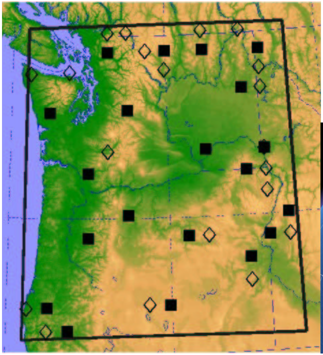
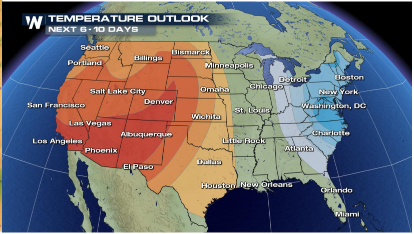
Data Analysis: Statistical Modeling and Computation in Applications

Spatial and Environmental Data:
Gaussian Processes

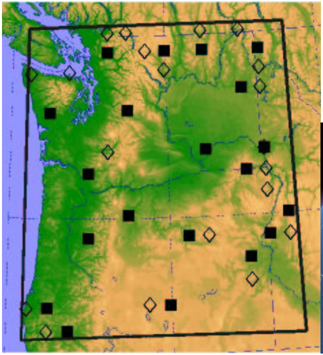
Sensing and correlations in space

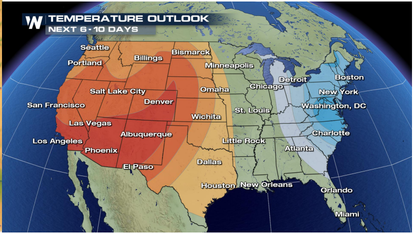




- Model correlations in space?

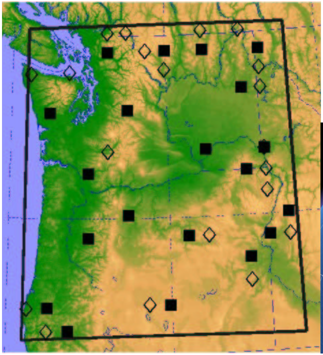
Sensing and correlations in space

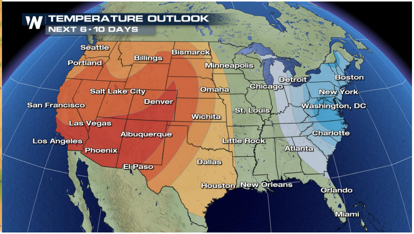




- Model correlations in space?
- *Intuition: correlation is a function of distance*

Sensing and correlations in space





- Model correlations in space?
- *Intuition: correlation is a function of distance*
- Idea: model as a collection of dependent Gaussian random variables. Covariance is a function of distance via kernel function.

Overview

- Prediction and the kernel function
- Gaussian Processes and the kernel function
- Effect of the kernel function
- Effect of measurement noise and nonstationarity

High level idea

- We determine a probabilistic model: multivariate Gaussian with pre-specified covariance – decaying with distance.

High level idea

- We determine a probabilistic model: multivariate Gaussian with pre-specified covariance – decaying with distance.
- We observe values $y_1, \ldots, y_N$ for some "locations" $x_1, \ldots, x_N$.

High level idea

- We determine a probabilistic model: multivariate Gaussian with pre-specified covariance – decaying with distance.
- We observe values $y_1, \ldots, y_N$ for some "locations" $x_1, \ldots, x_N$.
- We obtain a conditional distribution for the unobserved variables Y_* (posterior): $p(Y_* | y_1, \ldots, y_N)$ Gaussian, shifted towards observations.

High level idea

- We determine a probabilistic model: multivariate Gaussian with pre-specified covariance – decaying with distance.
- We observe values $y_1, \ldots, y_N$ for some "locations" $x_1, \ldots, x_N$.
- We obtain a conditional distribution for the unobserved variables Y_* (posterior): $p(Y_* | y_1, \ldots, y_N)$ Gaussian, shifted towards observations.

$\mathrm{cov}(Y_*, Y_i)$

$$\mu_{*|1:N} = \mu_* + \sigma_{N*}^\top \Sigma_N^{-1} (y_{1:N} - \mu_{1:N})$$

$$\sigma^2_{*|1:N} = \sigma^2_* - \sigma_{N*}^\top \Sigma_N^{-1} \sigma_{N*}.$$

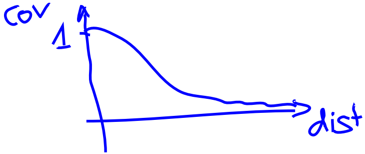
Correlations in space

- model measurements Y_i at locations x_i as Gaussian, $1 \leq i \leq N$
- *covariance is a function of locations*: kernel function,

Correlations in space

- model measurements Y_i at locations x_i as Gaussian, $1 \leq i \leq N$
- *covariance is a function of locations*: kernel function, e.g. RBF (squared exponential) kernel:

$$\mathrm{cov}(Y_i, Y_j) = k(x_i, x_j) = \exp\left(-\frac{\|x_i - x_j\|^2}{2\ell^2}\right)$$



Correlations in space

- model measurements Y_i at locations x_i as Gaussian, $1 \leq i \leq N$
- *covariance is a function of locations*: kernel function, e.g. RBF (squared exponential) kernel:

$$\mathrm{cov}(Y_i, Y_j) = k(x_i, x_j) = \exp\left(- \frac{\|x_i - x_j\|^2}{2\ell^2} \right)$$

