Data-Driven Design & Analyses of Structures & Materials (3dasm)

Lecture 13

Miguel A. Bessa | **miguel_bessa@brown.edu** | Associate Professor

Outline for today

- Introduction to Gaussian Processes
    - Using Scikit-learn for Gaussian Process Regression (noiseless and noisy datasets)

Reading material: This notebook + (GPs in Section 17.2 of book)
Optional reading material (GPs bible): Rasmussen, Carl E. *Gaussian Processes for Machine Learning*. MIT press, 2006. Available online **here**

- So, today we will focus on how to train:
    - **Gaussian Processes** using **scikit-learn**

Gaussian processes

Gaussian processes are a type of machine learning algorithm categorized as a kernel method. Kernel machines can be very powerful, but also have important limitations...

Understanding Gaussian Processes

We will derive Gaussian Processes (GPs) in weight-space, instead of function-space (the results are the same).

- This way, we will see that Gaussian Processes are a very natural generalization of Bayesian linear regression!

Recap of PPD for Bayesian linear regression

Last lecture we found the PPD of Bayesian linear regression to be:

$$p(y^*|\mathbf{x}^*, \mathcal{D}) = \mathcal{N}\left(y^* \mid \boldsymbol{\phi}(\mathbf{x}^*)^T \overset{>}{\boldsymbol{\Sigma}}_w \left(\overset{<}{\boldsymbol{\Sigma}}_w^{-1} \overset{<}{\boldsymbol{\mu}}_w + \frac{1}{\sigma^2}\boldsymbol{\Phi}^T \mathbf{y}\right),\ \sigma^2 + \boldsymbol{\phi}(\mathbf{x}^*)^T \overset{>}{\boldsymbol{\Sigma}}_w \boldsymbol{\phi}(\mathbf{x}^*)\right)$$

with the previously determined covariance: $\overset{>}{\boldsymbol{\Sigma}}_w = \left(\overset{<}{\boldsymbol{\Sigma}}_w^{-1} + \frac{1}{\sigma^2}\boldsymbol{\Phi}^T\boldsymbol{\Phi}\right)^{-1}$

This long expression of the PPD is **often simplified** by considering a prior with zero mean, i.e. $\overset{<}{\boldsymbol{\mu}}_w = \mathbf{0}$:

$$p(y^*|\mathbf{x}^*, \mathcal{D}) = \mathcal{N}\left(y^* \mid \boldsymbol{\phi}(\mathbf{x}^*)^T \overset{>}{\boldsymbol{\Sigma}}_w \frac{1}{\sigma^2}\boldsymbol{\Phi}^T \mathbf{y},\ \sigma^2 + \boldsymbol{\phi}(\mathbf{x}^*)^T \overset{>}{\boldsymbol{\Sigma}}_w \boldsymbol{\phi}(\mathbf{x}^*)\right)$$

We will consider this simplification, without loss of generality (it makes the expressions slightly shorter).

Furthermore, we will also abbreviate the PPD expression by using the notation:

$\boldsymbol{\phi}^* = \boldsymbol{\phi}(\mathbf{x}^*)$

from which the PPD can be rewritten as:

$$p(y^*|\mathbf{x}^*, \mathcal{D}) = \mathcal{N}\left(y^* \mid \boldsymbol{\phi}^{*T} \overset{>}{\boldsymbol{\Sigma}}_w \frac{1}{\sigma^2} \boldsymbol{\Phi}^T \mathbf{y}\,,\, \sigma^2 + \boldsymbol{\phi}^{*T} \overset{>}{\boldsymbol{\Sigma}}_w \boldsymbol{\phi}^*\right)$$

with the previously determined covariance: $\overset{>}{\boldsymbol{\Sigma}}_w = \left(\overset{<}{\boldsymbol{\Sigma}}_w^{-1} + \frac{1}{\sigma^2}\boldsymbol{\Phi}^T\boldsymbol{\Phi}\right)^{-1}$

Also recall that $\mathbf{\Phi}$ is where we group all N evaluations of the basis functions into the $N \times M$ matrix:

$$\mathbf{\Phi} = \begin{bmatrix} \phi_0(\mathbf{x}_1) & \phi_1(\mathbf{x}_1) & \cdots & \phi_{M-1}(\mathbf{x}_1) \\ \phi_0(\mathbf{x}_2) & \phi_1(\mathbf{x}_2) & \cdots & \phi_{M-1}(\mathbf{x}_2) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_0(\mathbf{x}_N) & \phi_1(\mathbf{x}_N) & \cdots & \phi_{M-1}(\mathbf{x}_N) \end{bmatrix}$$

where N is the number of training points and M is the number of basis functions.

