Lecture 15, Apr 5, 2024

Gaussian Processes – Regression in Function-Space

- Gaussian processes are a kernelized version of Bayesian linear regression
  - Allows scaling to infinitely many basis functions
  - Priors over functions instead of parameters, which is a lot more powerful (e.g. allows specifying smoothness, periodicity, etc)
- We want to compute the posterior predictive distribution $p(y'|\boldsymbol{y}) = \dfrac{p(y', \boldsymbol{y})}{\int p(y', \boldsymbol{y})\,\mathrm{d}y'}$
  - $\boldsymbol{y}$ is the data we have, and y' is the prediction we make about the future samples
- Derivation:
  - Since we assume both Gaussian weights and noise, the distribution of targets will also be Gaussian
  - $$\begin{aligned} y = \boldsymbol{w}^T\boldsymbol{\phi}(\boldsymbol{x}) + \varepsilon \implies p\left(\begin{bmatrix} y' \\ \boldsymbol{y} \end{bmatrix}\right) &= \mathcal{N}\left(\boldsymbol{0}, \alpha \begin{bmatrix} \boldsymbol{\phi}^T(\boldsymbol{x}') \\ \boldsymbol{\Phi} \end{bmatrix} \begin{bmatrix} \boldsymbol{\phi}(\boldsymbol{x}') & \boldsymbol{\Phi}^T \end{bmatrix} + \sigma^2\boldsymbol{1}\right) \\ &= \mathcal{N}\left(\boldsymbol{0}, \alpha \begin{bmatrix} \boldsymbol{\phi}^T(\boldsymbol{x}')\boldsymbol{\phi}(\boldsymbol{x}') & \boldsymbol{\phi}^T(\boldsymbol{x}')\boldsymbol{\Phi}^T \\ \boldsymbol{\Phi}\boldsymbol{\phi}(\boldsymbol{x}') & \boldsymbol{\Phi}\boldsymbol{\Phi}^T \end{bmatrix} + \sigma^2\boldsymbol{1}\right) \end{aligned}$$
    - $\boldsymbol{x}'$ is the test point and y' is our prediction for it
    - Note $\boldsymbol{w}\mathcal{N}(\boldsymbol{0}, \alpha\boldsymbol{1})$ is our prior (regularization) and $\varepsilon \sim \mathcal{N}(0, \sigma^2)$ is the noise
  - Let the Gram matrix $\boldsymbol{K}_{\boldsymbol{X},\boldsymbol{X}} = \boldsymbol{\Phi}\boldsymbol{\Phi}^T \in \mathbb{R}^{N \times N}$, where entry ij is $\alpha\boldsymbol{\phi}^T(\boldsymbol{x}^{(i)})\boldsymbol{\phi}(\boldsymbol{x}^{(j)}) = k(\boldsymbol{x}^{(i)}, \boldsymbol{x}^{(j)})$, where $k\colon \mathcal{X} \times \mathcal{X} \mapsto \mathbb{R}$ is the kernel
  - Let $\boldsymbol{k}_{\boldsymbol{X},\boldsymbol{x}'} = \begin{bmatrix} k(\boldsymbol{x}^{(1)}, \boldsymbol{x}') & k(\boldsymbol{x}^{(2)}, \boldsymbol{x}') & \dots & k(\boldsymbol{x}^{(N)}, \boldsymbol{x}') \end{bmatrix}^T$
  - Then $p\left(\begin{bmatrix} y' \\ \boldsymbol{y} \end{bmatrix}\right) = \mathcal{N}\left(\boldsymbol{0}, \begin{bmatrix} k_{\boldsymbol{x}',\boldsymbol{x}'} & \boldsymbol{k}_{\boldsymbol{x}',\boldsymbol{X}} \\ \boldsymbol{k}_{\boldsymbol{X},\boldsymbol{x}'} & \boldsymbol{K}_{\boldsymbol{X},\boldsymbol{X}} \end{bmatrix} + \sigma^2\boldsymbol{1}\right)$
    - The Gram matrix is a covariance matrix
    - Here we have implicitly marginalized out $\boldsymbol{w}$
  - Therefore $p(y'|\boldsymbol{y}) = \mathcal{N}(\mu_p, \sigma_p^2)$ where:
    - $\mu_p = \boldsymbol{k}_{\boldsymbol{x}',\boldsymbol{X}}(\boldsymbol{K}_{\boldsymbol{X},\boldsymbol{X}} + \sigma^2\boldsymbol{1})^{-1}\boldsymbol{y}$
    - $\sigma_p^2 = k_{\boldsymbol{x}',\boldsymbol{x}'} - \boldsymbol{k}_{\boldsymbol{x}',\boldsymbol{X}}(\boldsymbol{K}_{\boldsymbol{X},\boldsymbol{X}} + \sigma^2\boldsymbol{1})^{-1}\boldsymbol{k}_{\boldsymbol{X},\boldsymbol{x}'} + \sigma^2$
    - We have written the posterior predictive distribution entirely in terms of the kernel
    - Note this is equivalent to what we derived for a GLM, with squared error, l_2 regularization and $\lambda = \dfrac{\sigma^2}{\alpha}$
  - This is known as *Gaussian process regression*
- We have developed a kernelized version of Gaussian linear regression, similar to kernelized GLMs
  - The kernels we can use for this are the same as the ones for kernelized GLMs
- Compare the time and memory requirements:
  - With normal Bayesian linear regression, i.e. GP regression in weight-space, we need an expensive matrix inversion for $\boldsymbol{\Phi}^T\boldsymbol{\Phi}$ and also to store these matrices
    - $\mathcal{O}(NM^2 + M^3)$ time
    - $\mathcal{O}(NM + M^2)$ memory
  - With the kernelized , i.e. GP regression in function-space, the cost is independent of M
    - $\mathcal{O}(N^3)$ time
    - $\mathcal{O}(N^2)$ memory
  - Similar to kernelized GLMs, using GP in function space is much more efficient when we have $M \gg N$