- covariance: $\Sigma_N =$

$$\begin{bmatrix} \mathrm{cov}(Y_1, Y_1) & \dots & \mathrm{cov}(Y_1, Y_N) \\ \mathrm{cov}(Y_2, Y_1) & \dots & \vdots \\ \vdots & \ddots & \vdots \\ \mathrm{cov}(Y_N, Y_1) & \dots & \mathrm{cov}(Y_N, Y_N) \end{bmatrix}$$

Correlations in space

- model measurements Y_i at locations x_i as Gaussian, $1 \leq i \leq N$
- *covariance is a function of locations*: kernel function, e.g. RBF (squared exponential) kernel:

$$\text{cov}(Y_i, Y_j) = k(x_i, x_j) = \exp\Big(-\frac{\|x_i - x_j\|^2}{2\ell^2}\Big)$$

- covariance: $\Sigma_N =$

$$\begin{bmatrix} \text{cov}(Y_1, Y_1) & \dots & \text{cov}(Y_1, Y_N) \\ \text{cov}(Y_2, Y_1) & \dots & \vdots \\ \vdots & \ddots & \vdots \\ \text{cov}(Y_N, Y_1) & \dots & \text{cov}(Y_N, Y_N) \end{bmatrix} = \begin{bmatrix} k(x_1, x_1) & k(x_1, x_2) & \dots & k(x_1, x_N) \\ k(x_2, x_1) & k(x_2, x_2) & \vdots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ k(x_N, x_1) & & \dots & k(x_N, x_N) \end{bmatrix}$$

kernel matrix

Correlations in space

- model measurements Y_i at locations x_i as Gaussian, $1 \leq i \leq N$
- *covariance is a function of locations*: kernel function, e.g. RBF (squared exponential) kernel:

$$\mathrm{cov}(Y_i, Y_j) = k(x_i, x_j) = \exp\Big(-\frac{\|x_i - x_j\|^2}{2\ell^2}\Big)$$

- covariance: $\Sigma_N =$

$$\begin{bmatrix} \mathrm{cov}(Y_1, Y_1) & \dots & \mathrm{cov}(Y_1, Y_N) \\ \mathrm{cov}(Y_2, Y_1) & \dots & \vdots \\ \vdots & \ddots & \vdots \\ \mathrm{cov}(Y_N, Y_1) & \dots & \mathrm{cov}(Y_N, Y_N) \end{bmatrix} = \begin{bmatrix} k(x_1, x_1) & k(x_1, x_2) & \dots & k(x_1, x_N) \\ k(x_2, x_1) & k(x_2, x_2) & \vdots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ k(x_N, x_1) & \dots & & k(x_N, x_N) \end{bmatrix}$$

- *Now we can get a prediction $\hat{Y}_*$ for any location x_*!*

Prediction with Gaussian Processes (Kriging)

- build kernel matrix K_N of N observations

Prediction with Gaussian Processes (Kriging)

- build kernel matrix K_N of N observations
- for any new point x_*: compute $k_*^\top = [k(x_*, x_1), \dots \underbrace{k(x_*, x_N)}_{\mathrm{cov}(y_*, y_N)}]$ and predict

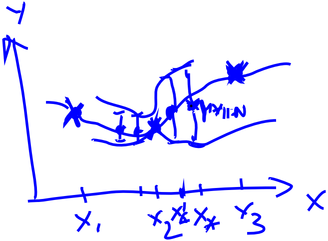
$$\mu_{*|1:N} = \mu_* + k_*^\top K_N^{-1}(y_{1:N} - \mu_{1:N})$$

Prediction with Gaussian Processes (Kriging)

- build kernel matrix K_N of N observations
- for any new point x_*: compute $k_*^\top = [k(x_*, x_1), \dots k(x_*, x_N)]$ and predict

$$\mu_{*|1:N} = \mu_* + k_*^\top K_N^{-1}(y_{1:N} - \mu_{1:N}) \quad \leftarrow \quad \hat{f}(x_*)$$

$$\sigma^2_{*|1:N} = \sigma^2_* - k_*{}^\top K_N^{-1} k_*$$

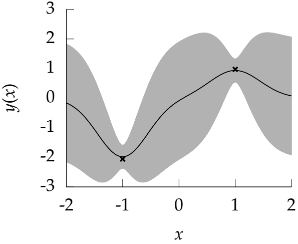


Prediction with Gaussian Processes (Kriging)

- build kernel matrix K_N of N observations
- for any new point x_*: compute $k_*^\top = [k(x_*, x_1), \dots k(x_*, x_N)]$ and predict

$$\mu_{*|1:N} = \mu_* + k_*^\top K_N^{-1}(y_{1:N} - \mu_{1:N})$$

$$\sigma^2_{*|1:N} = \sigma^2_* - {k_*}^\top K_N^{-1} k_*$$

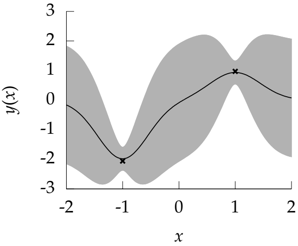


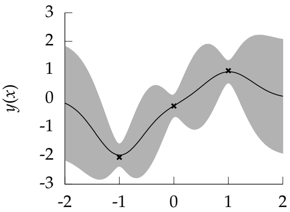
Prediction with Gaussian Processes (Kriging)