Moreover, if we define the matrix:

$\mathbf{K} = \mathbf{\Phi} \overset{<}{\mathbf{\Sigma}}_w \mathbf{\Phi}^T$

We can show that the PPD can be rewritten as:

$$p(y^*|\mathbf{x}^*, \mathcal{D}) = \mathcal{N}\left(y^* \mid \boldsymbol{\phi}^{*T} \overset{<}{\mathbf{\Sigma}}_w \mathbf{\Phi}^T (\mathbf{K} + \sigma^2 \mathbf{I}_N)^{-1} \mathbf{y}\,,\, \sigma^2 + \boldsymbol{\phi}^{*T} \overset{<}{\mathbf{\Sigma}}_w \boldsymbol{\phi}^* - \boldsymbol{\phi}^{*T} \overset{<}{\mathbf{\Sigma}}_w \mathbf{\Phi}^T (\mathbf{K} + \sigma^2 \mathbf{I}_N)^{-1} \mathbf{\Phi} \overset{<}{\mathbf{\Sigma}}_w \boldsymbol{\phi}^*\right)$$

I know... You don't think that this simplified anything... Be patient!

Equivalency of these two PPD expressions for Bayesian linear regression

The notes below show that the PPD we found in the last lecture for Bayesian linear regression:

$$p(y^*|\mathbf{x}^*, \mathcal{D}) = \mathcal{N}\left(y^* \mid \boldsymbol{\phi}^{*T} \hat{\boldsymbol{\Sigma}}_w \frac{1}{\sigma^2} \boldsymbol{\Phi}^T \mathbf{y}\,,\, \sigma^2 + \boldsymbol{\phi}^{*T} \hat{\boldsymbol{\Sigma}}_w \boldsymbol{\phi}^*\right)$$

with the previously determined covariance: $\hat{\boldsymbol{\Sigma}}_w = \left(\check{\boldsymbol{\Sigma}}_w^{-1} + \frac{1}{\sigma^2}\boldsymbol{\Phi}^T\boldsymbol{\Phi}\right)^{-1}$

is equivalent to the following expression (shown in 2 lines due to length):

$$p(y^*|\mathbf{x}^*, \mathcal{D}) = \mathcal{N}(y^*|\boldsymbol{\phi}^{*T} \check{\boldsymbol{\Sigma}}_w \boldsymbol{\Phi}^T (\mathbf{K} + \sigma^2 \mathbf{I}_N)^{-1} \mathbf{y}\,, \tag{1}$$

$$\sigma^2 + \boldsymbol{\phi}^{*T} \check{\boldsymbol{\Sigma}}_w \boldsymbol{\phi}^* - \boldsymbol{\phi}^{*T} \check{\boldsymbol{\Sigma}}_w \boldsymbol{\Phi}^T (\mathbf{K} + \sigma^2 \mathbf{I}_N)^{-1} \boldsymbol{\Phi} \check{\boldsymbol{\Sigma}}_w \boldsymbol{\phi}^*) \tag{2}$$

where $\mathbf{K} = \boldsymbol{\Phi} \check{\boldsymbol{\Sigma}}_w \boldsymbol{\Phi}^T$.

This result is incredibly powerful!

