Gaussian Process Methods in Machine Learning

Jonathan Scarlett
scarlett@comp.nus.edu.sg

Lecture 4: GP Methods in Non-Bayesian (RKHS) Settings

CS6216, Semester 1, AY2021/22

Outline of Lectures

- Lecture 0: Bayesian Modeling and Regression
- Lecture 1: Gaussian Processes, Kernels, and Regression
- Lecture 2: Optimization with Gaussian Processes
- Lecture 3: Advanced Bayesian Optimization Methods
- **Lecture 4: GP Methods in Non-Bayesian Settings**

Outline: This Lecture

- This lecture
  1. Kernels, feature maps, and the PSD property
  2. Reproducing Kernel Hilbert Space (RKHS)
  3. Kernel ridge regression and the representer theorem
  4. Bayesian optimization with non-Bayesian modeling

Motivation for Non-Bayesian Modeling

- We looked at some Bayesian Optimization theory and algorithms under the assumption that f is drawn from a GP with a known kernel

Motivation for Non-Bayesian Modeling

• We looked at some Bayesian Optimization theory and algorithms under the assumption that f is drawn from a GP with a known kernel

• **This lecture:** We can still use the same Bayesian methods even under non-Bayesian modeling, with similar guarantees

• First, we return to the study of general kernel methods

Recap on Kernels in Machine Learning

- Recap on kernels in machine learning:
  - Many machine learning algorithms depend on the data $\mathbf{x}_1, \dots, \mathbf{x}_n$ only through the inner products $\langle \mathbf{x}_i, \mathbf{x}_j \rangle$
    - Example 1: Ridge regression
    - Example 2: Dual form of Support Vector Machine (SVM)
    - Example 3: Nearest-neighbor methods
  - We know that moving to feature spaces can help, so we could map each $\mathbf{x}_i \to \phi(x_i)$ and apply the algorithm using $\langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle$
  - A kernel function $k(\mathbf{x}_i, \mathbf{x}_j)$ can be thought of as an inner product in a possibly implicit feature space
    - No need to explicitly map to feature space at all!
    - The implicit space may be infinite-dimensional (e.g., SE and Matérn), so we could not explicitly map to it even if we wanted to

PSD Kernel: Definition and Theorem

- **Definition.** A function $k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ is said to be a positive semidefinite (PSD) kernel if (i) it is symmetric, i.e., $k(\mathbf{x}, \mathbf{x}') = k(\mathbf{x}', \mathbf{x})$; (ii) For any integer $m > 0$ and any set of inputs $\mathbf{x}_1, \ldots, \mathbf{x}_m$ in $\mathbb{R}^d$, the following matrix is positive semi-definite:

$$\mathbf{K} = \begin{bmatrix} k(\mathbf{x}_1, \mathbf{x}_1) & \ldots & k(\mathbf{x}_1, \mathbf{x}_m) \\ \vdots & \ddots & \vdots \\ k(\mathbf{x}_m, \mathbf{x}_1) & \ldots & k(\mathbf{x}_m, \mathbf{x}_m) \end{bmatrix} \succeq \mathbf{0}.$$

This matrix, with (i, j)-th entry equal to $k(\mathbf{x}_i, \mathbf{x}_j)$, is called the kernel matrix (you might also see it referred to as the Gram matrix).

PSD Kernel: Definition and Theorem

• **Definition.** A function $k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ is said to be a positive semidefinite (PSD) kernel if (i) it is symmetric, i.e., $k(\mathbf{x}, \mathbf{x}') = k(\mathbf{x}', \mathbf{x})$; (ii) For any integer $m > 0$ and any set of inputs $\mathbf{x}_1, \dots, \mathbf{x}_m$ in $\mathbb{R}^d$, the following matrix is positive semi-definite:

$$\mathbf{K} = \begin{bmatrix} k(\mathbf{x}_1, \mathbf{x}_1) & \dots & k(\mathbf{x}_1, \mathbf{x}_m) \\ \vdots & \ddots & \vdots \\ k(\mathbf{x}_m, \mathbf{x}_1) & \dots & k(\mathbf{x}_m, \mathbf{x}_m) \end{bmatrix} \succeq \mathbf{0}.$$

This matrix, with (i,j)-th entry equal to $k(\mathbf{x}_i, \mathbf{x}_j)$, is called the kernel matrix (you might also see it referred to as the Gram matrix).

• **Theorem.** A function $k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ is a PSD kernel if and only if it equals an inner product $\langle \phi(\mathbf{x}), \phi(\mathbf{x}') \rangle$ for some (possibly infinite dimensional) map $\phi(\mathbf{x})$.