- build kernel matrix K_N of N observations
- for any new point x_*: compute $k_*^\top = [k(x_*, x_1), \ldots k(x_*, x_N)]$ and predict

$$\mu_{*|1:N} = \mu_* + k_*^\top K_N^{-1}(y_{1:N} - \mu_{1:N})$$

$$\sigma^2_{*|1:N} = \sigma^2_* - {k_*}^\top K_N^{-1} k_*$$



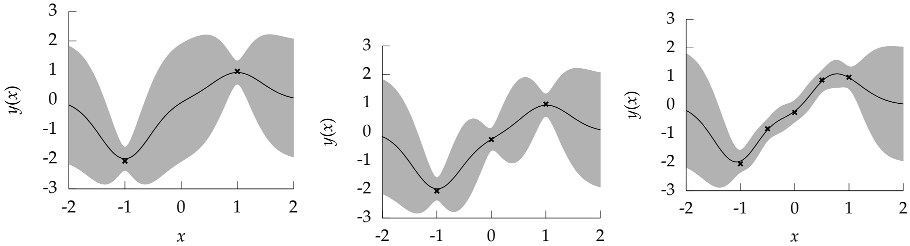


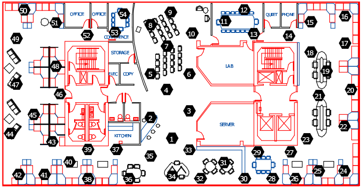
Prediction with Gaussian Processes (Kriging)

- build kernel matrix K_N of N observations
- for any new point x_*: compute $k_*^\top = [k(x_*, x_1), \dots k(x_*, x_N)]$ and predict

$$\mu_{*|1:N} = \mu_* + k_*^\top K_N^{-1}(y_{1:N} - \mu_{1:N})$$

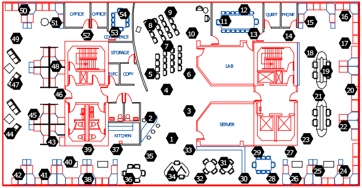
$$\sigma^2_{*|1:N} = \sigma^2_* - k_*{}^\top K_N^{-1} k_*$$

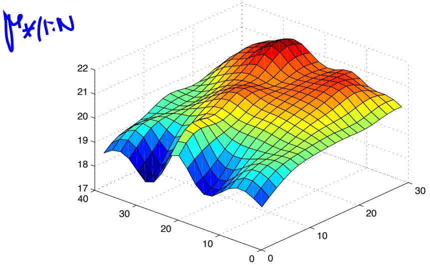




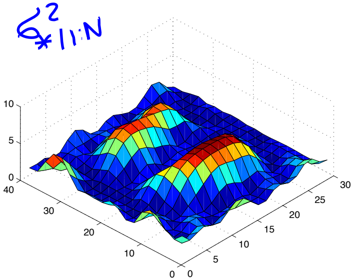
Source: Krause, Singh, Guestrin. Near-Optimal Sensor Placements in Gaussian Processes: Theory, Efficient Algorithms and Empirical Studies. JMLR, 2008.

Example





(a) *Temperature prediction using GP*



(b) *Variance of temperature prediction*

Source: Krause, Singh, Guestrin. Near-Optimal Sensor Placements in Gaussian Processes: Theory, Efficient Algorithms and Empirical Studies. JMLR, 2008.

Alternative viewpoints of prediction

$$\mu_{*|1:N} = \hat{y}_* = \hat{f}(x_*) = \underbrace{\mu_*}_{\text{assume} = 0} + \underbrace{k_*^\top}_{1:N} \underbrace{K_N^{-1}}_{N \times N} \underbrace{y_{1:N}}_{N:1}$$

$$= \underbrace{\square \; \square}_{w^*} \; \square$$

Alternative viewpoints of prediction

$$\hat{y}_* = \hat{f}(x_*) = \underbrace{\mu_*}_{\text{assume} = 0} + k_*^\top K_N^{-1} y_{1:N}$$

$$= \sum_{i=1}^{N} w_i^* y_i$$

Alternative viewpoints of prediction

$$\hat{y}_* = \hat{f}(x_*) = \underbrace{\mu_*}_{\text{assume} = 0} + k_*^\top K_N^{-1} y_{1:N}$$

$$= \sum_{i=1}^{N} w_i^* y_i$$

linear combination of N observed values

Alternative viewpoints of prediction

$$\hat{y}_* = \hat{f}(x_*) = \underbrace{\mu_*}_{\text{assume } = 0} + k_*^\top K_N^{-1} y_{1:N}$$

$$= \sum_{i=1}^{N} w_i^* y_i$$

linear combination of N observed values

$$= k_{\cancel{N}\,*}^\top \underbrace{K_N^{-1} y_{1:N}}_{\alpha}$$

$$k_* = [\mathrm{cov}(y_*, y_1), \ldots]$$

Alternative viewpoints of prediction

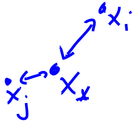
$$\hat{y}_* = \hat{f}(x_*) = \underbrace{\mu_*}_{\text{assume } = 0} + k_*^\top K_N^{-1} y_{1:N}$$

$$= \sum_{i=1}^{N} w_i^* y_i$$

linear combination of N observed values

$$= k_{\cancel{N}\,*}^\top K_N^{-1} y_{1:N}$$

$$= \sum_{i=1}^{N} k(x_*, x_i) \alpha_i$$



Alternative viewpoints of prediction

$$\hat{y}_* = \hat{f}(x_*) = \underbrace{\mu_*}_{\text{assume } = 0} + k_*^\top K_N^{-1} y_{1:N}$$

$$= \sum_{i=1}^{N} w_i^* y_i$$

linear combination of N observed values

$$= k_N^\top K_N^{-1} y_{1:N}$$

$$= \sum_{i=1}^{N} k(x_*, x_i)\alpha_i$$

linear combination of N nonlinear features

same α_is for all predictions

Overview

- Prediction and the kernel function
- Gaussian Processes and the kernel function
- Effect of the kernel function
- Effect of measurement noise and nonstationarity