Rewriting the PPD for Bayesian linear regression into this form:

$$p(y^*|\mathbf{x}^*, \mathcal{D}) = \mathcal{N}(y^*|\boldsymbol{\phi}^{*T}\hat{\boldsymbol{\Sigma}}_w\boldsymbol{\Phi}^T(\mathbf{K} + \sigma^2\mathbf{I}_N)^{-1}\mathbf{y}\,, \tag{3}$$

$$\sigma^2 + \boldsymbol{\phi}^{*T}\hat{\boldsymbol{\Sigma}}_w\boldsymbol{\phi}^* - \boldsymbol{\phi}^{*T}\hat{\boldsymbol{\Sigma}}_w\boldsymbol{\Phi}^T(\mathbf{K} + \sigma^2\mathbf{I}_N)^{-1}\boldsymbol{\Phi}\hat{\boldsymbol{\Sigma}}_w\boldsymbol{\phi}^*) \tag{4}$$

where $\mathbf{K} = \boldsymbol{\Phi}\hat{\boldsymbol{\Sigma}}_w\boldsymbol{\Phi}^T$,
is very interesting because the following **three terms of the same form** arise:

$$\boldsymbol{\Phi}\hat{\boldsymbol{\Sigma}}_w\boldsymbol{\Phi}^T \quad , \quad \boldsymbol{\phi}^{*T}\hat{\boldsymbol{\Sigma}}_w\boldsymbol{\Phi}^T \quad , \quad \boldsymbol{\phi}^{*T}\hat{\boldsymbol{\Sigma}}_w\boldsymbol{\phi}^*$$

Let's pause for a minute and observe the PPD expression carefully...

For simplicity of argument (but without loss of generality), consider for the time being that $\overset{<}{\boldsymbol{\Sigma}}_w = \overset{<}{\sigma}_w^2 \mathbf{I}_N$.
In this case, we can simplify each of the above terms in a very easy form:

$$\boldsymbol{\Phi} \overset{<}{\boldsymbol{\Sigma}}_w \boldsymbol{\Phi}^T = \overset{<}{\sigma}_w^2 \boldsymbol{\Phi}\boldsymbol{\Phi}^T$$

$$\boldsymbol{\phi}^{*T} \overset{<}{\boldsymbol{\Sigma}}_w \boldsymbol{\Phi}^T = \overset{<}{\sigma}_w^2 \boldsymbol{\phi}^{*T}\boldsymbol{\Phi}^T$$

$$\boldsymbol{\phi}^{*T} \overset{<}{\boldsymbol{\Sigma}}_w \boldsymbol{\phi}^* = \overset{<}{\sigma}_w^2 \boldsymbol{\phi}^{*T}\boldsymbol{\phi}^*$$

Where we see that the covariance matrix of the prior are simple "coefficients" for the basis functions (hyperparameters!). This holds even if we did not simplify the covariance matrix to be diagonal, but it just makes this argument easier to make.

Therefore, we have to analyze what it means to have pairs of basis functions being multiplied together.

Let's start with the term $\overset{<}{\sigma}_w^2 \mathbf{\Phi}\mathbf{\Phi}^T$. The $N \times M$ basis function matrix is:

$$\mathbf{\Phi} = \begin{bmatrix} \phi_0(\mathbf{x}_1) & \phi_1(\mathbf{x}_1) & \cdots & \phi_{M-1}(\mathbf{x}_1) \\ \phi_0(\mathbf{x}_2) & \phi_1(\mathbf{x}_2) & \cdots & \phi_{M-1}(\mathbf{x}_2) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_0(\mathbf{x}_N) & \phi_1(\mathbf{x}_N) & \cdots & \phi_{M-1}(\mathbf{x}_N) \end{bmatrix}$$

So, the matrix multiplication $\overset{<}{\sigma}_w^2 \mathbf{\Phi}\mathbf{\Phi}^T$ creates an $N \times N$ matrix (let's call it $\mathbf{A} = \overset{<}{\sigma}_w^2 \mathbf{\Phi}\mathbf{\Phi}^T$ for now) where each element is given by:

$$A_{ij} = \overset{<}{\sigma}_w^2 \phi_m(\mathbf{x}_i)\phi_m(\mathbf{x}_j)$$

Note that I am using Einstein's notation where repeated indices indicate a summation, i.e. $\phi_m(\mathbf{x}_i)\phi_m(\mathbf{x}_j) = \sum_{m=1}^{M} \phi_m(\mathbf{x}_i)\phi_m(\mathbf{x}_j)$.

