Gaussian processes

Bayesian Methodology in Biostatistics (BST 249)

Jeffrey W. Miller

Department of Biostatistics
Harvard T.H. Chan School of Public Health

Outline

Outline

Kernel ridge regression

- Consider the linear regression model: $y = A\beta + \varepsilon$.
- The ridge regression estimate can be written in two ways:

$$\hat{\beta} = (A^{\mathsf{T}}A + \lambda I)^{-1}A^{\mathsf{T}}y = A^{\mathsf{T}}(AA^{\mathsf{T}} + \lambda I)^{-1}y$$

  by linear algebra manipulations.
- Then, given a new point x_0, we would predict

$$\hat{y}_0 = x_0^{\mathsf{T}}\hat{\beta} = x_0^{\mathsf{T}}A^{\mathsf{T}}(AA^{\mathsf{T}} + \lambda I)^{-1}y.$$

- Define $z = Ax_0$. Note that

$$z = Ax_0 = \begin{bmatrix} x_1^{\mathsf{T}} \\ \vdots \\ x_n^{\mathsf{T}} \end{bmatrix} x_0 = \begin{bmatrix} x_1^{\mathsf{T}}x_0 \\ \vdots \\ x_n^{\mathsf{T}}x_0 \end{bmatrix} = \begin{bmatrix} k(x_1, x_0) \\ \vdots \\ k(x_n, x_0) \end{bmatrix}$$

  where $k(x_i, x_j) = x_i^{\mathsf{T}}x_j$.

Kernel ridge regression

- Similarly,

$$AA^{\mathsf{T}} = \begin{bmatrix} k(x_1, x_1) & \cdots & k(x_1, x_n) \\ \vdots & & \vdots \\ k(x_n, x_1) & \cdots & k(x_n, x_n) \end{bmatrix}.$$

- $k(x_i, x_j) = x_i^{\mathsf{T}} x_j$ is a special case of a class of functions called positive semidefinite (PSD) kernels.
- Then, letting $K = AA^{\mathsf{T}}$,

$$\hat{y}_0 = x_0^{\mathsf{T}} \hat{\beta} = z^{\mathsf{T}} (K + \lambda I)^{-1} y.$$

- **Key fact**: $\hat{y}_0$ depends on the x's only through the $k(x_i, x_j)$'s.

Kernel ridge regression

- **Key fact**: $\hat{y}_0$ depends on the x's only through the $k(x_i, x_j)$'s:

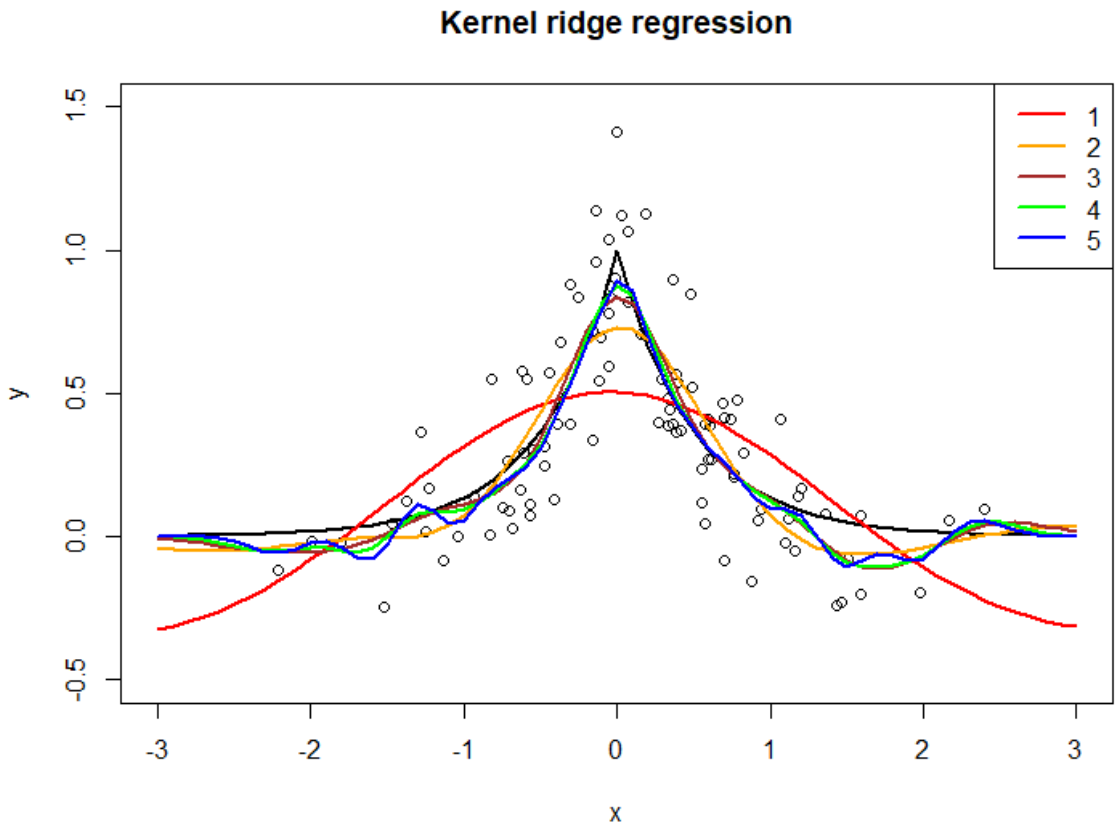
$$\hat{y}_0 = x_0^\mathsf{T}\hat{\beta} = z^\mathsf{T}(K + \lambda I)^{-1}y.$$