- Kernel selection is very important; changing the kernel drastically impacts the model, since it changes our assumptions about what possible models look like, including smoothness, periodicity, etc
  - As always kernels need to be positive definite
- We can compose new kernels from multiple kernels, by adding them together, multiplying them together, or by composing with a function as $k(x, y) = k_1(f(x), f(y))$; all these will preserve positive definiteness
  - e.g. if the data has both long-term trends and short-term trends (e.g. Mauna Loa dataset), we can add together a kernel with a large lengthscale and a kernel with a small one, to produce a better

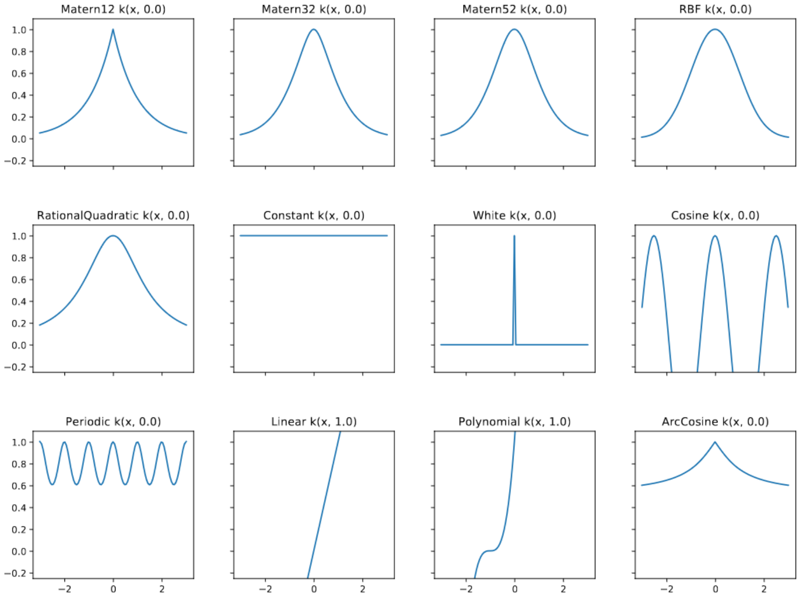

Figure 1: Visualization of some kernels in 1 dimension.