Similarly, the remaining two terms have the same construction.

Focusing on the term $\hat{\sigma}_w^2 \boldsymbol{\phi}^{*T} \boldsymbol{\Phi}^T$, it leads to an $1 \times N$ vector (let's call it $\mathbf{b} = \hat{\sigma}_w^2 \boldsymbol{\Phi} \boldsymbol{\Phi}^T$ for now) whose elements are:

$$b_j = \hat{\sigma}_w^2 \phi_m(\mathbf{x}^*) \phi_m(\mathbf{x}_j) \quad \text{where } j = 1, \ldots, N \text{ and } m = 1, \ldots, M$$

and where you should recall that $\mathbf{x}^*$ highlights that this is a point not seen in training (point where you want to make a prediction).

And, finally, the last term $\hat{\sigma}_w^2 \boldsymbol{\phi}^{*T} \boldsymbol{\phi}^*$ leads to a scalar (let's call it $c = \hat{\sigma}_w^2 \boldsymbol{\phi}^{*T} \boldsymbol{\phi}^*$ for now) given by:

$$c = \hat{\sigma}_w^2 \phi_m(\mathbf{x}^*) \phi_m(\mathbf{x}^*) \quad \text{where } m = 1, \ldots, M$$

Hello kernels!

So, we are seeing that all these terms that are in the PPD of Bayesian linear regression:

$$\boldsymbol{\Phi}\hat{\boldsymbol{\Sigma}}_w\boldsymbol{\Phi}^T \quad , \quad \boldsymbol{\phi}^{*T}\hat{\boldsymbol{\Sigma}}_w\boldsymbol{\Phi}^T \quad , \quad \boldsymbol{\phi}^{*T}\hat{\boldsymbol{\Sigma}}_w\boldsymbol{\phi}^*$$

are associated to a single mathematical entity:

$$k(\mathbf{x}_i, \mathbf{x}_j) = \sigma_{ij}^2 \phi_m(\mathbf{x}_i)\phi_m(\mathbf{x}_j) \quad \text{where } m = 1, \ldots, M$$

which is called a **kernel function** (a.k.a. Mercer kernel).

For example, if we consider N input points x_n **in one-dimension**, if we choose a quadratic polynomial basis function without the bias term, $\boldsymbol{\phi}(x_n) = [x_n, x_n^2]$, and assume the hyperparameter $\sigma_{ij} = 1$, then we construct a quadratic kernel:

$$k(x_i, x_j) = \phi_m(x_i)\phi_m(x_j) = x_i^2 x_j^2 + 2x_i x_j$$

for every pair of points x_i and x_j.

At this point you may say: "What the hell, Miguel? Why do we care???"

- I admit it, if you just use the same basis functions (polynomial with a given degree, i.e. a finite M), then writing Bayesian linear regression using the "kernel" formalism can be a waste of time...

BUT, here's something that will blow your mind 🤯:

- Ask yourself this question: do you know what happens if we make $M \to \infty$?

We can demonstrate that if $M \to \infty$ then the kernel function becomes a **Gaussian kernel** (a.k.a. **Radial Basis Function** or RBF kernel, a.k.a. **squared exponential kernel** or SE kernel):

$$k(\mathbf{x}_i, \mathbf{x}_j) = s^2 \exp\left(-\frac{||\mathbf{x}_i - \mathbf{x}_j||^2}{2l^2}\right)$$

where l is called the length scale of the kernel, and s^2 is often interpreted as the overall variance. However, there are more general versions of the RBF kernel (e.g. with anisotropic length scales). Note that from our derivation, we can understand that l can be related to the variance of the prior if we assume $\overset{<}{\boldsymbol{\Sigma}}_w = \overset{<}{\sigma}_w^2 \mathbf{I}$. But you don't need to think about it that way. Just think about l as a hyperparameter (just like $\overset{<}{\boldsymbol{\Sigma}}_w$ was a matrix of hyperparameters).

- (Want to see the proof for this?? Check the notes below!)