- What if we use a different PSD kernel $k(\cdot, \cdot)$ to compute z and K? We get a new prediction method!
- A popular choice of kernel is the squared exponential:

$$k_{\mathsf{se}}(x_i, x_j) = \exp\left(-\tfrac{1}{2\sigma^2}\|x_i - x_j\|^2\right)$$

  where $\|v\|^2 = \sum_{j=1}^p v_j^2$.
- Amazing fact: k_{se} actually corresponds to using an infinite-dimensional basis vector! Remarkably, we can bypass this infinite representation and make predictions using a finite amount of computation. This is the "kernel trick".

Kernel ridge regression: Example



Outline

Kernelization, in general

- $k(x_i, x_j) = x_i^\mathsf{T} x_j$ is a special case of a class of functions called positive semidefinite (PSD) kernels.

- Definition: $k(\cdot, \cdot)$ is a *PSD kernel* if for any $x_1, \ldots, x_n$, the matrix $K = [k(x_i, x_j)] \in \mathbb{R}^{n \times n}$ is symmetric positive semidefinite.

- Usually, $k(x_i, x_j)$ quantifies the similarity of x_i and x_j.

- Kernelization recipe, in general:
  1. Take any method that depends on the x_i's only through the dot products $x_i^\mathsf{T} x_j$.
  2. Choose a PSD kernel k.
  3. Replace every $x_i^\mathsf{T} x_j$ by $k(x_i, x_j)$.

  ...and voilà! You have a kernelized method.

Kernelization, in general

- A PSD kernel k can (usually) be expressed as

$$k(x_i, x_j) = \sum_{\ell} \varphi_\ell(x_i)\varphi_\ell(x_j) = \varphi(x_i)^{\mathtt{T}}\varphi(x_j)$$

  where the sum may be an infinite series. (Mercer's theorem)
- And any k of this form is a PSD kernel.

- More generally, if H is a Hilbert space with inner product $\langle \cdot, \cdot \rangle$, and $\varphi(x) \in H$, then $k(x, x') = \langle \varphi(x), \varphi(x') \rangle$ is a PSD kernel.

- So, kernelization is basically equivalent to using basis functions (possibly infinitely many):

$$\varphi(x_i) = (\varphi_1(x_i), \varphi_2(x_i), \ldots).$$

- What's the point, then? Why not just use basis functions? Computation, computation, computation!

Kernelization, in general

- Kernelization allows us to avoid explicitly computing $\varphi(x_i)$.
- $k(x_i, x_j)$ provides a shortcut to computing $\varphi(x_i)^\mathtt{T}\varphi(x_j)$.
  This is the "trick" in the kernel trick.
- This is advantageous for high- or infinite-dimensional $\varphi(x_i)$.
- Also, we can get flexibility without having to directly specify a bunch of basis functions.
- How do we avoid overfitting?
  Variance is controlled via regularization.

Examples of commonly used PSD kernels

- *Polynomial kernel of degree d*: For $x_i, x_j \in \mathbb{R}^p$,

$$k(x_i, x_j) = (c + x_i^\intercal x_j)^d.$$

  - Equivalent to using a certain set of polynomial basis functions up to degree d.
  - $c \geq 0$ controls the weight of lower vs higher order terms.