The role of the covariance kernel $\mathrm{cov}(Y_i, Y_j) = k(x_i, x_j)$

- *Why use a kernel function?*

- *What is the effect of $k(x_i, x_j)$?*

- *Can I use any function as $k(x_i, x_j)$?*

The role of the covariance kernel $\text{cov}(Y_i, Y_j) = k(x_i, x_j)$

- *Why use a kernel function?*
  - Prediction for *any* x_*: a function
- *What is the effect of $k(x_i, x_j)$?*
- *Can I use any function as $k(x_i, x_j)$?*

The role of the covariance kernel $\text{cov}(Y_i, Y_j) = k(x_i, x_j)$

- *Why use a kernel function?*
  - Prediction for *any* x_*: a function
  - Compactly implements our prior assumptions on correlation structure
- *What is the effect of $k(x_i, x_j)$?*
- *Can I use any function as $k(x_i, x_j)$?*

The role of the covariance kernel $\mathrm{cov}(Y_i, Y_j) = k(x_i, x_j)$

- *Why use a kernel function?*
  - Prediction for *any* x_*: a function
  - Compactly implements our prior assumptions on correlation structure
- *What is the effect of $k(x_i, x_j)$?*
  - determines "shape" of the predicted function (examples soon)
- *Can I use any function as $k(x_i, x_j)$?*

The role of the covariance kernel $\mathrm{cov}(Y_i, Y_j) = k(x_i, x_j)$

- *Why use a kernel function?*
  - Prediction for *any* x_*: a function
  - Compactly implements our prior assumptions on correlation structure
- *What is the effect of $k(x_i, x_j)$?*
  - determines "shape" of the predicted function (examples soon)
- *Can I use any function as $k(x_i, x_j)$?*
  - Must yield a valid covariance matrix: symmetric & positive semidefinite/ inner product matrix (examples soon).

$\mathrm{cov}(Y_i, Y_j) = \mathrm{cov}(Y_j, Y_i)$ $\quad k(x_i, x_j) = k(x_j, x_i)$

$\langle x_i, x_j \rangle$ $\quad \cancel{2x_i + 3x_j}$

Covariance function and Gaussian Processes

- express covariance by *covariance function**

$$\mathrm{cov}(y_i, y_j) = k(x_i, x_j)$$

  e.g., a function of $x_i - x_j$

- this generalizes to *all* points in our space: function $f(x)$.

Covariance function and Gaussian Processes

- express covariance by *covariance function**

$$\mathrm{cov}(y_i, y_j) = k(x_i, x_j)$$

  e.g., a function of $x_i - x_j$

- this generalizes to *all* points in our space: function $f(x)$.

> A *Gaussian Process* (GP) is a collection of random variables, any finite number of which are Gaussian.

Covariance function and Gaussian Processes

- express covariance by *covariance function**

$$\mathrm{cov}(y_i, y_j) = k(x_i, x_j)$$

  e.g., a function of $x_i - x_j$

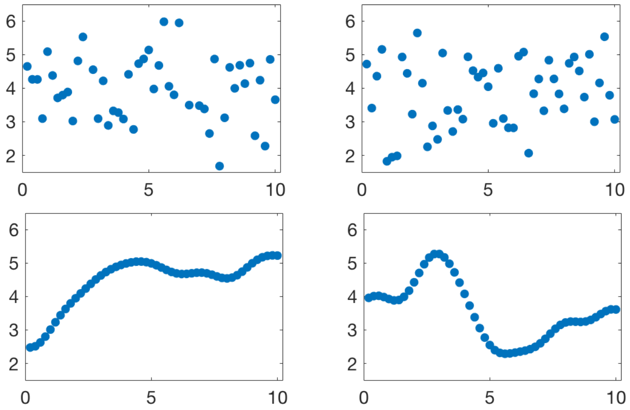
- this generalizes to *all* points in our space: function $f(x)$.

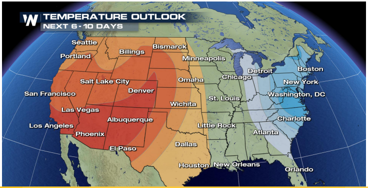
A *Gaussian Process* (GP) is a collection of random variables, any finite number of which are Gaussian.

- GP is fully specified by mean and covariance functions

$$m(x) = \mu_x = \mathbb{E}[f(x)] \qquad k(x, x') = \mathrm{cov}(f(x), f(x')).$$

Recall: 50 Gaussian random variables



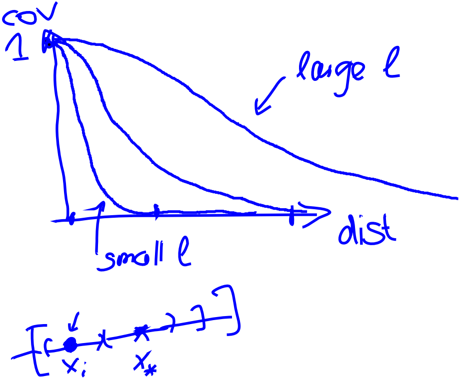


Overview

- Prediction and the kernel function
- Gaussian Processes and the kernel function
- Effect of the kernel function
- Effect of measurement noise and nonstationarity