Gaussian process regression

In conclusion, we found that the Gaussian process method is simply Bayesian linear regression using nonlinear kernel functions, instead of linear basis functions.
So, we fully derived the simplest PPD for a Gaussian process:

$$p(y^*|\mathbf{x}^*, \mathcal{D}) = \mathcal{N}\left(y^* \mid \mathbf{k}^{*T}(\mathbf{K} + \sigma^2\mathbf{I}_N)^{-1}\mathbf{y}\,,\, \sigma^2 + k(\mathbf{x}^*, \mathbf{x}^*) - \mathbf{k}^{*T}(\mathbf{K} + \sigma^2\mathbf{I}_N)^{-1}\mathbf{k}^*\right)$$

where it is assuming **homoscedastic noise** σ^2, i.e. we assumed constant noise (uncertainty) in the data.

- The homoscedastic noise that we consider here is a type of uncertainty that is called **aleatoric uncertainty** (a.k.a. data uncertainty) because even if we had infinite amounts of data we could not reduce the uncertainty below σ^2 at every point.
- The predicted uncertainty, i.e. the remaining terms in the PPD variance, is the **epistemic uncertainty** (a.k.a. model uncertainty) because it quantifies the error we make from our model predictions.

For completeness, let's also explicitly write the kernel function:

$$k(\mathbf{x}^*, \mathbf{x}^*) \quad \text{where } k \text{ is a chosen kernel function}$$

the kernel vector:

$$\mathbf{k}^{*T} = [k(\mathbf{x}_1, \mathbf{x}^*), k(\mathbf{x}_2, \mathbf{x}^*), \ldots, k(\mathbf{x}_N, \mathbf{x}^*)] \quad \text{where } N \text{ is the number of training points}$$

and the kernel matrix, that is usually called **Covariance matrix**:

$$\mathbf{K} = \begin{bmatrix} k(\mathbf{x}_1, \mathbf{x}_1) & k(\mathbf{x}_1, \mathbf{x}_2) & \cdots & k(\mathbf{x}_1, \mathbf{x}_N) \\ k(\mathbf{x}_2, \mathbf{x}_1) & k(\mathbf{x}_2, \mathbf{x}_2) & \cdots & k(\mathbf{x}_2, \mathbf{x}_N) \\ \vdots & \vdots & \ddots & \vdots \\ k(\mathbf{x}_N, \mathbf{x}_1) & k(\mathbf{x}_N, \mathbf{x}_2) & \cdots & k(\mathbf{x}_N, \mathbf{x}_N) \end{bmatrix}$$

Kernels and more kernels...

The freedom that we get from chosing kernels is the best asset of Gaussian processes. The kernel choice can be viewed as a hyperparameter itself (that has additional hyperparameters). The overwhelming majority of people simply use the RBF kernel with two hyperparameters that we introduced above:

$$k(\mathbf{x}_i, \mathbf{x}_j) = s^2 \exp\left(-\frac{||\mathbf{x}_i - \mathbf{x}_j||^2}{2l^2}\right)$$

But there are many more kernels and they can be very useful!

Here's some important basic kernels:

- ARD kernel (generalization of RBF kernel)
- Matern kernel (less smooth than RBF, as it has limited degree of differentiability)
- Periodic kernel (captures repeating structure of the data)

You can see their expressions in Murphy's book (Chapter 17.1).
You can see interesting plots of these kernels **here**.

- In fact, the complete **distill.pub article** where those plots are included is a great resource because it is interactive. Highly recommended!

Kernels are very interesting entities...
Not every funciton is a kernel! They have to obbey some rules, but there's a lot of flexibility in defining a kernel.
One of the most interesting aspects of kernels is that given two valid kernels $k_1(\mathbf{x}_i, \mathbf{x}_j)$ and $k_2(\mathbf{x}_i, \mathbf{x}_j)$, we can create a new kernel using any of the following methods:

$k(\mathbf{x}_i, \mathbf{x}_j) = ck_1(\mathbf{x}_i, \mathbf{x}_j), \quad \text{for any constant } c > 0$

$k(\mathbf{x}_i, \mathbf{x}_j) = f(\mathbf{x}_i)k_1(\mathbf{x}_i, \mathbf{x}_j)f(\mathbf{x}_j), \quad \text{for any function } f$