- **Note:** May not always be the "standard" inner product

Proof of "if" part

- **Proof of the "if" part:**
  - For simplicity, focus on the case that $\phi(\cdot)$ has finite length and the inner product is the standard one (the proof extends to the general case).
  - The inner product is certainly symmetric, and the kernel matrix can be written as $\mathbf{K} = \mathbf{\Phi}^T \mathbf{\Phi}$, where $\mathbf{\Phi} \in \mathbb{R}^{\dim(\phi) \times m}$ contains the m feature vectors $\{\phi(\mathbf{x}_t)\}_{t=1}^m$ as columns.
  - The matrix $\mathbf{K} = \mathbf{\Phi}^T \mathbf{\Phi}$ is certainly positive semidefinite, since for any $\mathbf{z}$ we have $\mathbf{z}^T \mathbf{\Phi}^T \mathbf{\Phi} \mathbf{z} = (\mathbf{\Phi} \mathbf{z})^T \mathbf{\Phi} \mathbf{z} = \|\mathbf{\Phi} \mathbf{z}\|^2 \geq 0$.

Proof of "only if" part

- **Proof of the "only if" part – finite domain:**
  - Suppose that $\mathbf{x}$ can only take values in a finite set $\{\mathbf{x}_1, \dots, \mathbf{x}_m\}$.
  - The entire function is described by an $m \times m$ matrix $\mathbf{K}_{\text{full}}$ with (i,j)-th entry $k(\mathbf{x}_i, \mathbf{x}_j)$.
  - By assumption $\mathbf{K}_{\text{full}}$ is a PSD matrix, and then it is known from linear algebra that it admits an eigenvalue decomposition of the form $\mathbf{K}_{\text{full}} = \sum_{j=1}^m \lambda_j \mathbf{v}_j \mathbf{v}_j^T$.
  - This fact allows us to consider a length-m feature map with i-th entry given by $\phi(\mathbf{x}_j) = \sqrt{\lambda_i}(\mathbf{v}_i)_j$, where $(\mathbf{v}_i)_j$ is the j-th entry of $\mathbf{v}_i$.

Proof of "only if" part

- **Proof of the "only if" part – finite domain:**
  - Suppose that $\mathbf{x}$ can only take values in a finite set $\{\mathbf{x}_1, \dots, \mathbf{x}_m\}$.
  - The entire function is described by an $m \times m$ matrix $\mathbf{K}_{\rm full}$ with (i,j)-th entry $k(\mathbf{x}_i, \mathbf{x}_j)$.
  - By assumption $\mathbf{K}_{\rm full}$ is a PSD matrix, and then it is known from linear algebra that it admits an eigenvalue decomposition of the form $\mathbf{K}_{\rm full} = \sum_{j=1}^m \lambda_j \mathbf{v}_j \mathbf{v}_j^T$.
  - This fact allows us to consider a length-m feature map with i-th entry given by $\phi(\mathbf{x}_j) = \sqrt{\lambda_i}(\mathbf{v}_i)_j$, where $(\mathbf{v}_i)_j$ is the j-th entry of $\mathbf{v}_i$.

- **Proof of the "only if" part – compact domain w/ mild continuity assumptions:**
  - The above approach can be extended to more general scenarios via Mercer's theorem, and provides an infinite-dimensional analog of the eigenvalue decomposition. (See also Bochner's theorem based on the Fourier transform.)

Proof of "only if" part

- **Proof of the "only if" part – finite domain:**
  - Suppose that $\mathbf{x}$ can only take values in a finite set $\{\mathbf{x}_1, \dots, \mathbf{x}_m\}$.
  - The entire function is described by an $m \times m$ matrix $\mathbf{K}_{\text{full}}$ with (i,j)-th entry $k(\mathbf{x}_i, \mathbf{x}_j)$.
  - By assumption $\mathbf{K}_{\text{full}}$ is a PSD matrix, and then it is known from linear algebra that it admits an eigenvalue decomposition of the form $\mathbf{K}_{\text{full}} = \sum_{j=1}^m \lambda_j \mathbf{v}_j \mathbf{v}_j^T$.
  - This fact allows us to consider a length-m feature map with i-th entry given by $\phi(\mathbf{x}_j) = \sqrt{\lambda_i}(\mathbf{v}_i)_j$, where $(\mathbf{v}_i)_j$ is the j-th entry of $\mathbf{v}_i$.

- **Proof of the "only if" part – compact domain w/ mild continuity assumptions:**
  - The above approach can be extended to more general scenarios via Mercer's theorem, and provides an infinite-dimensional analog of the eigenvalue decomposition. (See also Bochner's theorem based on the Fourier transform.)

- **Proof of the "only if" part – general case:**
  - Can be proved via the notion of a Reproducing Kernel Hilbert Space (RKHS), which we will turn to shortly (but we won't complete this proof).

