + \lambda \mathbf{I}_m)^{-1}\mathbf{y}$$

where $\mathbf{k}(\mathbf{x}) \in \mathbb{R}^m$ is defined by

$$\mathbf{k}(\mathbf{x}) = \begin{bmatrix} k(\mathbf{x}, \mathbf{x}_1) \\ k(\mathbf{x}, \mathbf{x}_2) \\ \vdots \\ k(\mathbf{x}, \mathbf{x}_m) \end{bmatrix}$$

$\Rightarrow$ Never need to compute the feature map ϕ explicitly!

$\Rightarrow$ Especially useful when ϕ as dimension $d \to \infty$

Example: Quadratic kernel

- Let $k(\mathbf{x}, \mathbf{x}') = \left(\mathbf{x}^\top \mathbf{x}'\right)^2$.
- Is this a kernel?

$$k(\mathbf{x}, \mathbf{x}') = \left(\sum_{i=1}^{n} x_i x'_i\right)\left(\sum_{j=1}^{n} x_j x'_j\right) = \sum_{i,j \in \{1 \dots n\}} x_i x'_i x_j x'_j$$

$$= \sum_{i,j \in \{1 \dots n\}} (x_i x_j)\left(x'_i x'_j\right)$$

- Hence, it is a kernel, with feature mapping:

$$\phi(\mathbf{x}) = \langle x_1^2,\ x_1 x_2,\ \dots,\ x_1 x_n,\ x_2 x_1,\ x_2^2,\ \dots,\ x_n^2 \rangle$$

  Feature vector includes all squares of elements and all cross terms.
- Note that computing ϕ takes $O(n^2)$ but *computing k takes only $O(n)$*!

Establishing "kernelhood"

- Suppose someone hands you a function k. How do you know that it is a kernel?
- More precisely, given a function $k : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$, under what conditions can $k(\mathbf{x}, \mathbf{x}')$ be written as a dot product $\phi(\mathbf{x})^\top \phi(\mathbf{x}')$ for some feature mapping ϕ?
- We want a general recipe, which does not require explicitly defining ϕ every time

Kernel matrix

- Suppose we have an arbitrary set of input vectors $\mathbf{x}_1, \mathbf{x}_2, \ldots \mathbf{x}_m$
- Recall: the *kernel matrix (or Gram matrix)* $\mathbf{K}$ corresponding to kernel function k is an $m \times m$ matrix such that $\mathbf{K}_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$ (notation is overloaded on purpose).
- What properties does the kernel matrix $\mathbf{K}$ have?
- Claims:
  1. $\mathbf{K}$ is symmetric
  2. $\mathbf{K}$ is positive semidefinite
- Note that these claims are consistent with the intuition that k is a "similarity" measure (and will be true regardless of the data)

Proving the first claim

If k is a valid kernel, then the kernel matrix is symmetric

$$\mathbf{K}_{ij} = \phi(\mathbf{x}_i)^\top \phi(\mathbf{x}_j) = \phi(\mathbf{x}_j)^\top \phi(\mathbf{x}_i) = \mathbf{K}_{ji}$$

Proving the second claim

If k is a valid kernel, then the kernel matrix is positive semidefinite

Proof: Consider an arbitrary vector $\mathbf{z}$

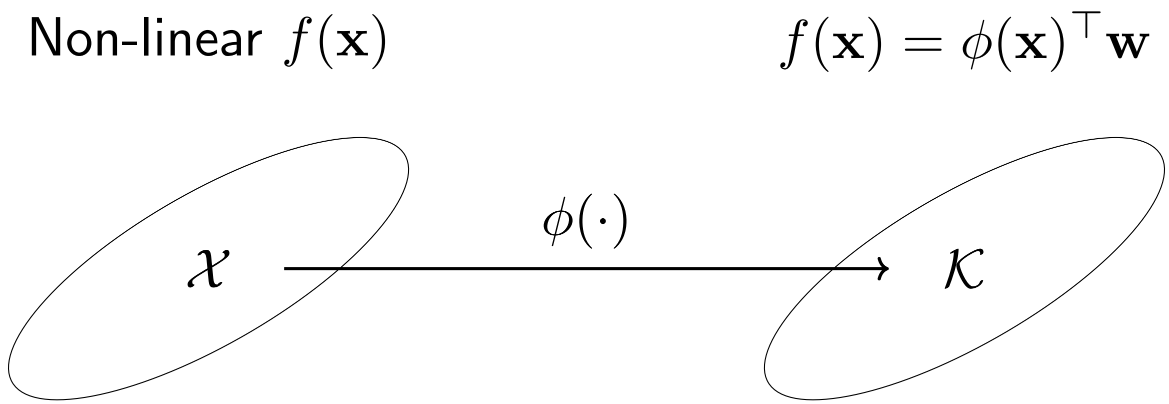
$$\begin{aligned}
\mathbf{z}^\top \mathbf{K}\mathbf{z} &= \sum_i \sum_j z_i K_{ij} z_j = \sum_i \sum_j z_i \phi(\mathbf{x}_i)^\top \phi(\mathbf{x}_j) z_j \\
&= \sum_i \sum_j z_i \left(\sum_k \phi_k(\mathbf{x}_i)\phi_k(\mathbf{x}_j) \right) z_j \\
&= \sum_k \sum_i \sum_j z_i \phi_k(\mathbf{x}_i)\phi_k(\mathbf{x}_j) z_j \\
&= \sum_k \left(\sum_i z_i \phi_k(\mathbf{x}_i) \right)^2 \geq 0
\end{aligned}$$