- *Squared exponential kernel*: For $x_i, x_j \in \mathbb{R}^p$,

$$k(x_i, x_j) = \exp\Big(-\gamma \sum_{\ell=1}^{p} (x_{i\ell} - x_{j\ell})^2 \Big).$$

  - Equivalent to using a particular infinite basis.
  - $\gamma > 0$ controls the precision/flexibility.
  - Special case of a radial basis function (RBF) kernel.

- These kernels are easy to compute compared to their basis function representations.

Group activity: Check your understanding

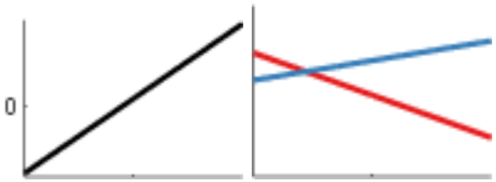
Go to breakout rooms and work together to answer these questions:
https://forms.gle/E5ECpvmDYHmiguX26

(Three people per room, randomly assigned. 15 minutes.)

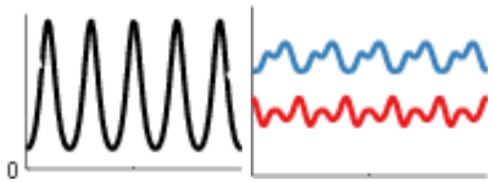
Visualizing kernels (1d)

A wide range of structures in $f(x)$ can be obtained.