Operations that Preserve the PSD Kernel Property

- **Claim.** If k_1 and k_2 are kernels, then so are the following:
  1. $k(\mathbf{x}, \mathbf{x}') = f(\mathbf{x}) k_1(\mathbf{x}, \mathbf{x}') f(\mathbf{x}')$ for some function f
  2. $k(\mathbf{x}, \mathbf{x}') = k_1(\mathbf{x}, \mathbf{x}') + k_2(\mathbf{x}, \mathbf{x}')$
  3. $k(\mathbf{x}, \mathbf{x}') = k_1(\mathbf{x}, \mathbf{x}') k_2(\mathbf{x}, \mathbf{x}')$

These properties can be useful when trying to verify whether a given $k(\cdot, \cdot)$ is indeed a PSD kernel

Operations that Preserve the PSD Kernel Property

- **Claim.** If k_1 and k_2 are kernels, then so are the following:
  1. $k(\mathbf{x}, \mathbf{x}') = f(\mathbf{x}) k_1(\mathbf{x}, \mathbf{x}') f(\mathbf{x}')$ for some function f
  2. $k(\mathbf{x}, \mathbf{x}') = k_1(\mathbf{x}, \mathbf{x}') + k_2(\mathbf{x}, \mathbf{x}')$
  3. $k(\mathbf{x}, \mathbf{x}') = k_1(\mathbf{x}, \mathbf{x}') k_2(\mathbf{x}, \mathbf{x}')$

These properties can be useful when trying to verify whether a given $k(\cdot, \cdot)$ is indeed a PSD kernel

- **Proof.** See Lecture 5 at https://www.comp.nus.edu.sg/~scarlett/CS5339_notes/

Note:

To be rigorous/robust when applying the kernel trick, the selected kernel should satisfy the PSD property

RKHS: Motivating Example

- For each $c \in \mathbb{R}$, consider defining

$$k_c(x) = e^{-(x-c)^2}.$$

Imagine producing some $f(x)$ as a weighted combination of these k_c's:

$$f(x) = \sum_{i=1}^{m} \alpha_i k_{c_i}(x)$$

for some $\alpha_1, \dots, \alpha_m$ and $c_1, \dots, c_m$. What sorts of functions can we produce?

RKHS: Motivating Example

- For each $c \in \mathbb{R}$, consider defining

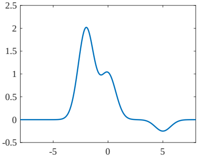
$$k_c(x) = e^{-(x-c)^2}.$$

Imagine producing some $f(x)$ as a weighted combination of these k_c's:

$$f(x) = \sum_{i=1}^{m} \alpha_i k_{c_i}(x)$$

for some $\alpha_1, \dots, \alpha_m$ and $c_1, \dots, c_m$. What sorts of functions can we produce?

- **Example function:** $(m = 3)$



  - Note: Each k_{c_i} has a Gaussian shape, but their combination is more complex

- **Idea:** Treat the $k_c(\cdot)$ as basis functions for a function space, and additionally define geometric notions of inner product and norm in this function space.
  - Perform regression (or classification / optimization) using functions from this space.

Hilbert Space

- Mathematically, a vector space is a set coupled with operations of addition and scalar multiplication that obey natural axioms (e.g., adding any two elements of the set produced another element of the set)

- A Hilbert space $\mathcal{H}$ is a vector space that additionally has a notion of inner product $\langle \cdot, \cdot \rangle_{\mathcal{H}}$ and a norm $\| \cdot \|_{\mathcal{H}}$, and satisfies a technical condition called completeness (roughly regarding limits of elements of $\mathcal{H}$ being well-behaved, e.g., like the reals but unlike the rationals)

Hilbert Space

- Mathematically, a vector space is a set coupled with operations of addition and scalar multiplication that obey natural axioms (e.g., adding any two elements of the set produced another element of the set)

- A Hilbert space $\mathcal{H}$ is a vector space that additionally has a notion of inner product $\langle \cdot, \cdot \rangle_{\mathcal{H}}$ and a norm $\| \cdot \|_{\mathcal{H}}$, and satisfies a technical condition called completeness (roughly regarding limits of elements of $\mathcal{H}$ being well-behaved, e.g., like the reals but unlike the rationals)

- **Example 1.** For fixed d, the space $\mathbb{R}^d$ is a Hilbert space with the usual notions of addition ($\mathbf{u} + \mathbf{v}$), scalar multiplication ($c\mathbf{u}$), inner product ($\langle \mathbf{u}, \mathbf{v} \rangle$), and norm ($\|\mathbf{u}\|$).