Mercer's theorem

- Mercer's theorem states that the reverse is also true: Given a function $k : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$, *$k$ is a kernel if and only if, for any data set, the corresponding kernel matrix $\mathbf{K}$ is symmetric and positive semidefinite*
- The reverse direction of the proof is much harder (see e.g. Vapnik's book for details)
- This result gives us a way to check if a given function is a kernel, by checking these two properties of its kernel matrix.
- Kernels can also be obtained by combining other kernels, or by learning from data

RKHS

- Let $f : \mathcal{X} \mapsto \mathcal{Y}$ denote the function generating the outputs, s.t. $y = f(\mathbf{x})$



- We say that f belongs to $\mathcal{K}$ if $\|f\|_{\mathcal{K}}^2 = \|\mathbf{w}\|^2 < \infty$
- $\mathcal{K}$ is known as the reproducing kernel Hilbert space (RKHS) associated with kernel k

More on RKHS

- The feature space is the RKHS

$$\mathcal{K} = \left\{ \sum_j \alpha_j \phi(\mathbf{x}_j) \ : \ \mathbf{x}_j \in \mathcal{X}, \ \alpha_j \in \mathbb{R} \right\}$$

  with inner product $\langle \phi(\mathbf{x}), \phi(\mathbf{x}') \rangle_{\mathcal{K}} = k(\mathbf{x}, \mathbf{x}')$

- The term reproducing comes from the reproducing property of the kernel function:

$$\forall f \in \mathcal{K}, \ \mathbf{x} \in \mathcal{X} \ : \quad f(\mathbf{x}) = \langle f(\cdot), \phi(\mathbf{x}) \rangle_{\mathcal{K}}$$

- The solution of the regularized least square in the feature space associated to a kernel function k has the form $h_\phi(\mathbf{x}) = \sum_{i=1}^m \alpha_i k(\mathbf{x}_i, \mathbf{x})$
  $\Rightarrow$ Show it as an exercise
  This is a particular case of the representer theorem...

Representer Theorem

Theorem 1. *Let $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ be a positive definite kernel and let $\mathcal{K}$ be the corresponding RKHS.*

Then for any training sample $\mathcal{S} = \{(x_i, y_i)\}_{i=1}^m \subset \mathcal{X} \times \mathbb{R}$, any loss function $\ell : (\mathcal{X} \times \mathbb{R} \times \mathbb{R})^m \to \mathbb{R}$ and any real-valued non-decreasing function g, the solution of the optimization problem

$$\arg\min_{f \in \mathcal{K}} \ell\left((\mathbf{x}_1, y_1, f(\mathbf{x}_1)), \cdots, (\mathbf{x}_m, y_m, f(\mathbf{x}_m))\right) + g(\|f\|_{\mathcal{K}})$$

admits a representation of the form

$$f^*(\cdot) = \sum_{i=1}^m \alpha_i \phi(\mathbf{x}_i).$$

[Schölkopf, Herbrich and Smola. *A generalized representer Theorem.* COLT 2001.]

Summary

- Use feature mapping ϕ to send data from $\mathcal{X}$ to $\mathcal{K}$, s.t. $f(\mathbf{x}) = \phi(\mathbf{x})^\top \mathbf{w}$
- We can solve the system in closed-form: $\mathbf{w} = (\mathbf{\Phi}^\top \mathbf{\Phi})^{-1} \mathbf{\Phi}^\top \mathbf{y}$
- Ridge regression make things invertible: $\mathbf{w} = (\mathbf{\Phi}^\top \mathbf{\Phi} + \lambda \mathbf{I})^{-1} \mathbf{\Phi}^\top \mathbf{y}$
- Parametric regression scales with feature mappings ϕ dimension d
- For large d: Non-parametric regression + kernel trick!
- No need to explicit ϕ anymore
- Different kernels to encode different prior knowledge on the function