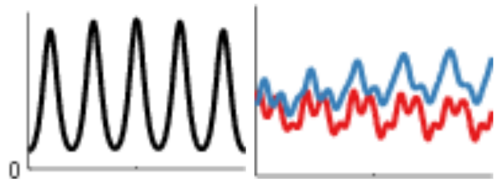
Linear Kernel



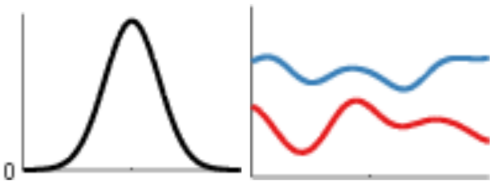
Periodic Kernel



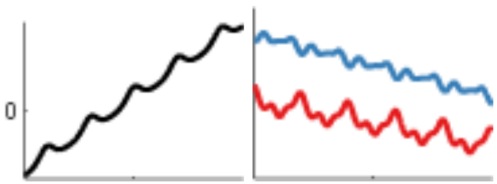
Locally Periodic Kernel



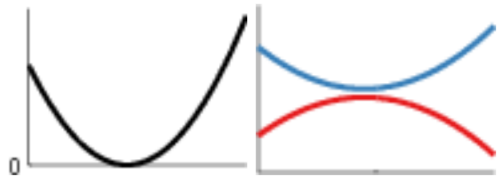
Squared Exponential Kernel



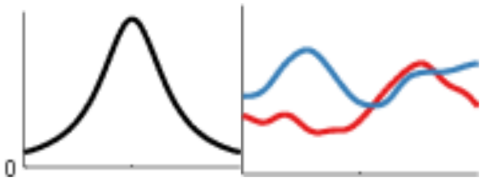
Linear plus Periodic



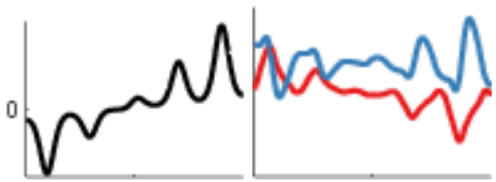
Linear times Linear



Rational Quadratic Kernel



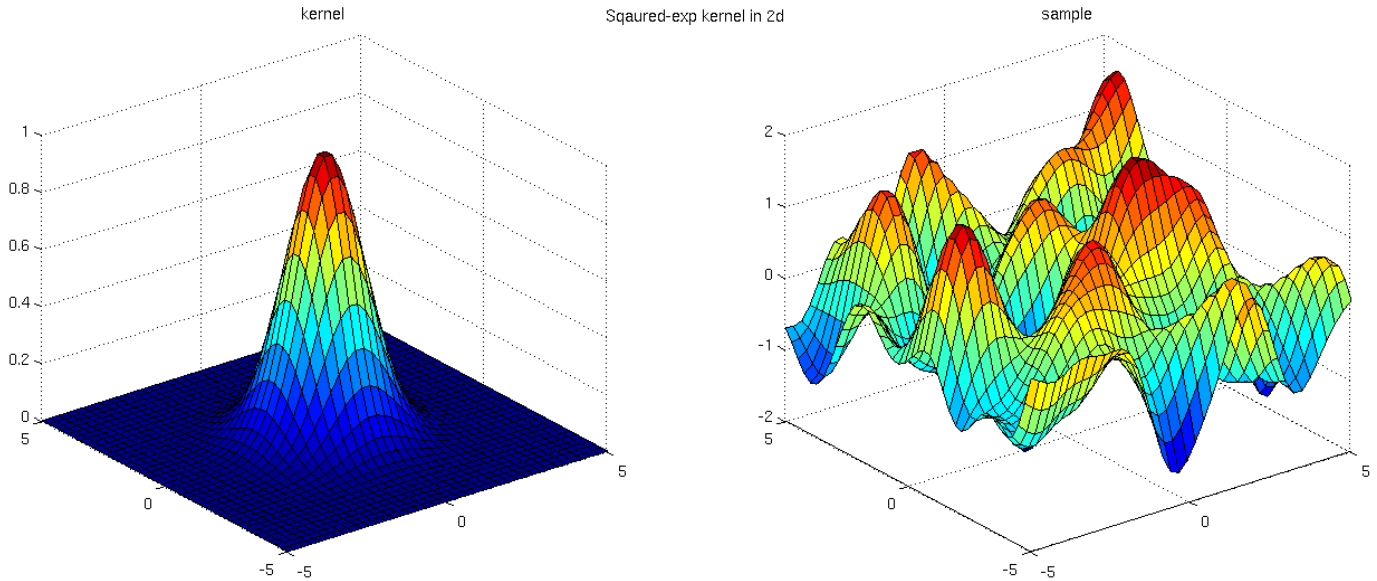
Linear times Periodic



(Figures from David Duvenaud, "Kernel cookbook", `https://www.cs.toronto.edu/~duvenaud/cookbook`)

Visualizing kernels (2d)

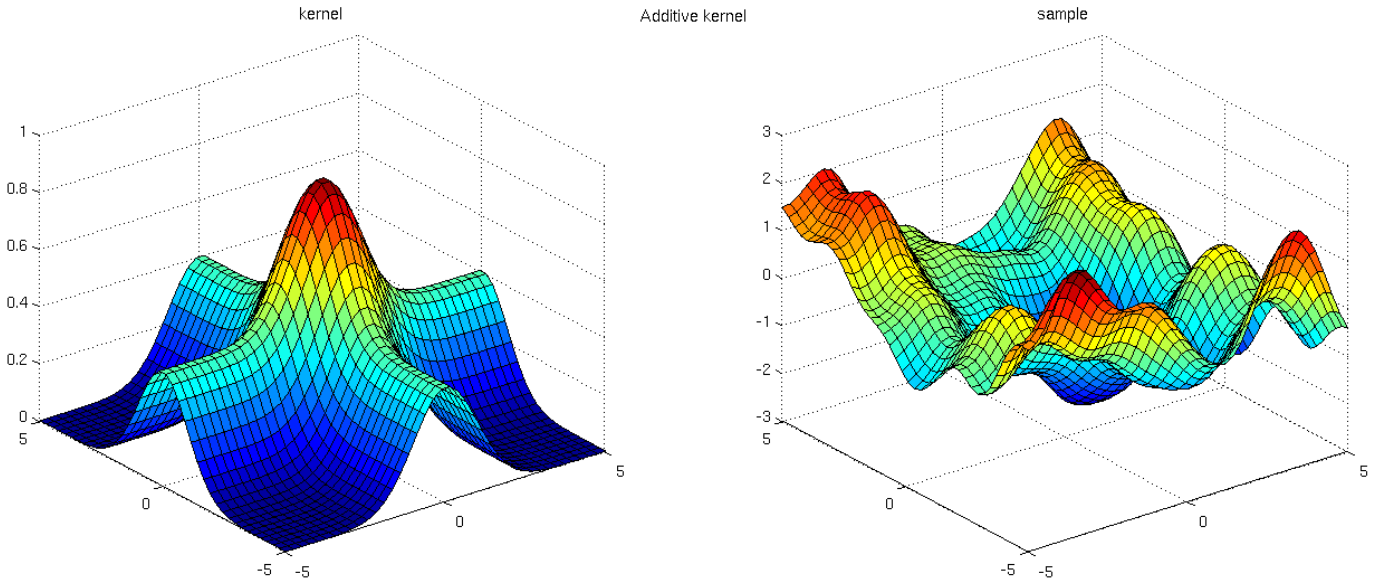
Squared exp. kernel (2d)



