Recap from last time

- Gaussian processes for regression
- Conceptually: work with probability distr. on function space
- Start: $p(f_*, \bar{f} \mid X, \bar{x}_*)$ ← joint prior for all relevant function values (training + test)
- Goal: $p(f_* \mid \bar{f}, X, \bar{x}_*)$ ← Posterior pred. for $f_* \equiv f(\bar{x}_*)$ given the now known training data
- Start from Gaussian prior → end with Gaussian posterior

$$p(f_* \mid \bar{f}, X, \bar{x}_*) = \mathcal{N}(\mu_*, \sigma_*^2)$$

- Know μ_*, σ_*^2 analytically:

$$\begin{bmatrix} \mu_* = m(\bar{x}_*) + k(\bar{x}_*, X)\,\Sigma^{-1}(\bar{f} - m(X)) \\ \sigma_* = k(\bar{x}, \bar{x}_*) - k(\bar{x}_*, X)\,\Sigma^{-1}\,k(X, \bar{x}_*) \end{bmatrix}$$

Our prediction!

- $m(\cdot)$ and $k(\cdot,\cdot)$ are the mean and covariance functions we chose to define the mean vector and covariance matrix of our prior $p(f_*, \bar{f} \mid X, \bar{x}_*)$ $\left(\Sigma = k(X,X)\right)$
- Note: So far we have assumed fully specified $m(\cdot)$ and $k(\cdot,\cdot)$

Choosing and optimising covariance function (kernel)

- By choosing kernel we define our prior on function space.
- Encodes our expectations about the true function, but without specifying and assumed functional form for the true $f(\bar{x})$
- Kernels encode expectations about e.g.
  - smoothness
  - typical length scales
  - periodicity
  - trend (increase/decrease)
  - symmetry

  etc.
- This is the main modelling step in GP regression!

[Choice of mean function $m(\bar{x})$ usually much less important. Often set to $m(\bar{x}) = 0$ or $m(\bar{x}) = \text{constant}$. But keep in mind that predictions are "pulled" towards $m(\bar{x})$ in regions of $\bar{x}$ space far away from known points]

- Need a fully specified prior → fully specified kernel
- Common approach:

  1) Construct a kernel with some free params. (hyperparameters)

  2) Fit these hyperparameters in a max. likelihood fit using training data (the training step in GPR)

- Note that this is a bit "un-Bayesian".

  Proper Bayesian approach: Assign priors for the hyperparameters and marginalise over them

  $\bar{\theta}$: hyperparameters

$$\text{GP posterior} = p(f_* \mid \bar{f}, X, \bar{x}_*) = \int p(f_*, \bar{\theta} \mid \bar{f}, X, \bar{x}_*)\, d\bar{\theta}$$

$$= \int p(f_* \mid \bar{\theta}, \bar{f}, X, \bar{x}_*)\, p(\bar{\theta} \mid \bar{f}, X)\, d\bar{\theta}$$

$$\propto \int p(f_* \mid \bar{\theta}, \bar{f}, X, \bar{x}_*)\, p(\bar{f} \mid \bar{\theta}, X)\, p(\bar{\theta})\, d\bar{\theta}$$

- But this integration is usually very expensive

- So common approach is to instead use a point est. for $\bar{\theta}$, namely the $\bar{\theta}$ value that maximises the likelihood

$$L(\bar{\theta}) = p(\bar{f} \mid \bar{\theta}, X) = N(m(X), \Sigma(\bar{\theta}))$$

- Effectively like "peeking" at the data and setting a delta function prior $p(\bar{\theta}) = \delta(\bar{\theta} - \bar{\theta}_{ML})$. ("Empirical Bayes")

- Usually maximise the log-likelihood

$$\ln L(\bar{\theta}) = -\frac{1}{2}(\bar{f} - m(X))^T \Sigma^{-1}(\bar{\theta})\,(\bar{f} - m(X)) - \frac{1}{2}\ln|\Sigma(\bar{\theta})| - \frac{n}{2}\ln(2\pi)$$

- Note: For every $\bar{\theta}$ we try we need a new matrix inverse $\Sigma^{-1}(\bar{\theta})$ and determinant $|\Sigma(\bar{\theta})|$

  - Standard techniques for matrix inversion: $O(n^3)$
  - Can use Cholesky decomp. technique instead: $O(n^2)$

  Still very slow when n grows large [See algo. 2.1 in Rasmussen and Williams]

Some kernels

[Browse scikit-learn documentation to see kernel examples.]