Example kernel: Gaussian

$$k(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}'\|^2}{2\rho^2}\right)$$

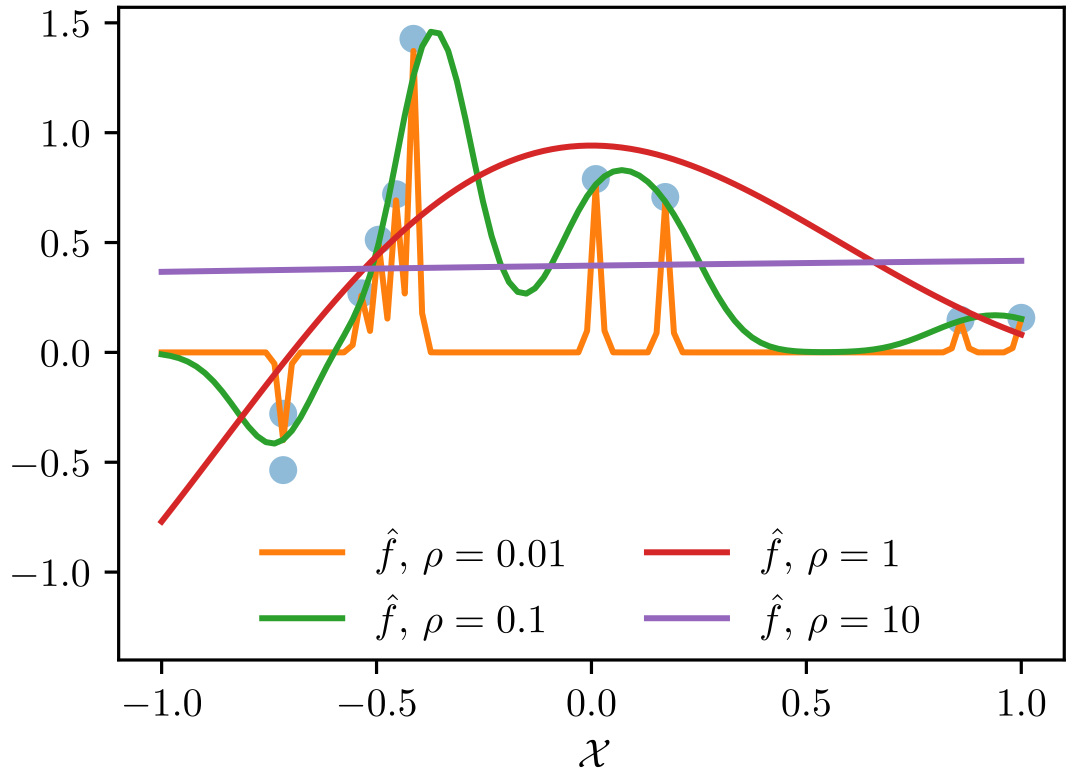
- Also known as squared exponential kernel
- The feature space of this kernel has infinite dimension d
- We can approximate $\phi(\cdot)$ with Taylor expansion, e.g. for $x \in \mathbb{R}$

$$\phi_i(x) = \exp\left(-\frac{x^2}{2\rho^2}\right)\frac{x^{i-1}}{\rho^{i-1}\sqrt{(i-1)!}}$$