(Figures from David Duvenaud, "Kernel cookbook", `https://www.cs.toronto.edu/~duvenaud/cookbook`)

Visualizing kernels (2d)

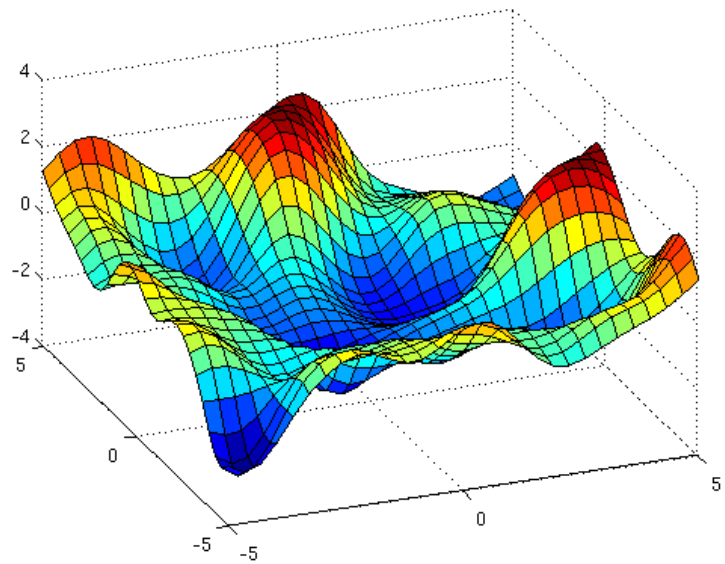
Squared exp. kernel (1d) + Squared exp. kernel (1d)



(Figures from David Duvenaud, "Kernel cookbook", `https://www.cs.toronto.edu/~duvenaud/cookbook`)

Visualizing kernels (2d)

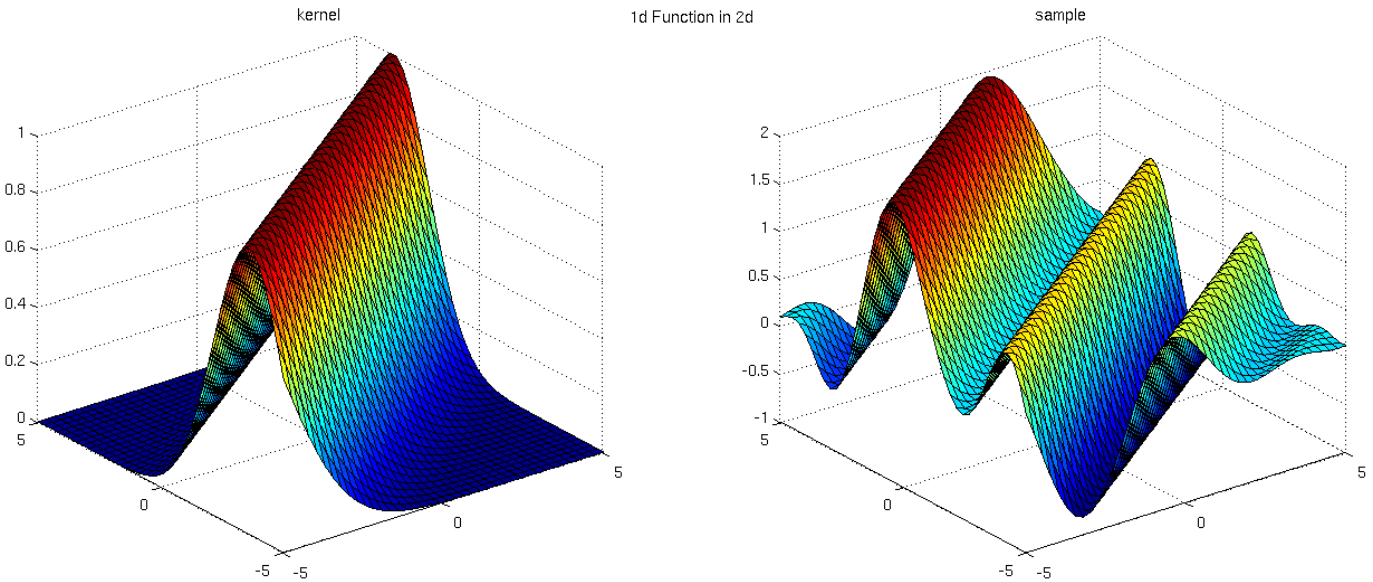
Symmetric kernel



(Figures from David Duvenaud, "Kernel cookbook", `https://www.cs.toronto.edu/~duvenaud/cookbook`)