$k(\mathbf{x}_i, \mathbf{x}_j) = q\left[k_1(\mathbf{x}_i, \mathbf{x}_j)\right], \quad \text{for any function polynomial } q \text{ with non-negative coefficients}$

$k(\mathbf{x}_i, \mathbf{x}_j) = \exp\left[k(\mathbf{x}_i, \mathbf{x}_j)\right]$

$k(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^T \mathbf{A} \mathbf{x}_j, \quad \text{for any positive semi-definite matrix } \mathbf{A}$

Even more interestingly, kernels can also be combined using **addition** or **multiplication**:

$k(\mathbf{x}_i, \mathbf{x}_j) = k_1(\mathbf{x}_i, \mathbf{x}_j) + k_2(\mathbf{x}_i, \mathbf{x}_j)$

- Adding two positive-definite kernels together always results in another positive definite kernel. This is a way to get a disjunction of the individual properties of each kernel.

$k(\mathbf{x}_i, \mathbf{x}_j) = k_1(\mathbf{x}_i, \mathbf{x}_j) \times k_2(\mathbf{x}_i, \mathbf{x}_j)$

- Multiplying two positive-definite kernels together always results in another positive definite kernel. This is a way to get a conjunction of the individual properties of each kernel.

A great resource discussing the creation of non-trivial kernels is in **Duvenaud's webpage**. Highly recommended!

Now that we know all of this, we can write a general expression for the PPD of a Gaussian process becomes:

$$p(y^*|\mathbf{x}^*, \mathcal{D}) = \mathcal{N}\left(y^* \mid \mu^* + \mathbf{k}^{*T}(\mathbf{K}+\mathbf{R})^{-1}\mathbf{y}\,,\, \sigma^{*2} + k(\mathbf{x}^*, \mathbf{x}^*) - \mathbf{k}^{*T}(\mathbf{K}+\mathbf{R})^{-1}\mathbf{k}^*\right)$$

- where we added a mean function $\mu^* = \mu(\mathbf{x}^*)$, instead of assuming it to be zero (like we did in the beginning of the lecture, for conciseness)
- where the noise $\sigma^* = \sigma(\mathbf{x}^*)$ is also allowed to change for each point, i.e. we are assuming **heteroscedcastic noise**
- and where the noise matrix $\mathbf{R}$ is (usually) diagonal but where each entry is the noise level at data point $\mathbf{x}_i$ (this at least can be measured at every training point):
  $R_{ij} = \sigma_i^2 \delta_{ij}$
  where $\sigma_i \equiv \sigma(\mathbf{x}_i)$ is the noise assumed (or measured) at each training point $\mathbf{x}_i$, and δ_{ij} is the Kronecker delta (Identity matrix).

Summary of Gaussian Processes

In fact, the previous PPD expression for Gaussian processes can be generalized by absorbing terms into the kernel (because the kernels have all those fancy properties we just talked about!):

$$p(y^*|\mathbf{x}^*, \mathcal{D}) = \mathcal{N}\left(y^* \mid \mu^* + \mathbf{k}^{*T}\mathbf{K}^{-1}\mathbf{y}\,,\, k(\mathbf{x}^*, \mathbf{x}^*) - \mathbf{k}^{*T}\mathbf{K}^{-1}\mathbf{k}^*\right)$$

- where μ^* is usually zero (it depends on the prior mean $\hat{\boldsymbol{\mu}}_w$, but often we choose the prior mean as zero, leading to $\mu^* = 0$).
- and where the noise (uncertainty) assumed in the observation distribution (previously, we assumed it to be constant: σ^2) can be incorporated in the kernel (and can even change at each input point!).
  For example, by adding a White Kernel to an RBF kernel:
  $k(\mathbf{x}_i, \mathbf{x}_j) = \sigma_i^2 \delta_{ij} + s^2 \exp\left(-\frac{||\mathbf{x}_i - \mathbf{x}_j||^2}{2l^2}\right)$

You will explore these and other things in Homework 5

In the next two lectures and in Homework 5 you are going to visualize and explore some of the effects of the hyperparameter and kernel choices.
Have fun!