- ρ is the lengthscale or bandwidth: radius of *information sharing*

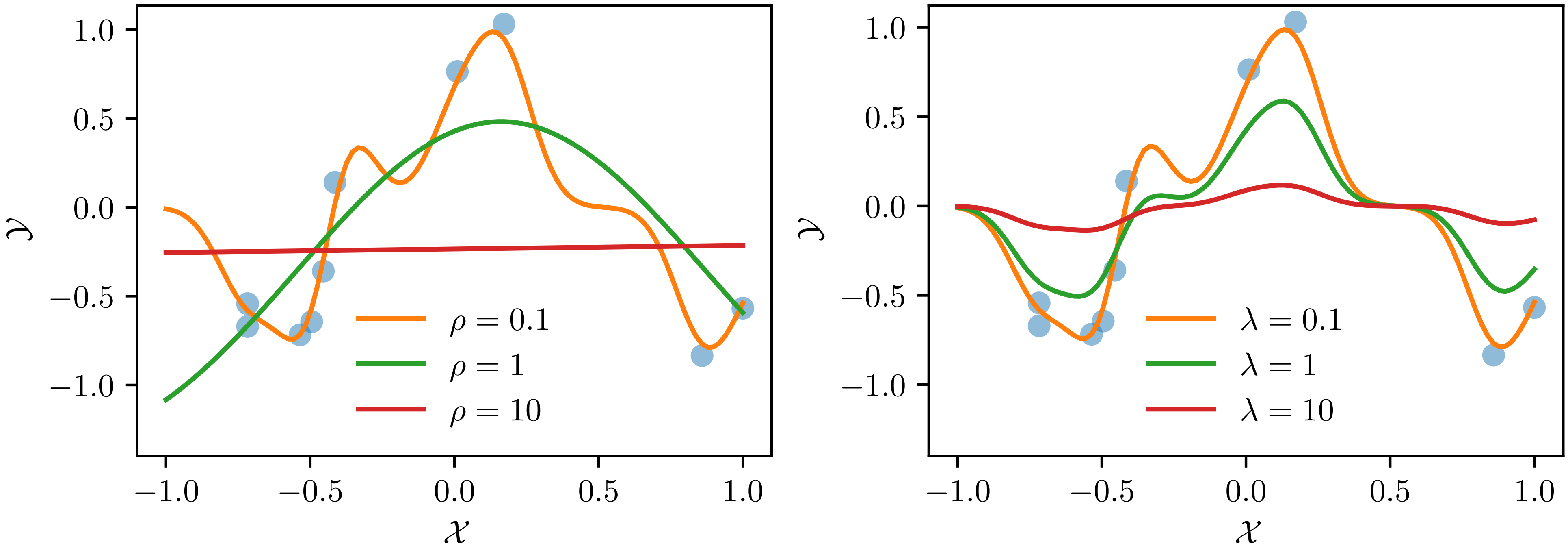
Gaussian kernel bandwidth

- Large bandwidth: all points contribute equally
- Small bandwidth: only local points contribute



Bandwidth vs Regularization

- Bandwidth controls smoothness
- Example: fixed $\lambda = 0.1$ vs fixed $\rho = 0.1$



Kernel hyperparameters

- Kernel hyperparameters can cause overfitting
- Sometimes prior knowledge is enough to pick *appropriate* values
- One could use cross-validation to find *good* values
- One can pick the *most likely* values

Bayesian view of regression

- Consider noisy observations $y = f(\mathbf{x}) + \epsilon$
- With Gaussian noise $\epsilon \sim \mathcal{N}(0, \sigma^2)$

$$P_\phi(\mathbf{y}|\mathbf{X},\mathbf{w}) = \prod_{i=1}^{m} P_\phi(y_i|\mathbf{x}_i,\mathbf{w}) = \prod_{i=1}^{m} \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(y_i - \phi(\mathbf{x}_i)^\top \mathbf{w})^2}{2\sigma^2}\right)$$
$$= \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{\|\mathbf{y} - \mathbf{\Phi}\mathbf{w}\|^2}{2\sigma^2}\right) = \mathcal{N}\left(\mathbf{\Phi}\mathbf{w}, \sigma^2 \mathbf{I}_m\right)$$

- Recall Bayes' rule: posterior $= \frac{\text{likelihood} \times \text{prior}}{\text{marginal likelihood}}$

$$P_\phi(\mathbf{w}|\mathbf{y},\mathbf{X}) = \frac{P_\phi(\mathbf{y}|\mathbf{X},\mathbf{w})P(\mathbf{w})}{P_\phi(\mathbf{y}|\mathbf{X})}$$

$\Rightarrow$ Marginal likelihood is independent of weights $\mathbf{w}$

Posterior distribution on parameters

- With Gaussian prior on parameters $\mathbf{w} \sim \mathcal{N}(0, \Sigma_{\mathbf{w}})$

$$
\begin{aligned}
P_\phi(\mathbf{w}|\mathbf{y}, \mathbf{X}) &\propto \exp\left(-\frac{\|\mathbf{y} - \mathbf{\Phi}\mathbf{w}\|^2}{2\sigma^2}\right) \exp\left(-\frac{\mathbf{w}^\top \Sigma_{\mathbf{w}}^{-1} \mathbf{w}}{2}\right) \\
&= \exp\left(-\frac{\mathbf{y}^\top\mathbf{y} - \mathbf{y}^\top\mathbf{\Phi}\mathbf{w} - \mathbf{w}^\top\mathbf{\Phi}\mathbf{y} + \mathbf{w}^\top\mathbf{\Phi}^\top\mathbf{\Phi}\mathbf{w} + \sigma^2\mathbf{w}^\top\Sigma_{\mathbf{w}}^{-1}\mathbf{w}}{2\sigma^2}\right) \\
&= \exp\left(-\frac{\mathbf{y}^\top\mathbf{y} - \mathbf{y}^\top\mathbf{\Phi}\mathbf{w} - \mathbf{w}^\top\mathbf{\Phi}\mathbf{y} + \mathbf{w}^\top(\mathbf{\Phi}^\top\mathbf{\Phi} + \sigma^2\Sigma_{\mathbf{w}}^{-1})\mathbf{w}}{2\sigma^2}\right) \\
&\propto \exp\left((\mathbf{w} - \mathbf{b})^\top \mathbf{A}^{-1} (\mathbf{w} - \mathbf{b})\right)
\end{aligned}
$$

where $\mathbf{A}^{-1} = \sigma^{-2}(\mathbf{\Phi}^\top\mathbf{\Phi} + \sigma^2\Sigma_{\mathbf{w}}^{-1})$ and $\mathbf{b} = (\mathbf{\Phi}^\top\mathbf{\Phi} + \sigma^2\Sigma_{\mathbf{w}}^{-1})^{-1}\mathbf{\Phi}^\top\mathbf{y}$