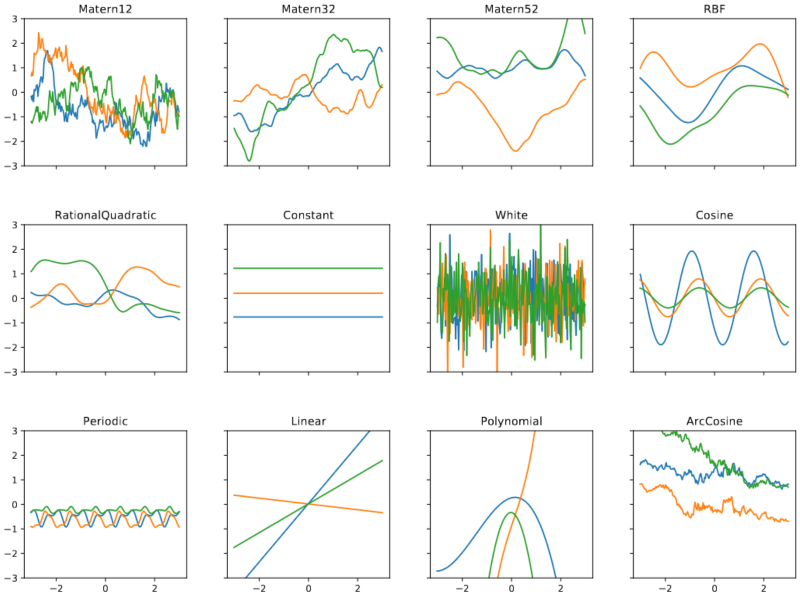

Figure 2: Visualization of the priors encoded by the kernels in the previous figure. These are different possibilities of $\hat{\boldsymbol{f}}$ sampled from the prior.

kernel overall

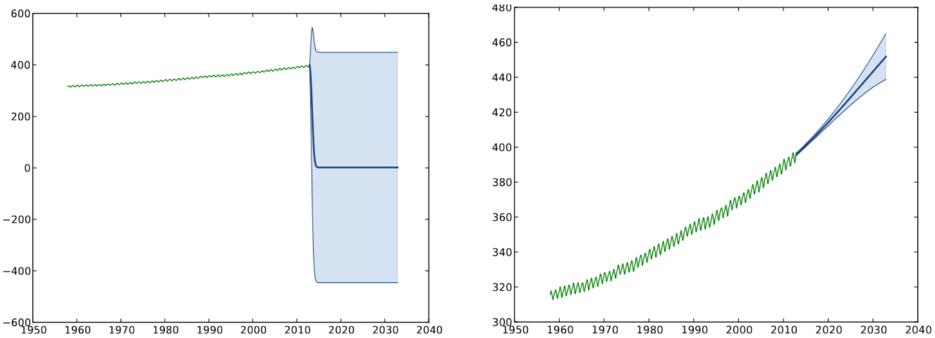

Figure 3: Examples of predictions using only a large lengthscale kernel and a small lengthscale one.

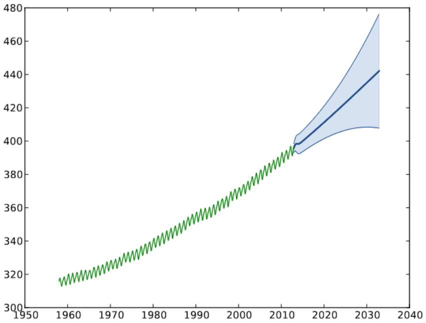

Figure 4: Predictions using the sum of both kernels.

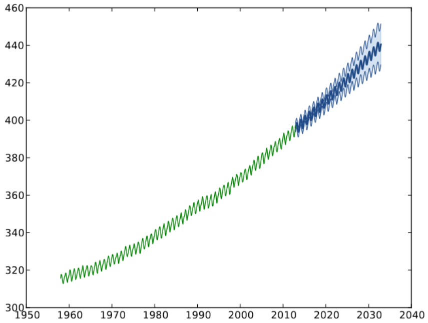

Figure 5: Predictions using the sum of two kernels with different lengthscales, plus a periodic kernel, and a degree 2 polyomial kernel.

- Kernels also have hyperparameters, e.g. in Gaussian kernel $k(x, y) = \sigma^2 e^{-\frac{(x-y)^2}{2\theta}}$ the output variance σ^2 and lengthscale $l = 1/\theta$ are important hyperparameters
- These hyperparameters can be selected through a number of means, like with Bayesian linear regression, e.g. prior knowledge, cross validation, full Bayesian inference and type-II maximum likelihood
  - Recall that in type-II maximum likelihood we try to maximize $p(\boldsymbol{y}|\boldsymbol{X})$ as a function of hyperparameters
  - $\log p(\boldsymbol{y}|\boldsymbol{X}) = -\frac{N}{2}\log\alpha - \frac{N}{2}\log(\sigma^2) - \frac{1}{2\sigma^2}\boldsymbol{y}^T\boldsymbol{y} + \frac{1}{2}\boldsymbol{\mu}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu} + \frac{1}{2}\log\det(\boldsymbol{\Sigma}) - \frac{N}{2}\log(2\pi)$
    - Used for weight space
  - $\log(\boldsymbol{y}|\boldsymbol{X}) = -\frac{N}{2}\log(2\pi) - \frac{1}{2}\log\det(\boldsymbol{K}_{\boldsymbol{X},\boldsymbol{X}} + \sigma^2\boldsymbol{1}) - \frac{1}{2}\boldsymbol{y}^T(\boldsymbol{K}_{\boldsymbol{X},\boldsymbol{X}} + \sigma^2\boldsymbol{1})^{-1}\boldsymbol{y}$