Visualizing kernels (2d)

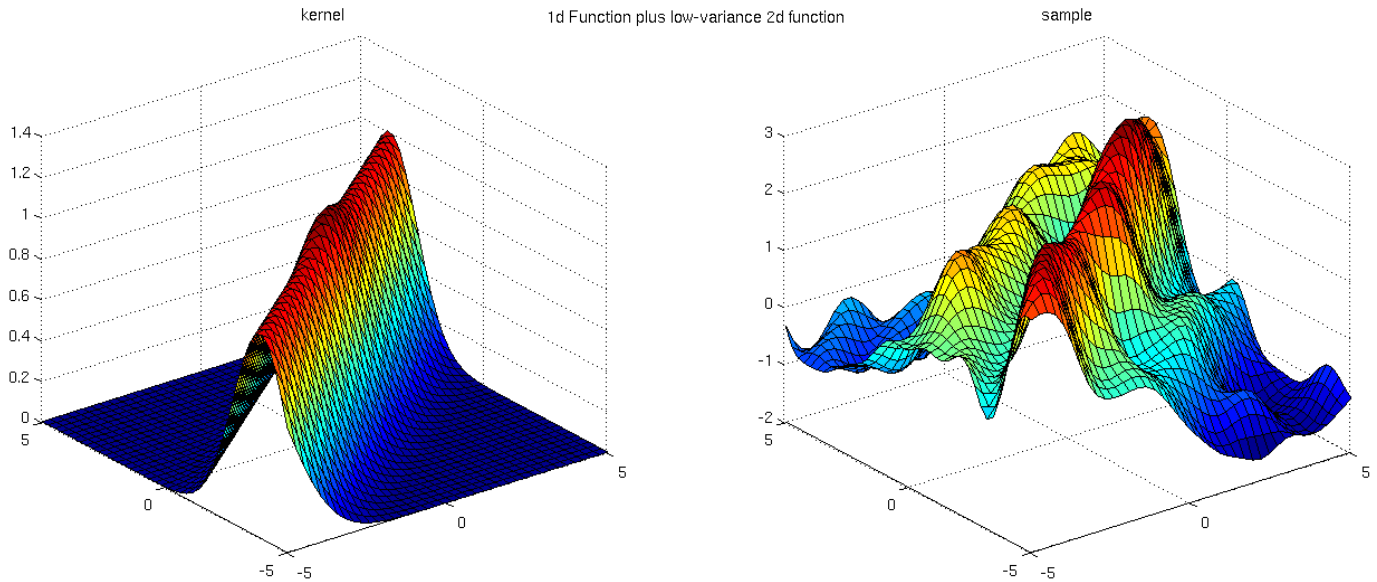
Low-dimensional subspace kernel



(Figures from David Duvenaud, "Kernel cookbook", `https://www.cs.toronto.edu/~duvenaud/cookbook`)

Visualizing kernels (2d)

Close to low-dimensional subspace



(Figures from David Duvenaud, "Kernel cookbook", `https://www.cs.toronto.edu/~duvenaud/cookbook`)

Constructing PSD kernels

If k, k_1, k_2 are PSD kernels, then the following are PSD kernels:

1. $ck(x, x')$, for any $c \geq 0$.
2. $f(x)k(x, x')f(x')$, for any $f : \mathcal{X} \to \mathbb{R}$.
3. $k_1(x, x')k_2(x, x')$
4. $k_1(x, x') + k_2(x, x')$
5. $p(k(x, x'))$ for any polynomial p with nonnegative coefs.
6. $\exp(k(x, x'))$
7. $k(\psi(x), \psi(x'))$, for any function $\psi : \mathcal{X}' \to \mathcal{X}$.

Proof: For all except #3, this is straightforward to show by using the $z^{\mathsf{T}} K z \geq 0 \ \ \forall z \in \mathbb{R}^n$ characterization of PSD matrices.

Proof that the product of PSD kernels is a PSD kernel

- Suppose $C_1, C_2 \in \mathbb{R}^{n \times n}$ are symmetric PSD matrices, and define $C \in \mathbb{R}^{n \times n}$ such that $C_{ij} = C_{1ij} C_{2ij}$.

- Let $A, B \in \mathbb{R}^{n \times n}$ such that $A^\mathsf{T} A = C_1$ and $B^\mathsf{T} B = C_2$.