$\Rightarrow$ The posterior distribution is Gaussian!

Predictive distribution

- The pointwise posterior predictive distribution is a normal distribution

$$\tilde{f}(\mathbf{x})|\mathbf{x}_1, \ldots, \mathbf{x}_m, y_1, \ldots, y_m \sim \mathcal{N}\left(\hat{f}(\mathbf{x}), s^2(\mathbf{x})\right)$$

of expectation

$$\begin{aligned}\hat{f}(\mathbf{x}) &= \phi(\mathbf{x})^\top(\mathbf{\Phi}^\top\mathbf{\Phi} + \sigma^2\Sigma_{\mathbf{w}}^{-1})^{-1}\mathbf{\Phi}^\top\mathbf{y} \\ &= \phi(\mathbf{x})^\top\Sigma_{\mathbf{w}}\mathbf{\Phi}^\top(\mathbf{\Phi}\Sigma_{\mathbf{w}}\mathbf{\Phi}^\top + \sigma^2\mathbf{I}_m)^{-1}\mathbf{y}\end{aligned}$$

and variance

$$\begin{aligned}s^2(\mathbf{x}) &= \sigma^2\phi(\mathbf{x})^\top(\mathbf{\Phi}^\top\mathbf{\Phi} + \sigma^2\Sigma_{\mathbf{w}}^{-1})^{-1}\phi(\mathbf{x}) \\ &= \phi(\mathbf{x})^\top\Sigma_{\mathbf{w}}\phi(\mathbf{x}) - \phi(\mathbf{x})^\top\Sigma_{\mathbf{w}}\mathbf{\Phi}^\top(\mathbf{\Phi}^\top\Sigma_{\mathbf{w}}\mathbf{\Phi} + \sigma^2\mathbf{I}_m)^{-1}\mathbf{\Phi}\Sigma_{\mathbf{w}}\phi(\mathbf{x}) \\ &\rightarrow \text{using Sherman-Morrison}\end{aligned}$$

Reinterpreting regularization

- Recall kernel regression predictions:

$$\hat{f}(\mathbf{x}) = \mathbf{k}(\mathbf{x})^\top (\mathbf{K} + \lambda \mathbf{I})^{-1} \mathbf{y}$$

- Using prior $\Sigma_{\mathbf{w}} = \frac{\sigma^2}{\lambda} \mathbf{I}_d$, the predictive mean rewrites as:

$$\begin{aligned}
\hat{f}(\mathbf{x}) &= \phi(\mathbf{x})^\top \Sigma_{\mathbf{w}} \mathbf{\Phi}^\top (\mathbf{\Phi} \Sigma_{\mathbf{w}} \mathbf{\Phi}^\top + \sigma^2 \mathbf{I}_m)^{-1} \mathbf{y} \\
&= \phi(\mathbf{x})^\top \frac{\sigma^2}{\lambda} \mathbf{\Phi}^\top \left(\mathbf{\Phi} \frac{\sigma^2}{\lambda} \mathbf{\Phi}^\top + \sigma^2 \mathbf{I}_m \right)^{-1} \mathbf{y} \\
&= \mathbf{k}(\mathbf{x})^\top (\mathbf{K} + \lambda \mathbf{I})^{-1} \mathbf{y}
\end{aligned}$$

$\Rightarrow$ λ encodes some prior on weights $\mathbf{w}$

Reinterpreting regularization (cont'd)