        - Used for function space

Approximate Bayesian Methods

- Generally, given a set of observed evidence, X_E and a set of unobserved variables that we want to infer, X_F, a general class of problems is computing $p(X_F|X_E) = \dfrac{p(X_E, X_F)}{p(X_E)}$
    - Often we know the joint distribution, but not the conditional distribution, because finding $p(X_E)$ is difficult or impractical
    - This is is a generalization of Bayesian inference estimation of $p(\boldsymbol{w}|\mathcal{D}) = \dfrac{p(\mathcal{D}|\boldsymbol{w})p(\boldsymbol{w})}{p(\mathcal{D})}$
        - In this case we know $p(\mathcal{D}|\boldsymbol{w})$ from our model setup + noise, and $p(\boldsymbol{w})$ from our prior on the parameters
- Since we often have $p(X_E, X_F)$, we know $p(X_F|X_E)$ up to a normalization constant, which is intractable to compute due to having to integrate $p(X_E) = \displaystyle\int p(X_E, X_F)\,\mathrm{d}X_F$
- We can try to estimate the $p(X_E)$ integral through quadrature numerical integration, but the number of points we need to sample increases exponentially with the dimensionality of X_F, making this impractical in most cases
- The *Laplace approximation* finds a Gaussian approximation of the posterior, based on a second-order Taylor approximation at the MAP
    - Let $X_F = \boldsymbol{z}$, then $p(\boldsymbol{z}|X_E) = \dfrac{1}{Z}p(X_E, \boldsymbol{z}) = \dfrac{1}{Z}\tilde{p}(\boldsymbol{z})$
    - Consider the MAP, $\hat{\boldsymbol{z}}_{\text{MAP}} = \underset{\boldsymbol{z}}{\operatorname{argmax}} \log\tilde{p}(\boldsymbol{z})$; this must be a critical point of $\log\tilde{p}(\boldsymbol{z})$, so the gradient is zero
    - The second-order Taylor expansion is then $\log p(\boldsymbol{z}|X_E) \approx \log\tilde{p}(\hat{\boldsymbol{z}}_{\text{MAP}}) - \dfrac{1}{2}(\boldsymbol{z} - \hat{\boldsymbol{z}}_{\text{MAP}})^T\boldsymbol{A}(\boldsymbol{z} - \hat{\boldsymbol{z}}_{\text{MAP}})$
        - $\boldsymbol{A} = -\vec{\nabla}^2\log\tilde{p}(\boldsymbol{z})$ is the (negative) Hessian, evaluated at $\hat{\boldsymbol{z}}_{\text{MAP}}$
            - Note we define $\boldsymbol{A}$ with a negative sign, since the Hessian at a maximum is negative-definite, but we need a positive-definite matrix later to be the covariance
        - The first-order term is zero here because the gradient is zero at a critical point
    - Exponentiate the approximation, then $p(\boldsymbol{z}|X_E) \approx \mathcal{N}(\boldsymbol{z}|\hat{\boldsymbol{z}}_{\text{MAP}}, \boldsymbol{A}^{-1})$
- The Laplace approximation is often used due to its simplicity; we only need to estimate the MAP, then approximate and invert the Hessian at the MAP
    - However, it often does a poor job
    - The main limitation is that it only approximates the posterior around the MAP and doesn't account for global properties
- We will introduce another method, based on Monte Carlo expectation approximation
    - $\mathbb{E}_{x\sim p(x)}[f(x)] \approx \dfrac{1}{M}\displaystyle\sum_{i=1}^{M} f(\boldsymbol{x}^{(i)})$ is the Monte Carlo approximation for the expectation of $f(x)$, given a distribution $p(x)$, for M samples chosen independently from $p(x)$
    - It is an unbiased estimator and has variance proportional to $\dfrac{1}{\sqrt{M}}$
    - Important, the accuracy of the Monte Carlo estimate is independent of the dimensionality of x, making it much more useful in high-dimension contexts