- Then for any $z \in \mathbb{R}^n$,

$$
\begin{aligned}
z^\mathsf{T} C z &= \sum_{i,j} z_i z_j C_{1ij} C_{2ij} = \sum_{i,j} z_i z_j \big(\textstyle\sum_k a_{ik} a_{jk} \big) \big(\textstyle\sum_\ell b_{i\ell} b_{j\ell} \big) \\
&= \sum_{k,\ell} \sum_{i,j} z_i z_j a_{ik} a_{jk} b_{i\ell} b_{j\ell} \\
&= \sum_{k,\ell} \Big(\sum_i z_i a_{ik} b_{i\ell} \Big) \Big(\sum_j z_j a_{jk} b_{j\ell} \Big) \\
&= \sum_{k,\ell} \Big(\sum_i z_i a_{ik} b_{i\ell} \Big)^2 \geq 0.
\end{aligned}
$$

Kernelization

- The kernel trick is exploited in many methods:
  - Kernel ridge regression
  - Gaussian processes
  - Support vector machines
  - Kernel PCA
  - Spectral clustering
  - Semi-supervised learning
  - . . . and others.

Outline

Gaussian processes: Introduction

- A Gaussian process (GP) is a distribution on random functions that can be thought of as an infinite-dimensional generalization of a multivariate Gaussian.

- In Bayesian statistics, GPs are used for nonparametric regression. Basically, a GP can be used as a flexible prior on the regression function.

- Inference in GPs can be done by simply using properties of multivariate Gaussians.

- GPs have many applications, for example:
  - spatial statistics
  - meteorology
  - geostatistics
  - geology
  - oceanography
  - finance

Gaussian processes: Definition

- For any set $\mathcal{X}$, a *Gaussian process* (GP) on $\mathcal{X}$ is a set of random variables $(Z_x : x \in \mathcal{X})$ such that any finite subset $(Z_{x_1}, \ldots, Z_{x_N})$ is multivariate Gaussian.
- In other words, for all N, for all $x_1, \ldots, x_N \in \mathcal{X}$, the vector $(Z_{x_1}, \ldots, Z_{x_N})$ is multivariate Gaussian.
- The *mean function* of a GP is $\mu(x) := \mathrm{E}(Z_x)$.
- The *covariance function* (or *kernel*) of a GP is $k(x, x') := \mathrm{Cov}(Z_x, Z_{x'})$.

Gaussian processes: Examples

(Simulation examples in R)

`gp-examples.r`

Gaussian processes: Examples

1. Random subspaces: $\mathcal{X} = \mathbb{R}^d$, $\mu(x) = 0$, $k(x, x') = x^{\mathtt{T}} x'$.

2. Squared exponential: $\mathcal{X} = \mathbb{R}^d$, $\mu(x) = 0$, $k(x, x') = \exp(-\alpha \|x - x'\|^2)$ where $\alpha \geq 0$.

3. Polynomial: $\mathcal{X} = \mathbb{R}$, $\mu(x) = 0$, $k(x, x') = a(c + x^{\mathtt{T}} x')^d$ where $a, c, d \geq 0$.

4. Ornstein-Uhlenbeck: $\mathcal{X} = \mathbb{R}$, $\mu(x) = 0$, $k(x, x') = \exp(-\alpha |x - x'|)$ where $\alpha \geq 0$.

5. Periodic example: $\mathcal{X} = \mathbb{R}$, $\mu(x) = 0$, $k(x, x') = \exp\big(-\alpha \sin(\beta \pi (x - x')^2)\big)$ where $\alpha, \beta \geq 0$.

6. Symmetric example: $\mathcal{X} = \mathbb{R}$, $\mu(x) = 0$, $k(x, x') = \exp\big(-\alpha \min(|x - x'|, |x + x'|)^2\big)$ where $\alpha \geq 0$.

Existence of Gaussian processes

- For any set $\mathcal{X}$, any PSD kernel $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, and any mean function $\mu : \mathcal{X} \to \mathbb{R}$, there exists a Gaussian process $(Z_x : x \in \mathcal{X})$ such that $\mathrm{E}(Z_x) = \mu(x)$ and $\mathrm{Cov}(Z_x, Z_{x'}) = k(x, x')$.

- Proof: Kolmogorov's extension theorem.

- The nice thing about this is that it lets us define GPs with any given mean function and covariance function.

- We write $Z \sim \mathrm{GP}(\mu, k)$ to denote that Z is a GP with mean function $\mu(\cdot)$ and covariance function $k(\cdot, \cdot)$.