- Still using $\Sigma_{\mathbf{w}} = \frac{\sigma^2}{\lambda}\mathbf{I}_d$, the predictive variance rewrites as:

$$
\begin{aligned}
s^2(\mathbf{x}) &= \phi(\mathbf{x})^\top \Sigma_{\mathbf{w}} \phi(\mathbf{x}) - \phi(\mathbf{x})^\top \Sigma_{\mathbf{w}} \mathbf{\Phi}^\top (\mathbf{\Phi}^\top \Sigma_{\mathbf{w}} \mathbf{\Phi} + \sigma^2 \mathbf{I}_m)^{-1} \mathbf{\Phi} \Sigma_{\mathbf{w}} \phi(\mathbf{x}) \\
&= \phi(\mathbf{x})^\top \frac{\sigma^2}{\lambda} \phi(\mathbf{x}) - \phi(\mathbf{x})^\top \frac{\sigma^2}{\lambda} \mathbf{\Phi}^\top \left(\mathbf{\Phi}^\top \frac{\sigma^2}{\lambda} \mathbf{\Phi} + \sigma^2 \mathbf{I}_m \right)^{-1} \mathbf{\Phi} \frac{\sigma^2}{\lambda} \phi(\mathbf{x}) \\
&= \frac{\sigma^2}{\lambda} k_\lambda(\mathbf{x}, \mathbf{x}) \qquad \text{with} \\
k_\lambda(\mathbf{x}, \mathbf{x}') &= k(\mathbf{x}, \mathbf{x}') - \mathbf{k}(\mathbf{x})^\top (\mathbf{K} + \lambda \mathbf{I}_m)^{-1} \mathbf{k}(\mathbf{x}')
\end{aligned}
$$

Joint distribution

- Suppose you *query* your model at locations $\mathbf{X}_*$
- Extend the prior to include query points:

$$\begin{bmatrix}\mathbf{f}\\ \mathbf{f}_*\end{bmatrix} |\mathbf{X},\mathbf{X}_* \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix}\mathbf{K}_{\mathbf{X},\mathbf{X}}+ & \mathbf{K}_{\mathbf{X},\mathbf{X}_*}\\ \mathbf{K}_{\mathbf{X}_*,\mathbf{X}} & \mathbf{K}_{\mathbf{X}_*,\mathbf{X}_*}\end{bmatrix}\right)$$

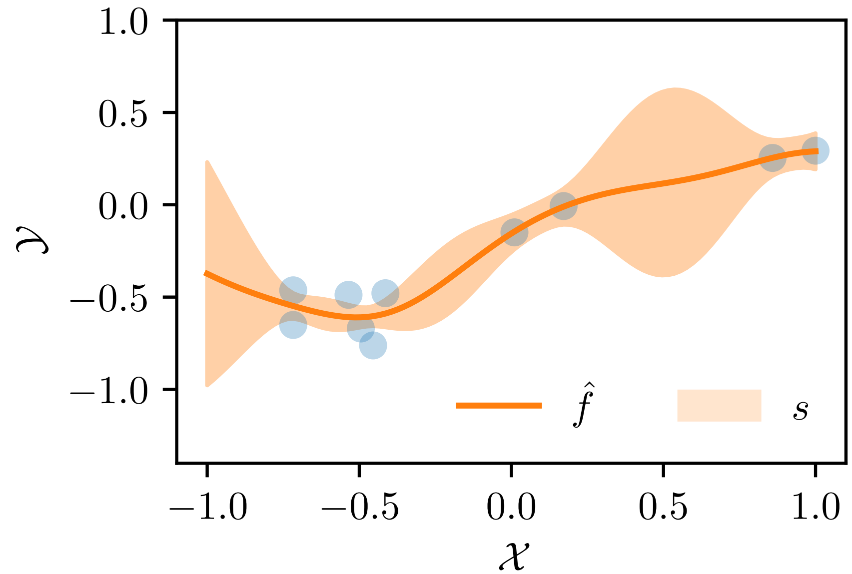
$$\mathbf{y}|\mathbf{f} \sim \mathcal{N}(\mathbf{f}, \sigma^2\mathbf{I})$$

- Using joint normality of $\mathbf{f}_*$ and $\mathbf{y}$:

$$\begin{bmatrix}\mathbf{y}\\ \mathbf{f}_*\end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix}\mathbf{K}_{\mathbf{X},\mathbf{X}}+\lambda\mathbf{I} & \mathbf{K}_{\mathbf{X},\mathbf{X}_*}\\ \mathbf{K}_{\mathbf{X}_*,\mathbf{X}} & \mathbf{K}_{\mathbf{X}_*,\mathbf{X}_*}\end{bmatrix}\right)$$