1) The squared-exponential kernel

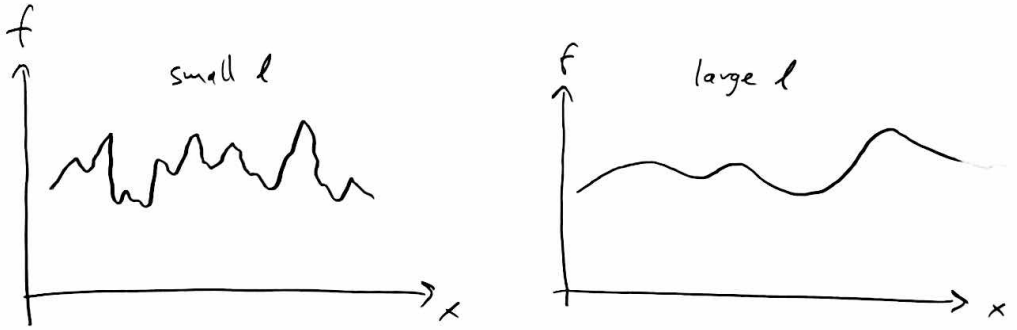
$$k(\bar{x}, \bar{x}') = \sigma_f^2 \, e^{-\frac{1}{2}\frac{(\bar{x}-\bar{x}')^2}{\ell^2}}$$

Hyperpars.:

$$\bar{\theta} = \left\{ \sigma_f^2, \ell \right\}$$

As the Euclidean dist. between the input points $\bar{x}, \bar{x}'$ increases, the covariance between $f(\bar{x})$ and $f(\bar{x}')$ decreases exponentially.

- Universal kernel (can approx any cont. function given enough data)
- Prefers very smooth functions
- σ_f^2 : Scale factor, sets average dist away from the mean function
- ℓ : Length scale, sets how quickly correlation between f-values drop as $\bar{x}$-dist. increase. Sets the typical length scale for "wiggles" in the function



- Example of stationary kernel, i.e. only depends on distance between $\bar{x}$ and $\bar{x}'$, not the values of $\bar{x}$ and $\bar{x}'$ in an absolute sense. (The kernel acts the same across all of $\bar{x}$ space). A non-stationary kernel depends on the specific locations of $\bar{x}$ and $\bar{x}'$ in $\bar{x}$ space.

2) The Matérn kernel

$$k(\bar{x},\bar{x}') = \frac{2^{1-\nu}}{\Gamma(\nu)}\left[\sqrt{2\nu}\,\frac{|\bar{x}-\bar{x}'|}{\ell}\right]^{\nu} K_\nu\left[\sqrt{2\nu}\,\frac{|\bar{x}-\bar{x}'|}{\ell}\right]$$

- Hyperparams: $\bar{\theta} = \{\nu, \ell\}$
- $\Gamma(\nu)$: gamma function
- K_ν: modified Bessel function
- Common choices: $\nu = \frac{1}{2}, \frac{3}{2}, \frac{5}{2}$
- ν sets smoothness, $\nu \to \infty$ corresponds to the squared exp. kernel

- Other common kernels [Browse the Kernel Cookbook for examples.]
  - Linear kernel
  - Periodic kernel
  - Rational quadratic kernel (effectively infinite sum of squared-exp. kernels w/ different length scales)
  - Noise kernel

- Kernels can be summed and multiplied to construct new kernels

$k(\bar{x},\bar{x}') = k_1(\bar{x},\bar{x}')\,k_2(\bar{x},\bar{x}')$: "AND operator" (Both k_1 and k_2 must be large for k to be large)

$k(\bar{x},\bar{x}') = k_1(\bar{x},\bar{x}') + k_2(\bar{x},\bar{x}')$: "OR operator"

- Can use different kernel components for different input components, e.g. with $\bar{x} = [x_1, x_2]$

$$k(\bar{x}, \bar{x}') = e^{-\frac{1}{2}\frac{(x_1 - x_1')^2}{l_1^2}} \; e^{-\frac{1}{2}\frac{(x_2 - x_2')^2}{l_2^2}}$$

to allow different lengthscales in different directions in $\bar{x}$ space. (But end up with more hyperpars. to determine.)