RBF kernel: Effect of kernel bandwidth on the prediction

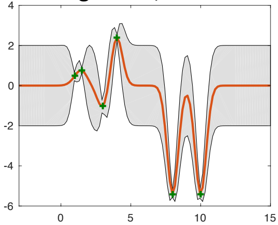
$$k(x_i, x_j) = \exp\left(-\frac{||x_i - x_j||^2}{2\ell^2}\right)$$



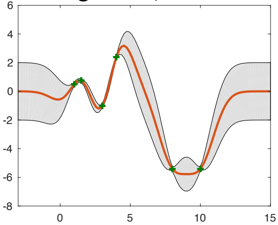
RBF kernel: Effect of kernel bandwidth on the prediction

$$k(x_i, x_j) = \exp\left(-\frac{||x_i - x_j||^2}{2\ell^2}\right)$$

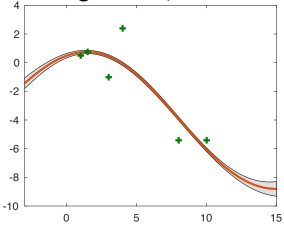
length=0.3, tau=0.1



length=1.0, tau=0.1



length=10.0, tau=0.1

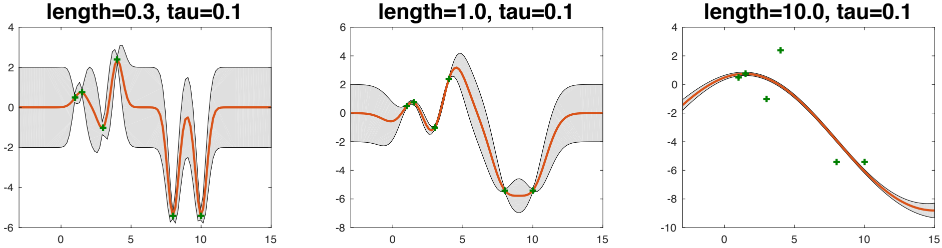


varies quickly
close interpolation

varies slowly
not full interpol.

RBF kernel: Effect of kernel bandwidth on the prediction

$$k(x_i, x_j) = \exp\left(-\frac{||x_i - x_j||^2}{2\ell^2}\right)$$

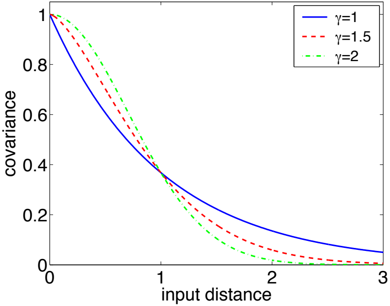


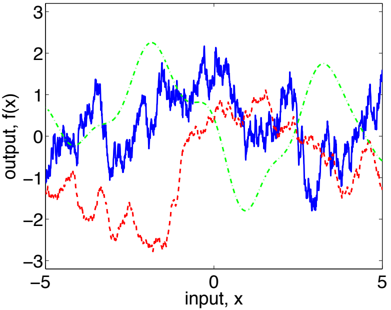
kernel function determines shape of interpolation:
larger bandwidth ⇒ smoother function

Sharpness of the covariance function

Example: $k(x, x') = \exp(-\left(\frac{\|x-x'\|}{\ell}\right)^{\gamma})$
(Gamma-exponential kernel)
What happens as we vary γ?

Example: $k(x, x') = \exp(-\left(\frac{\|x-x'\|}{\ell}\right)^{\gamma})$
(Gamma-exponential kernel)
What happens as we vary γ?





Influence of the covariance function: polynomial kernels

linear kernel
$k(x, x') = \langle x, x' \rangle$

quadratic kernel

Influence of the covariance function: polynomial kernels

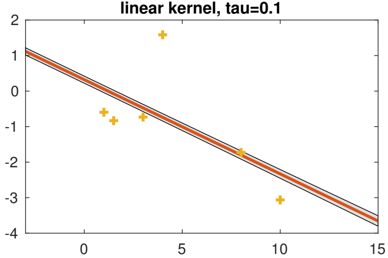
linear kernel
$k(x, x') = \langle x, x' \rangle$

quadratic kernel
$k(x, x') = (\langle x, x' \rangle + 1)^2$

Influence of the covariance function: polynomial kernels

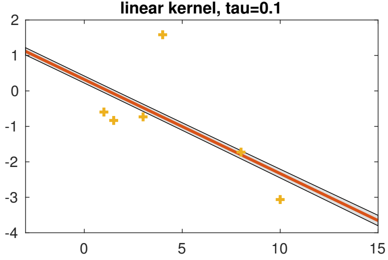
linear kernel
$k(x, x') = \langle x, x' \rangle$

quadratic kernel
$k(x, x') = (\langle x, x' \rangle + 1)^2$

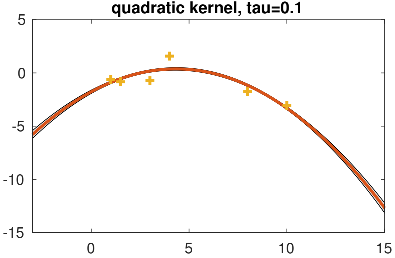


Influence of the covariance function: polynomial kernels

linear kernel
$$k(x, x') = \langle x, x' \rangle$$



quadratic kernel
$$k(x, x') = (\langle x, x' \rangle + 1)^2$$



Periodic kernel

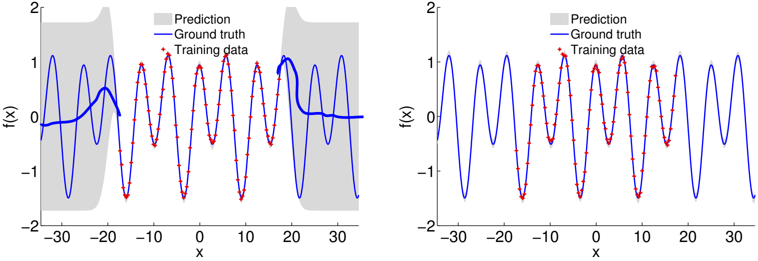
Kernel by David MacKay:

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

Periodic kernel

Kernel by David MacKay:

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



(a) Gaussian kernel. (b) Periodic kernel.