Outline

GP regression: Introduction

- GP regression is a Bayesian nonparametric approach to regression.
- Basic idea: Put a GP prior on the regression function, and model the outcomes as normal.
- Even though the GP is a prior on infinite-dimensional objects (Z_x), in practice, we only have a finite number of data points, so working with GPs simplifies to multivariate Gaussians.
- Prediction and inference in GP regression is essentially the same as Bayesian linear regression, and just involves some basic properties of multivariate Gaussians.

Key property of multivariate Gaussians

- Suppose $Y = Z + \varepsilon \in \mathbb{R}^N$ where $Z \sim \mathcal{N}(m, K)$ and $\varepsilon \sim \mathcal{N}(0, \sigma^2 I)$ independently.

- Then $Y \sim \mathcal{N}(m,\, K + \sigma^2 I)$.

- Let $a = (1, \ldots, n)$ and $b = (n+1, \ldots, N)$, and write

$$Y = \begin{bmatrix} Y_a \\ Y_b \end{bmatrix} \qquad m = \begin{bmatrix} m_a \\ m_b \end{bmatrix} \qquad K = \begin{bmatrix} K_{aa} & K_{ab} \\ K_{ba} & K_{bb} \end{bmatrix}.$$

- Then $Y_b \mid Y_a = y_a \sim \mathcal{N}(\eta, C)$ where

$$\eta = m_b + K_{ba}(K_{aa} + \sigma^2 I)^{-1}(y_a - m_a),$$
$$C = (K_{bb} + \sigma^2 I) - K_{ba}(K_{aa} + \sigma^2 I)^{-1} K_{ab}.$$

GP regression: Setup

- Covariates: $x_1, \ldots, x_N \in \mathcal{X}$.
- Outcomes: $y_1, \ldots, y_N \in \mathbb{R}$.

- Suppose $y_a := y_{1:n}$ is observed, and $y_b := y_{n+1:N}$ is unobserved.

- Goal: Predict and quantify uncertainty in y_b.

- Model:

$$Z = (Z_x : x \in \mathcal{X}) \sim \mathrm{GP}(\mu, k)$$
$$Y_i | Z \sim \mathcal{N}(Z_{x_i}, \sigma^2).$$

- Interpretation: Z_x is the regression function.

GP regression: Inference

- We would like to predict and quantify our uncertainty in the unobserved outcomes y_b.
- Bayesian approach: Use the posterior predictive, $Y_b \mid Y_a = y_a$.

- Let $\tilde{Z} = (Z_{x_1}, \ldots, Z_{x_N})^\intercal$. Then $\tilde{Z} \sim \mathcal{N}(m, K)$ where

$$m := \begin{bmatrix} \mu(x_1) \\ \vdots \\ \mu(x_N) \end{bmatrix} = \begin{bmatrix} m_a \\ m_b \end{bmatrix}, \quad K := \left[k(x_i, x_j)\right]_{i,j=1}^N = \begin{bmatrix} K_{aa} & K_{ab} \\ K_{ba} & K_{bb} \end{bmatrix}.$$

- By the key property above, $Y_b \mid Y_a = y_a \sim \mathcal{N}(\eta, C)$ where

$$\eta = m_b + K_{ba}(K_{aa} + \sigma^2 I)^{-1}(y_a - m_a),$$
$$C = (K_{bb} + \sigma^2 I) - K_{ba}(K_{aa} + \sigma^2 I)^{-1} K_{ab}.$$

- m is the "best fit curve" and C quantifies our uncertainty.

GP regression: Inference

Interactive demo

`http://chifeng.scripts.mit.edu/stuff/gp-demo/`

GP regression: Inference

Tutorial and interactive demos

`https://distill.pub/2019/visual-exploration-gaussian-processes/`

GP regression: Comments

- Advantages
  - Flexible nonparametric prior on regression function.
  - Wide range of dependency structures can be obtained via the choice of kernel.
- Disadvantages
  - Computation: Computing the matrix inverse $(K_{aa} + \sigma^2 I)^{-1}$ takes $O(n^3)$ time. There are approximations that reduce this to $O(n)$ for GPs.
  - Designing new kernels is something of an art, so most people use default kernels.

Outline

Supplementary reading: Application example

Application of GPs to computer model calibration

Slides 29-53 from

`https://astrostatistics.psu.edu/su18/18Lectures/w2haranGaussianProc2018.pdf`