Pointwise posterior distribution

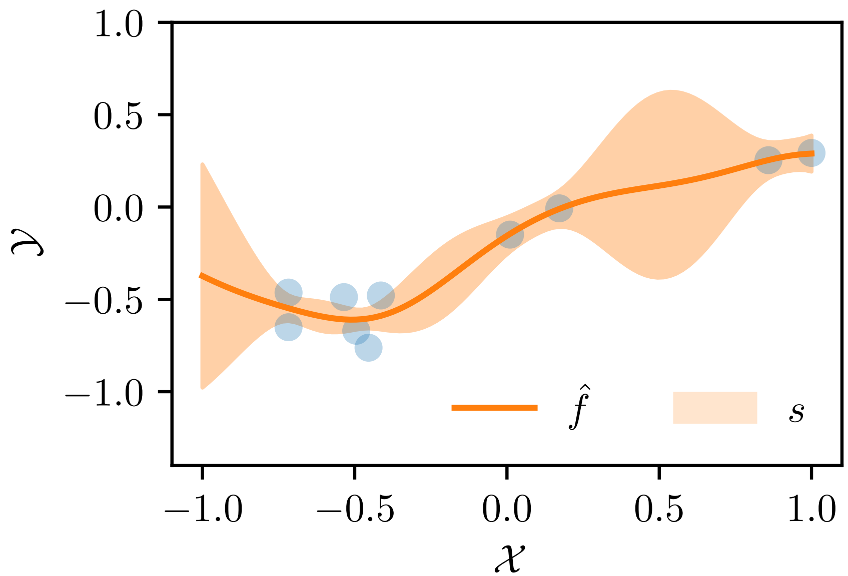
- At each point $\mathbf{x} \in \mathcal{X}$, we have a distribution $\mathcal{N}\left(\hat{f}(\mathbf{x}), s^2(\mathbf{x})\right)$
- We can sample from these $\tilde{f}(\mathbf{x}) \sim \mathcal{N}\left(\hat{f}(\mathbf{x}), s^2(\mathbf{x})\right)$



Gaussian Process (GP)

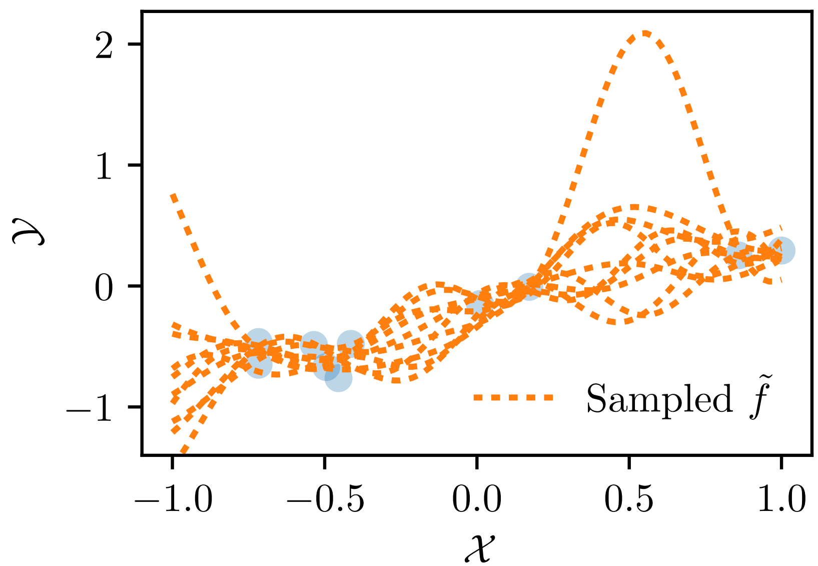
- By considering the covariance between *every points in the space*, we get a distribution over functions!
- Posterior distribution on f:

$$P[f|\mathbf{X},\mathbf{y}] \sim \mathcal{N}\left(\left[\hat{f}(\mathbf{x})\right]_{\mathbf{x}\in\mathcal{X}}, \frac{\sigma^2}{\lambda}\left[k_\lambda(\mathbf{x},\mathbf{x}')\right]_{\mathbf{x},\mathbf{x}'\in\mathcal{X}}\right)$$



Sampling from a Gaussian Process

- Generalization of normal probability distribution to the function space
  - From a normal distribution we sample variables
  - From a GP we sample *functions*!



Sampling from a Gaussian Process: How to

- Observe that

$$\mathcal{N}\left(\left[\hat{f}(\mathbf{x})\right]_{\mathbf{x}\in\mathcal{X}}, \frac{\sigma^2}{\lambda}\left[k_\lambda(\mathbf{x},\mathbf{x}')\right]_{\mathbf{x},\mathbf{x}'\in\mathcal{X}}\right)$$

  defines a $|\mathcal{X}|$-dimensional multivariate Gaussian distribution
- If $\mathcal{X}$ is continuous (e.g. $\mathcal{X} = [-1, 1]$), $|\mathcal{X}| = \infty$
- We can consider a discrete, finite, set $\mathbb{X} \subset \mathcal{X}$ and sample from

$$\mathcal{N}\left(\left[\hat{f}(\mathbf{x})\right]_{\mathbf{x}\in\mathbb{X}}, \frac{\sigma^2}{\lambda}\left[k_\lambda(\mathbf{x},\mathbf{x}')\right]_{\mathbf{x},\mathbf{x}'\in\mathbb{X}}\right)$$