RBF

Plot uses a come advanced variation of this kernel.
Image source: Ghassemi & Deisenroth, Analytic Long-Term Forecasting with Periodic Gaussian Processes. AISTATS 2014.

Building more covariance functions

- A *sum* of kernel functions is a kernel function:

Images: Kernel cookbook, *https://www.cs.toronto.edu/~duvenaud/cookbook/*

Building more covariance functions

- A *sum* of kernel functions is a kernel function:
$$k(x, x') = k_1(x, x') + k_2(x, x').$$

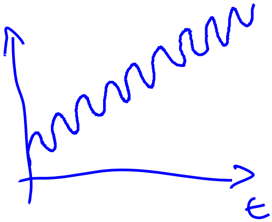
Images: Kernel cookbook, *https://www.cs.toronto.edu/~duvenaud/cookbook/*

Building more covariance functions

- A *sum* of kernel functions is a kernel function:

$$k(x, x') = k_1(x, x') + k_2(x, x').$$

  - e.g. $k_{\text{linear}} + k_{\text{periodic}}$



Images: Kernel cookbook, *https://www.cs.toronto.edu/~duvenaud/cookbook/*

Building more covariance functions

- A *sum* of kernel functions is a kernel function:
$$k(x,x') = k_1(x,x') + k_2(x,x').$$
    - e.g. $k_{\text{linear}} + k_{\text{periodic}}$

- A *product* of kernel functions is a kernel function:
$$k(x,x') = k_1(x,x') \cdot k_2(x,x')$$

Images: Kernel cookbook, `https://www.cs.toronto.edu/~duvenaud/cookbook/`

Building more covariance functions

- A *sum* of kernel functions is a kernel function:

$$k(x, x') = k_1(x, x') + k_2(x, x').$$

  - e.g. $k_{\text{linear}} + k_{\text{periodic}}$

- A *product* of kernel functions is a kernel function:

$$k(x, x') = k_1(x, x') \cdot k_2(x, x')$$

  - $k_{\text{linear}} \cdot k_{\text{periodic}}$

Images: Kernel cookbook, `https://www.cs.toronto.edu/~duvenaud/cookbook/`

Building more covariance functions

- A *sum* of kernel functions is a kernel function:

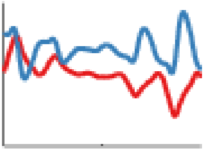
$$k(x, x') = k_1(x, x') + k_2(x, x').$$

  - e.g. $k_{\text{linear}} + k_{\text{periodic}}$

- A *product* of kernel functions is a kernel function:

$$k(x, x') = k_1(x, x') \cdot k_2(x, x')$$

  - $k_{\text{linear}} \cdot k_{\text{periodic}}$



Images: Kernel cookbook, `https://www.cs.toronto.edu/~duvenaud/cookbook/`

Building more covariance functions

- A *sum* of kernel functions is a kernel function:

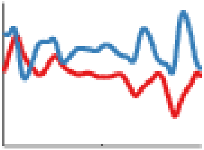
$$k(x, x') = k_1(x, x') + k_2(x, x').$$

  - e.g. $k_{\text{linear}} + k_{\text{periodic}}$

- A *product* of kernel functions is a kernel function:

$$k(x, x') = k_1(x, x') \cdot k_2(x, x')$$

  - $k_{\text{linear}} \cdot k_{\text{periodic}}$
  - $k_{\text{RBF}} \cdot k_{\text{periodic}}$



Images: Kernel cookbook, `https://www.cs.toronto.edu/~duvenaud/cookbook/`

Building more covariance functions

- A *sum* of kernel functions is a kernel function:

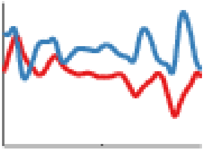
$$k(x, x') = k_1(x, x') + k_2(x, x').$$

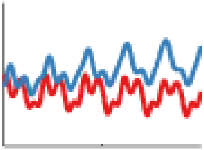
  - e.g. $k_{\text{linear}} + k_{\text{periodic}}$

- A *product* of kernel functions is a kernel function:

$$k(x, x') = k_1(x, x') \cdot k_2(x, x')$$

  - $k_{\text{linear}} \cdot k_{\text{periodic}}$
  - $k_{\text{RBF}} \cdot k_{\text{periodic}}$





Images: Kernel cookbook, `https://www.cs.toronto.edu/~duvenaud/cookbook/`

Building more covariance functions

- A *sum* of kernel functions is a kernel function:

$$k(x, x') = k_1(x, x') + k_2(x, x').$$

  - e.g. $k_{\text{linear}} + k_{\text{periodic}}$

- A *product* of kernel functions is a kernel function:

$$k(x, x') = k_1(x, x') \cdot k_2(x, x')$$

  - $k_{\text{linear}} \cdot k_{\text{periodic}}$
  - $k_{\text{RBF}} \cdot k_{\text{periodic}}$
  - splitting coordinates:
    $k\big((x_1, x_2, t), (x_1', x_2', t')\big)$