- **Example 2.** Let $\mathcal{H}$ be the set of all functions f mapping $[0,1] \to \mathbb{R}$ such that $\int_0^1 |f(x)|^2 dx < \infty$. Define the following operations:
  - Adding $f_1 \in \mathcal{H}$ and $f_2 \in \mathcal{H}$ gives $f = f_1 + f_2$, where $f(x) = f_1(x) + f_2(x)$.
  - Scalar multiplication of $c \in \mathbb{R}$ by $f_0 \in \mathcal{H}$ gives $f = cf_0$, where $f(x) = cf_0(x)$.
  - The inner product of $f_1 \in \mathcal{H}$ and $f_2 \in \mathcal{H}$ is $\int_0^1 f_1(x) f_2(x) dx$.
  - The norm of $f \in \mathcal{H}$ is $\|f\|_{\mathcal{H}} = \sqrt{\int_0^1 |f(x)|^2 dx}$.

This is an infinite-dimensional Hilbert space.

RKHS: General Definition

Reproducing Kernel Hilbert Space (RKHS)

A Reproducing Kernel Hilbert Space (RKHS) $\mathcal{H}$ with respect to a kernel k is a Hilbert space with inner product $\langle \cdot, \cdot \rangle_k$ satisfying the following:

(i) For all $\mathbf{x}$, $\mathcal{H}$ contains the function $\delta_{\mathbf{x}}(\cdot)$ defined as $\delta_{\mathbf{x}}(\mathbf{x}') = k(\mathbf{x}, \mathbf{x}')$.

(ii) The following reproducing property holds for all $f \in \mathcal{H}$ and all $\mathbf{x}$:

$$\langle f, \delta_{\mathbf{x}} \rangle_k = f(\mathbf{x}).$$

RKHS: General Definition

Reproducing Kernel Hilbert Space (RKHS)

A Reproducing Kernel Hilbert Space (RKHS) $\mathcal{H}$ with respect to a kernel k is a Hilbert space with inner product $\langle \cdot, \cdot \rangle_k$ satisfying the following:

(i) For all $\mathbf{x}$, $\mathcal{H}$ contains the function $\delta_{\mathbf{x}}(\cdot)$ defined as $\delta_{\mathbf{x}}(\mathbf{x}') = k(\mathbf{x}, \mathbf{x}')$.

(ii) The following reproducing property holds for all $f \in \mathcal{H}$ and all $\mathbf{x}$:

$$\langle f, \delta_{\mathbf{x}} \rangle_k = f(\mathbf{x}).$$

- **Example:** If $\mathcal{H}$ is $\mathbb{R}^d$ with the linear kernel $k(\mathbf{x}, \mathbf{x}') = \langle \mathbf{x}, \mathbf{x}' \rangle$, then for a fixed $\mathbf{c} \in \mathbb{R}^d$, we have $\delta_{\mathbf{c}}(\mathbf{x}) = \langle \mathbf{c}, \mathbf{x} \rangle = \sum_{i=1}^d c_i x_i$. Then:
  - Property (i) states that $\mathcal{H}$ contains all linear functions of $\mathbf{x}$, i.e., $f_{\mathbf{c}}(\mathbf{x}) = \langle \mathbf{c}, \mathbf{x} \rangle$
  - Since the sum of linear functions is still linear, further combining these "basis" functions doesn't expand the space any further.
  - Property (ii) holds under the RKHS inner product $\langle \delta_{\mathbf{c}}, \delta_{\mathbf{c}'} \rangle_k = \langle \mathbf{c}, \mathbf{c}' \rangle$.

- **Notes:**
  - See Lecture 3 of YouTube kernel lectures for a more sophisticated example
  - Given a PSD kernel, the RKHS always exists and is unique
  - Since $\delta_{\mathbf{x}}(\mathbf{x}')$ is in the RKHS, so as any f of the form $f(\mathbf{x}') = \sum_i \alpha_i \delta_{\mathbf{x}_i}(\mathbf{x}')$.

RKHS Inner Product and Norm

- The RKHS inner product $\langle \cdot, \cdot \rangle_k$ generalizes the normal "dot product" you are used to, though not always in the most intuitive way.