- This will result in a function $\tilde{f}$ evaluated at every $\mathbf{x} \in \mathbb{X}$

Learning the hyperparameters

- If we assume that $\Sigma_{\mathbf{w}} = \mathbf{I}_d$, then we have $\lambda = \sigma^2$
- Recall: multivariate normal density

$$P(\mathbf{y}|\boldsymbol{\theta}) = \frac{\exp\left(-\frac{1}{2}\mathbf{y}^\top(\mathbf{K}_{\boldsymbol{\theta}} + \sigma^2\mathbf{I}_m)^{-1}\mathbf{y}\right)}{\sqrt{(2\pi)^D|\mathbf{K}_{\boldsymbol{\theta}} + \sigma^2\mathbf{I}_m|}}$$

- Maximize the marginal likelihood $\mathcal{L} = \log P(\mathbf{y}|\boldsymbol{\theta})$ w.r.t. kernel hyperparameters (e.g. ρ) and noise σ:

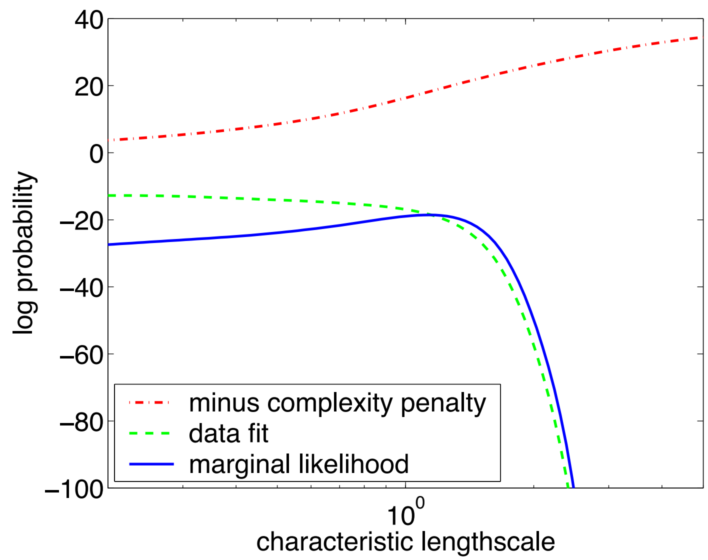
$$\mathcal{L} = -\frac{1}{2}\mathbf{y}^\top(\mathbf{K}_{\boldsymbol{\theta}} + \sigma^2\mathbf{I}_m)^{-1}\mathbf{y} - \frac{D}{2}\log(2\pi) - \frac{1}{2}\log|\mathbf{K}_{\boldsymbol{\theta}} + \sigma^2\mathbf{I}_m|$$

Anatomy of marginal likelihood

- Marginal likelihood:

$$\mathcal{L} = \log P(\mathbf{y}|\boldsymbol{\theta}) = -\frac{1}{2}\mathbf{y}^\top(\mathbf{K}_{\boldsymbol{\theta}} + \sigma^2\mathbf{I}_m)^{-1}\mathbf{y} - \frac{1}{2}\log|\mathbf{K}_{\boldsymbol{\theta}} + \sigma^2\mathbf{I}_m|$$

- 1st term: quality of predictions; 2nd term: model complexity
- Trade-off (from Rasmussen & Williams, 2006):



Gradient-based optimization

- Compute gradients:

$$
\begin{aligned}
\frac{\partial \mathcal{L}}{\partial \theta_i} =& \frac{1}{2}\mathbf{y}^\top(\mathbf{K}_{\boldsymbol{\theta}} + \sigma^2\mathbf{I}_m)^{-1}\frac{\partial(\mathbf{K}_{\boldsymbol{\theta}} + \sigma^2\mathbf{I}_m)}{\partial \theta_i}(\mathbf{K}_{\boldsymbol{\theta}} + \sigma^2\mathbf{I}_m)^{-1}\mathbf{y} \\
&- \frac{1}{2}\,\mathrm{Tr}\left((\mathbf{K}_{\boldsymbol{\theta}} + \sigma^2\mathbf{I}_m)^{-1}\frac{\partial(\mathbf{K}_{\boldsymbol{\theta}} + \sigma^2\mathbf{I}_m)}{\partial \theta_i}\right)
\end{aligned}
$$

- Minimize the negative
- Non-convex optimization task

Summary

- Normal priors on the weights distribution $\rightarrow$ Gaussian Process
- Regularization $\rightarrow$ prior on the weights covariance
- GP provides a posterior distribution on functions
  - Expectation: kernel regression model
  - Covariance $\rightarrow$ confidence intervals
- Sample discretized functions from a GP