Images: Kernel cookbook, `https://www.cs.toronto.edu/~duvenaud/cookbook/`

Building more covariance functions

- A *sum* of kernel functions is a kernel function:

$$k(x, x') = k_1(x, x') + k_2(x, x').$$

  - e.g. $k_{\text{linear}} + k_{\text{periodic}}$

- A *product* of kernel functions is a kernel function:

$$k(x, x') = k_1(x, x') \cdot k_2(x, x')$$

  - $k_{\text{linear}} \cdot k_{\text{periodic}}$
  - $k_{\text{RBF}} \cdot k_{\text{periodic}}$
  - splitting coordinates:
    $k\big((x_1, x_2, t), (x_1', x_2', t')\big) = k_{\text{space}}((x_1, x_2), (x_1', x_2')) \cdot k_{\text{time}}(t, t')$

Images: Kernel cookbook, `https://www.cs.toronto.edu/~duvenaud/cookbook/`

Overview

- Prediction and the kernel function
- Gaussian Processes and the kernel function
- Effect of the kernel function
- Effect of measurement noise and nonstationarity

Summary: covariance functions

- covariance function expresses our assumptions on smoothness / shape of f

- $k(x, x')$ needs to be
  - symmetric
  - positive semidefinite

Summary: covariance functions

- covariance function expresses our assumptions on smoothness / shape of f
- $k(x, x')$ needs to be
  - symmetric
  - positive semidefinite
- more examples of valid covariance functions:
  - many other examples: Chapter 4 in Rasmussen & Williams
  - any function such that $k(x, x')$ is an inner product

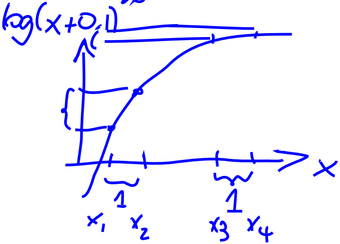
Summary: covariance functions

- covariance function expresses our assumptions on smoothness / shape of f
- $k(x, x')$ needs to be
  - symmetric
  - positive semidefinite
- more examples of valid covariance functions:
  - many other examples: Chapter 4 in Rasmussen & Williams
  - any function such that $k(x, x')$ is an inner product
- stationary: $k(x, x')$ is a function of $x - x'$
- isotropic: $k(x, x')$ is a function of $\|x - x'\|$

Nonstationary kernels

Example: $k(x, x') = \exp(-\|\log(0.1 + x) - \log(0.1 + x')\|^2)$

$k_{RBF}(x, x') = k_{RBF}(\phi(x), \phi(x'))$



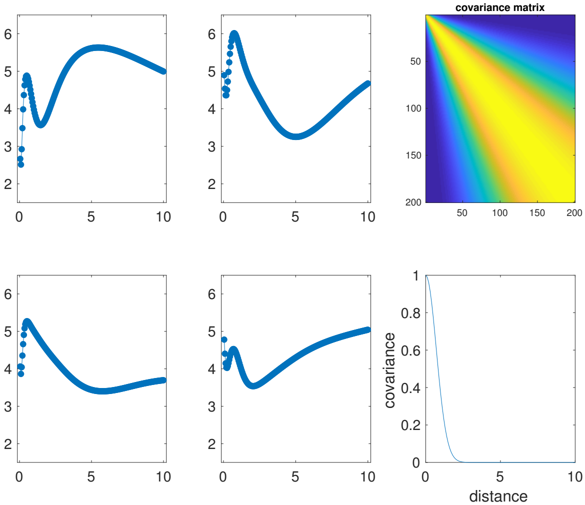
$\mathrm{dist}(\phi(x_1), \phi(x_2)) > \mathrm{dist}(\phi(x_3), \phi(x_4))$

$\Rightarrow \mathrm{cov}(Y_1, Y_2) < \mathrm{cov}(Y_3, Y_4)$

$\longrightarrow$ smoother

Nonstationary kernels

Example: $k(x, x') = \exp(-\| \log(0.1 + x) - \log(0.1 + x')\|^2)$



Noisy observations

- observed points: $y_i = \overset{y_i'}{f(x_i)} + \epsilon_i, \quad \epsilon_i \sim \mathcal{N}(0, \tau^2)$
  noise independent across locations

  how does noise affect variance / covariance?

$$
\begin{aligned}
\mathrm{cov}(y_i, y_j) &= \mathrm{cov}(y_i' + \epsilon_i, y_j' + \epsilon_j) \\
&= \mathrm{cov}(y_i', y_j') + \mathrm{cov}(\epsilon_i, \epsilon_j) \\
&= \underbrace{k(x_i, x_j) + \begin{cases} 0 & \text{if } i \neq j \\ \tau^2 & \text{if } i = j \end{cases}}_{}
\end{aligned}
$$

Noisy observations

- observed points: $y_i = f(x_i) + \epsilon_i, \quad \epsilon_i \sim \mathcal{N}(0, \tau^2)$
  noise independent across locations

  how does noise affect variance / covariance?

- new covariance matrix of observed $y_i, \ldots, y_N$: $K_N + \tau^2 I$
  add τ^2 on diagonal

Noisy observations

- observed points: $y_i = f(x_i) + \epsilon_i, \quad \epsilon_i \sim \mathcal{N}(0, \tau^2)$
  noise independent across locations

  how does noise affect variance / covariance?