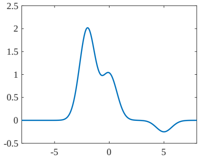
- Given the RKHS inner product, $\langle \cdot, \cdot \rangle_k$, we can define the RKHS norm as $\|f\|_k = \sqrt{\langle f, f \rangle_k}$, which roughly measures the smoothness of f.
  - Gaussian process priors encode statistical smoothness properties, but the RKHS norm encodes deterministic smoothness properties

- The useful properties of inner product and norm still apply, e.g.:
  - Linearity: $\langle f_1 + f_2, g \rangle_k = \langle f_1, g \rangle_k + \langle f_2, g \rangle_k$
  - Cauchy-Schwarz: $|\langle f, g \rangle_k| \leq \|f\|_k \|g\|_k$
  - Triangle inequality: $\|f_1 + f_2\|_k \leq \|f_1\|_k + \|f_2\|_k$

RKHS Norm

• **Example 1:** As shown above, for linear kernels all functions can be written as $f_{\mathbf{c}}(\mathbf{x}) = \mathbf{c}^T\mathbf{x}$, and the RKHS norm is given by $\|f_{\mathbf{c}}\|_k = \|\mathbf{c}\|$. Smaller $\mathbf{c}$ entries means a smoother function (slower varying).

• **Example 2:** This function we saw earlier is smooth (low $\|f\|_k$) w.r.t. the RBF kernel, but would become less so as we add more bumps, rapid fluctuations, etc.:

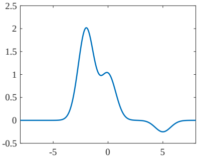


- <u>Note</u>: Even tiny changes (e.g., a sudden change in gradient) can make the RKHS norm jump to $+\infty$ (or more precisely, the function is no longer part of the RKHS)

RKHS Norm

• **Example 1:** As shown above, for linear kernels all functions can be written as $f_{\mathbf{c}}(\mathbf{x}) = \mathbf{c}^T\mathbf{x}$, and the RKHS norm is given by $\|f_{\mathbf{c}}\|_k = \|\mathbf{c}\|$. Smaller $\mathbf{c}$ entries means a smoother function (slower varying).

• **Example 2:** This function we saw earlier is smooth (low $\|f\|_k$) w.r.t. the RBF kernel, but would become less so as we add more bumps, rapid fluctuations, etc.:



- Note: Even tiny changes (e.g., a sudden change in gradient) can make the RKHS norm jump to $+\infty$ (or more precisely, the function is no longer part of the RKHS)

• **Fourier view:** If you are familiar with Fourier analysis, it's useful to know that for stationary kernels (i.e., $k(\mathbf{x}, \mathbf{x}')$ only depends on $\mathbf{x} - \mathbf{x}'$), we can write the RKHS norm in terms of Fourier transforms:

$$\|f\|_k^2 = \int \frac{|F(\boldsymbol{\xi})|^2}{K(\boldsymbol{\xi})} d\boldsymbol{\xi},$$

where $F(\cdot)$ and $K(\cdot)$ are the Fourier transforms of the function and the kernel.

Note:

Every PSD kernel has a unique RKHS that provides a norm measuring the function's smoothness (according to that kernel).

("smooth" for one kernel may be "non-smooth" for another kernel).

Kernel Ridge Regression

- Previously we derived kernel ridge regression by finding the closed-form solution to $\min_{\boldsymbol{\theta}} \sum_{i=1}^n (y_i - \boldsymbol{\theta}^T \mathbf{x}_i)^2 + \lambda \|\boldsymbol{\theta}\|^2$, expressing it in terms of inner products, and replacing those by kernel evaluations.

- We are now in a position to take a different view based on the RKHS norm:

$$\hat{f} = \arg\min_{f \in \mathcal{H}_k} \sum_{i=1}^n (y_i - f(\mathbf{x}_i))^2 + \lambda \|f\|_k^2.$$

For the linear kernel $k(\mathbf{x}, \mathbf{x}') = \langle \mathbf{x}, \mathbf{x}' \rangle$, it can be checked that this reduces to standard ridge regression.

- **Problem.** Directly solving a minimization problem over f (lying in a Hilbert space) is non-standard and difficult – how to proceed?

Representer Theorem (I)

Representer Theorem:

Consider any minimization problem of the form

$$\text{minimize}_f \; \Psi\Big(f(\mathbf{x}_1), \dots, f(\mathbf{x}_n), \|f\|_k^2\Big)$$

for some function $\Psi : \mathbb{R}^{n+1} \to \mathbb{R}$. Then, if Ψ is a strictly increasing function with respect to its final argument, the optimal solution can be expressed as

$$f(\mathbf{x}) = \sum_{i=1}^n \alpha_i k(\mathbf{x}_i, \mathbf{x})$$

for some $\alpha_1, \dots, \alpha_n$ (i.e. $f = \sum_{i=1}^n \alpha_i \delta_{\mathbf{x}_i}$).

Representer Theorem (I)

Representer Theorem:

Consider any minimization problem of the form

$$\text{minimize}_f \ \Psi\Big(f(\mathbf{x}_1), \dots, f(\mathbf{x}_n), \|f\|_k^2\Big)$$

for some function $\Psi : \mathbb{R}^{n+1} \to \mathbb{R}$. Then, if Ψ is a strictly increasing function with respect to its final argument, the optimal solution can be expressed as

$$f(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i k(\mathbf{x}_i, \mathbf{x})$$

for some $\alpha_1, \dots, \alpha_n$ (i.e. $f = \sum_{i=1}^{n} \alpha_i \delta_{\mathbf{x}_i}$).

- **Proof idea.** We can decompose any f into $f_{\mathbf{X}} + f_{\mathbf{X}}^{\perp}$, where $f_{\mathbf{X}}$ lies in the space of functions admitting the above form, and $f_{\mathbf{X}}^{\perp}$ lies in the orthogonal space. Then replacing $f_{\mathbf{X}}^{\perp}$ by zero keeps every $f(\mathbf{x}_i)$ identical, but reduces $\|f\|_k$.

- **Implication:** We can reduce an infinite-dimensional optimization over f to a finite-dimensional optimization over $\boldsymbol{\alpha} = [\alpha_1, \dots, \alpha_n]^T$

Representer Theorem (II)

- To substitute $f(\mathbf{x}) = \sum_{i=1}^n \alpha_i k(\mathbf{x}_i, \mathbf{x})$ into the original formulation of minimizing $\Psi\Big(f(\mathbf{x}_1), \dots, f(\mathbf{x}_n), \|f\|_k^2\Big)$, we would like to express each $f(\mathbf{x}_j)$ and $\|f\|_k^2$ in terms of $\boldsymbol{\alpha} = [\alpha_1, \dots, \alpha_n]^T$.
  - We will also make use of the $n \times n$ pairwise kernel matrix $\mathbf{K}$

Representer Theorem (II)

- To substitute $f(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i k(\mathbf{x}_i, \mathbf{x})$ into the original formulation of minimizing $\Psi\Big(f(\mathbf{x}_1), \dots, f(\mathbf{x}_n), \|f\|_k^2\Big)$, we would like to express each $f(\mathbf{x}_j)$ and $\|f\|_k^2$ in terms of $\boldsymbol{\alpha} = [\alpha_1, \dots, \alpha_n]^T$.
    - We will also make use of the $n \times n$ pairwise kernel matrix $\mathbf{K}$

- For $f(\mathbf{x}_j)$, we simply write

$$f(\mathbf{x}_j) = \sum_{i=1}^{n} \alpha_i k(\mathbf{x}_i, \mathbf{x}_j) = [\mathbf{K}\boldsymbol{\alpha}]_j,$$

where $[\mathbf{K}\boldsymbol{\alpha}]_j$ is the j-th entry of the vector $\mathbf{K}\boldsymbol{\alpha} \in \mathbb{R}^n$

- For $\|f\|_k^2$, substituting f and expanding the square gives

$$\|f\|_k^2 = \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j k(\mathbf{x}_i, \mathbf{x}_j) = \boldsymbol{\alpha}^T \mathbf{K} \boldsymbol{\alpha}.$$

Representer Theorem (II)

- To substitute $f(\mathbf{x}) = \sum_{i=1}^n \alpha_i k(\mathbf{x}_i, \mathbf{x})$ into the original formulation of minimizing $\Psi\Big(f(\mathbf{x}_1), \dots, f(\mathbf{x}_n), \|f\|_k^2\Big)$, we would like to express each $f(\mathbf{x}_j)$ and $\|f\|_k^2$ in terms of $\boldsymbol{\alpha} = [\alpha_1, \dots, \alpha_n]^T$.
  - We will also make use of the $n \times n$ pairwise kernel matrix $\mathbf{K}$

- For $f(\mathbf{x}_j)$, we simply write

$$f(\mathbf{x}_j) = \sum_{i=1}^n \alpha_i k(\mathbf{x}_i, \mathbf{x}_j) = [\mathbf{K}\boldsymbol{\alpha}]_j,$$

where $[\mathbf{K}\boldsymbol{\alpha}]_j$ is the j-th entry of the vector $\mathbf{K}\boldsymbol{\alpha} \in \mathbb{R}^n$

- For $\|f\|_k^2$, substituting f and expanding the square gives

$$\|f\|_k^2 = \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j k(\mathbf{x}_i, \mathbf{x}_j) = \boldsymbol{\alpha}^T \mathbf{K} \boldsymbol{\alpha}.$$

- We will proceed with the ridge regression example, but this approach applies even to methods without closed-form solutions (e.g., logistic regression)