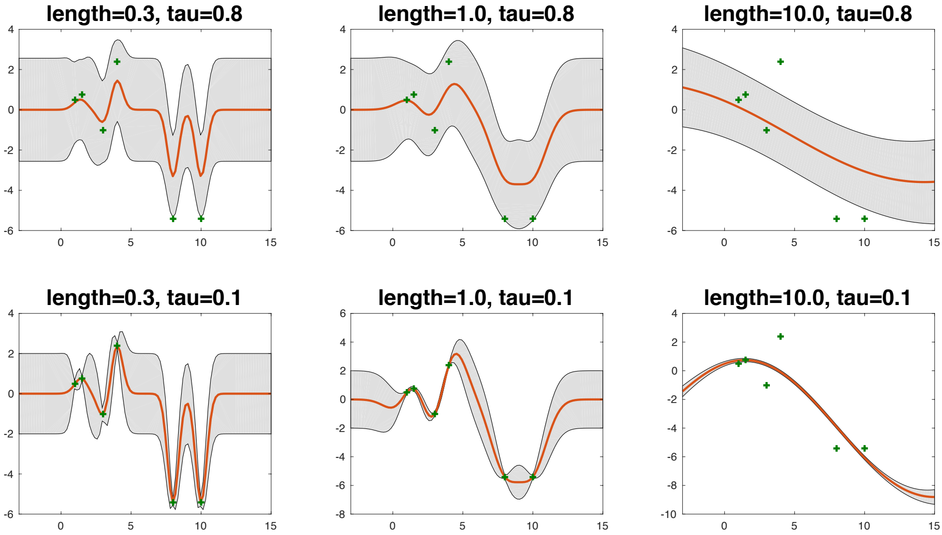
- new covariance matrix of observed $y_i, \ldots, y_N$: $K_N + \tau^2 I$
  add τ^2 on diagonal

- predicted mean and variance ($m(x) = 0$):

$$\mu_{*|1:N} = 0 + k_*^\top (K_N + \tau^2 I)^{-1} y_{1:N}$$

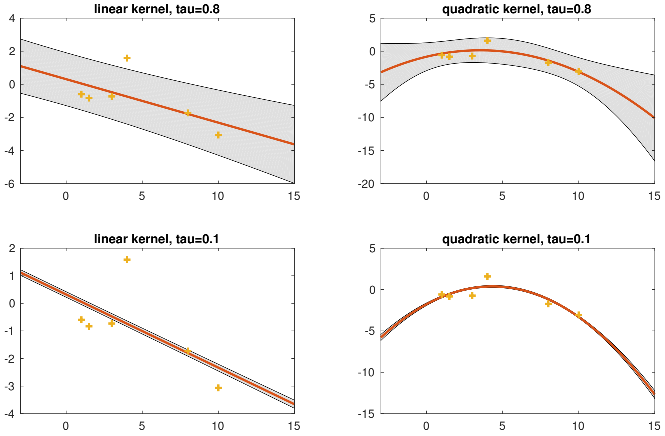
$$\sigma^2_{*|1:N} = \underbrace{k(x_*, x_*)}_{\sigma_*^2} - k_*^\top (K_N + \tau^2 I)^{-1} k_*$$

Influence of noise τ and kernel bandwidth (RBF kernel)



Influence of noise

Examples: $k(x, x') = \langle x, x' \rangle$ (linear kernel) $\qquad k(x, x') = (\langle x, x' \rangle + 1)^2$ (quadratic kernel)



General comments on Gaussian Processes

- flexible, nonlinear regression method
- kernel determines what kind of function we fit

General comments on Gaussian Processes

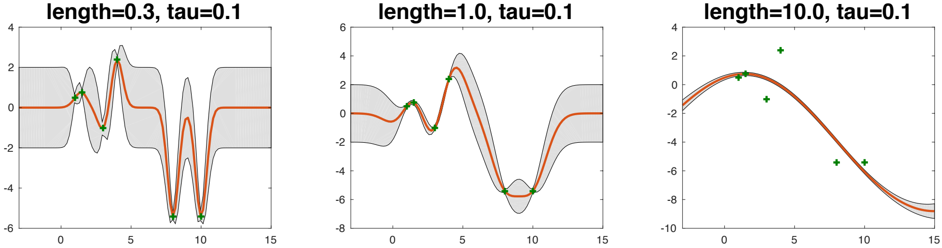
- flexible, nonlinear regression method
- kernel determines what kind of function we fit
- applicable far beyond spatial models – many settings of nonlinear regression (including time series)

General comments on Gaussian Processes

- flexible, nonlinear regression method
- kernel determines what kind of function we fit
- applicable far beyond spatial models – many settings of nonlinear regression (including time series)
- predicted variance: uncertainty. Can help guide sensor placement, measurement selection, "Bayesian Optimization"

  (e.g.: Krause, Singh, Guestrin. Near-Optimal Sensor Placements in Gaussian Processes: Theory, Efficient Algorithms and Empirical Studies. JMLR, 2008.)

Remaining Question: Which kernel?



Summary

- Prediction with Gaussian Processes: closed form to obtain Gaussian distribution for each function value
- Kernel function playes a key role in implementing our assumptions on shape and smoothness
- Measurement noise: larger variance, dampens influence of data
- Final question: how select the kernel?

References and Further Reading

- C. E. Rasmussen & C. K. I. Williams. Gaussian Processes for Machine Learning, 2006. Chapter 4, 5.4.
- D. Duvenaud. Kernel Cookbook. https://www.cs.toronto.edu/~duvenaud/cookbook/