Application to Ridge Regression

- Returning to the problem

$$\hat{f} = \arg\min_f \sum_{i=1}^n (y_i - f(\mathbf{x}_i))^2 + \lambda \|f\|_k^2,$$

we can substitute the expressions on the previous slide to get the equivalent problem

$$\hat{\boldsymbol{\alpha}} = \arg\min_{\boldsymbol{\alpha} \in \mathbb{R}^n} \; (\mathbf{K}\boldsymbol{\alpha} - \mathbf{y})^T (\mathbf{K}\boldsymbol{\alpha} - \mathbf{y}) + \boldsymbol{\alpha}^T \mathbf{K} \boldsymbol{\alpha}.$$

Application to Ridge Regression

- Returning to the problem

$$\hat{f} = \arg\min_{f} \sum_{i=1}^{n} (y_i - f(\mathbf{x}_i))^2 + \lambda \|f\|_k^2,$$

we can substitute the expressions on the previous slide to get the equivalent problem

$$\hat{\boldsymbol{\alpha}} = \arg\min_{\boldsymbol{\alpha} \in \mathbb{R}^n} \ (\mathbf{K}\boldsymbol{\alpha} - \mathbf{y})^T (\mathbf{K}\boldsymbol{\alpha} - \mathbf{y}) + \boldsymbol{\alpha}^T \mathbf{K} \boldsymbol{\alpha}.$$

- Since this is finite-dimensional, it can be solved using standard optimization solvers, though this is also a rare case where we get a closed-form solution:

$$\hat{\boldsymbol{\alpha}} = (\mathbf{K} + \lambda \mathbf{I})^{-1} \mathbf{y}.$$

Application to Ridge Regression

- Returning to the problem

$$\hat{f} = \arg\min_{f} \sum_{i=1}^{n} (y_i - f(\mathbf{x}_i))^2 + \lambda \|f\|_k^2,$$

we can substitute the expressions on the previous slide to get the equivalent problem

$$\hat{\boldsymbol{\alpha}} = \arg\min_{\boldsymbol{\alpha} \in \mathbb{R}^n} \; (\mathbf{K}\boldsymbol{\alpha} - \mathbf{y})^T (\mathbf{K}\boldsymbol{\alpha} - \mathbf{y}) + \boldsymbol{\alpha}^T \mathbf{K} \boldsymbol{\alpha}.$$

- Since this is finite-dimensional, it can be solved using standard optimization solvers, though this is also a rare case where we get a closed-form solution:

$$\hat{\boldsymbol{\alpha}} = (\mathbf{K} + \lambda \mathbf{I})^{-1} \mathbf{y}.$$

- We now apply $f(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i k(\mathbf{x}_i, \mathbf{x})$ one more time: Defining $\mathbf{k}(\mathbf{x}) = [k(\mathbf{x}_1, \mathbf{x}), \dots, k(\mathbf{x}_n, \mathbf{x})]^T$ yields $\hat{f}(\mathbf{x}) = \mathbf{k}(\mathbf{x})^T \boldsymbol{\alpha}$, and substituting $\hat{\boldsymbol{\alpha}}$ gives

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

which is exactly what we got via the kernel trick (or the Gaussian Process approach).

Note:

The representer theorem reduces an infinite-dimensional optimization problem to a finite-dimensional one, and serves as a useful alternative to the kernel trick for applying kernel methods.

Bayesian Optimization: Recap

black-box **function optimization:**

$$\mathbf{x}^\star \in \arg\max_{x \in D \subseteq \mathbb{R}^d} f(\mathbf{x})$$

- **Bayesian model:** f is a (zero-mean) GP with kernel k

- **Bayesian confidence bound:** With probability at least $1-\delta$, it holds for all $\mathbf{x} \in D$ and $t > 0$ that

$$\underbrace{\mu_{t-1}(\mathbf{x}) - \sqrt{\beta_t}\sigma_{t-1}(\mathbf{x})}_{\text{LCB}} \le f(\mathbf{x}) \le \underbrace{\mu_{t-1}(\mathbf{x}) + \sqrt{\beta_t}\sigma_{t-1}(\mathbf{x})}_{\text{UCB}}$$

- **GP-UCB algorithm:** Choose $\mathbf{x}_t$ to maximize the UCB (optimism under uncertainty)

Bayesian Optimization with Non-Bayesian Modeling

- **Non-Bayesian model:** $\|f\|_k \le B$ for some $B > 0$ (i.e., f is smooth w.r.t. the RKHS norm – smaller B means more smooth)

Bayesian Optimization with Non-Bayesian Modeling

- **Non-Bayesian model:** $\|f\|_k \le B$ for some $B > 0$ (i.e., f is smooth w.r.t. the RKHS norm – smaller B means more smooth)

- **Non-Bayesian confidence bound (simplified version):** If

$$\beta_t^{1/2} = B + \sqrt{2(\gamma_{t-1} + \ln(1/\delta))},$$

then with probability at least $1-\delta$, it holds that

$$\underbrace{\mu_{t-1}(\mathbf{x}) - \sqrt{\beta_t}\sigma_{t-1}(\mathbf{x})}_{\text{LCB}} \le f(\mathbf{x}) \le \underbrace{\mu_{t-1}(\mathbf{x}) + \sqrt{\beta_t}\sigma_{t-1}(\mathbf{x})}_{\text{UCB}}$$

(Same formula as before, but different β_t)

- Definitions: γ_{t-1} is the maximum information gain we introduced previously
- Note: We are using Bayesian update equations (μ_t and σ_t) even though the model is non-Bayesian

Bayesian Optimization with Non-Bayesian Modeling

- **Non-Bayesian model:** $\|f\|_k \leq B$ for some $B > 0$ (i.e., f is smooth w.r.t. the RKHS norm – smaller B means more smooth)

- **Non-Bayesian confidence bound (simplified version):** If

$$\beta_t^{1/2} = B + \sqrt{2(\gamma_{t-1} + \ln(1/\delta))},$$

then with probability at least $1 - \delta$, it holds that

$$\underbrace{\mu_{t-1}(\mathbf{x}) - \sqrt{\beta_t}\sigma_{t-1}(\mathbf{x})}_{\text{LCB}} \leq f(\mathbf{x}) \leq \underbrace{\mu_{t-1}(\mathbf{x}) + \sqrt{\beta_t}\sigma_{t-1}(\mathbf{x})}_{\text{UCB}}$$

(Same formula as before, but different β_t)

  - Definitions: γ_{t-1} is the maximum information gain we introduced previously
  - Note: We are using Bayesian update equations (μ_t and σ_t) even though the model is non-Bayesian

- **GP-UCB algorithm:** As before, just use the new β_t instead!
  - Similar regret bounds as the Bayesian case then follow
  - Caveat: β_t is now a lot larger for (e.g.) the Matérn kernel; improvements exist

Summary:

- The RKHS associated with k is a (Hilbert) space of functions
- The RKHS norm $\|f\|_k$ measures the level of smoothness of f, where the precise meaning of "smoothness" is dictated by the kernel k
- This gives another viewpoint on kernel ridge regression, and provides many other kernelized algorithms (not covered here)
- With only minor changes, the same Bayesian Optimization algorithms work with rigorous guarantees even under non-Bayesian (RKHS) modeling assumptions (namely, that f is any function with $\|f\|_k$ below some threshold)

Further Results and Problem Settings

- Apart from nicely complementing the Bayesian theoretical results and giving useful general tools for kernel methods, the RKHS-based model has been found to be more amenable to various theoretical studies and problem settings.

- **Example 1:** Algorithm-independent lower bounds (arXiv:1706.00090)

- **Example 2:** Reinforcement learning (arXiv:1805.08052)

- **Example 3:** Safety constraints (http://proceedings.mlr.press/v37/sui15.html)

Useful Materials

- **Full YouTube course on kernel methods (including RKHS):**
  - Lecturers: Julien Mairal and Jean-Philippe
  - Link: `https://www.youtube.com/channel/UCotztBOmGVl9pPGIN4YqcRw/videos`

- **Other resources on kernel methods:**
  - Mathematical introduction: From Zero to Reproducing Kernel Hilbert Spaces in Twelve Pages or Less
  - Comprehensive textbook: Kernel Methods in Machine Learning (Hofmann, Schölkopf, and Smola)

References I

[1] Sayak Ray Chowdhury and Aditya Gopalan.
On kernelized multi-armed bandits.
In *International Conference on Machine Learning*, 2017.

[2] Sayak Ray Chowdhury and Aditya Gopalan.
Online learning in kernelized markov decision processes.
In *Int. Conf. Art. Intel. Stats. (AISTATS)*, pages 3197–3205, 2019.

[3] David Janz, David Burt, and Javier González.
Bandit optimisation of functions in the matérn kernel rkhs.
In *International Conference on Artificial Intelligence and Statistics*, pages 2486–2495. PMLR, 2020.

[4] Jonathan Scarlett, Ilija Bogunovic, and Volkan Cevher.
Lower bounds on regret for noisy Gaussian process bandit optimization.
In *Conf. Learn. Theory (COLT)*. 2017.

[5] N. Srinivas, A. Krause, S.M. Kakade, and M. Seeger.
Information-theoretic regret bounds for Gaussian process optimization in the bandit setting.
IEEE Trans. Inf. Theory, 58(5):3250–3265, May 2012.

[6] Yanan Sui, Alkis Gotovos, Joel Burdick, and Andreas Krause.
Safe exploration for optimization with Gaussian processes.
In *Proc. Int. Conf. Mach. Learn.*, 2015.