Noisy data

- So far we've focused on the case of <u>noise-free data</u>, i.e. training data were values of the true $f(\bar{x})$ directly.

$$\Rightarrow \text{Got posterior } p(f_* \mid \bar{f}, X, \bar{x}_*)$$

- In this case $p(f_* \mid \bar{f}, X, \bar{x}_*) \to \delta(f_* - f_i)$ when $\bar{x}_* \to \bar{x}_i$, i.e. posterior collapses to deltafunction when $\bar{x}_*$ is a a known point

- Reasonable in theory (given assumption of training points with no uncert.), but in practice numerically problematic

- Need to allow for uncertainty in data, either because this is actually the case, or just for numerical stability

- Assume data y_i are from

$$Y_i \equiv Y(\bar{x}_i) = f(\bar{x}_i) + \varepsilon_i$$

ε_i: noise, $\varepsilon \sim \mathcal{N}(0, \sigma_\varepsilon^2)$

σ_ε^2: noise level

[Note: We say "noise", but there doesn't have to be randomness involved. $\mathcal{N}(0, \sigma_\varepsilon^2)$ could just express our degree of certainty in the training values]

- Now we must distinguish between

1) $p(f_* \mid \bar{y}, X, \bar{x}_*)$: Degree of belief in underlying true function value f_* at $\bar{x}_*$

and

2) $p(y_* \mid \bar{y}, X, \bar{x}_*)$: Degree of belief in value for a data point y_* at $\bar{x}_*$

- If we only add the noise variance to the <u>training set</u> part of the covariance matrix, we get case 1:

$$\Sigma \longrightarrow \Sigma + \sigma_\varepsilon^2 I$$

[Note: we can use different uncert. at different training points.]

$$\Rightarrow \quad p(f_* \mid \bar{y}, X, \bar{x}_*) = \mathcal{N}(\mu_*, \sigma_*^2)$$

where

$$\mu_* = m(\bar{x}_*) + k(\bar{x}_*, X)\left[\Sigma + \sigma_\varepsilon^2 I\right]^{-1}(\bar{y} - m(X))$$

$$\sigma_*^2 = k(\bar{x}_*, \bar{x}_*) - k(\bar{x}_*, X)\left[\Sigma + \sigma_\varepsilon^2 I\right]^{-1} k(X, \bar{x}_*)$$

- If we add noise variance to <u>both</u> Σ and $k(\bar{x}_*, \bar{x}_*)$ we get case 2:

$$\Sigma \rightarrow \Sigma + \sigma_\varepsilon^2 I$$

$$k(\bar{x}_*, \bar{x}_*) \rightarrow k(\bar{x}_*, \bar{x}_*) + \sigma_\varepsilon^2$$

$$\Rightarrow \quad p(y_* \mid \bar{y}, X, \bar{x}_*) = N(\mu_*, \sigma_*^2)$$

where

$$\mu_* = m(\bar{x}_*) + k(\bar{x}_*, X)\left[\Sigma + \sigma_\varepsilon^2 I\right]^{-1}(\bar{y} - m(X)) \quad \left(\text{same as before}\right)$$

$$\sigma_*^2 = k(\bar{x}_*, \bar{x}_*) + \sigma_\varepsilon^2 - k(\bar{x}_*, X)\left[\Sigma + \sigma_\varepsilon^2 I\right]^{-1} k(X, \bar{x}_*)$$

↑ note!

- Common approach: Try to learn the typical noise level from data by treating σ_ε^2 as a hyperparameter, i.e. <u>add</u> a kernel term of the form $k(\bar{x}, \bar{x}') = \delta_{\bar{x}\bar{x}'}\,\sigma_\varepsilon^2$

- Pitfall: In hyperparameter space, there will then often be a huge region towards <u>large</u> σ_ε^2 where the model can obtain OK fit to data simply by "assuming" that all variation in the data is due to noise.

[See documentation of scikit-learn for example.]

Some final points

- **GPs "=" infinite NNs**

  A fully connected, infinitely wide neural network with independent and identically distributed (i.i.d.) network parameters $\Longleftrightarrow$ a Gaussian process!

  [Basically a conseq. of the Central Limit Theorem.]

- **Underest. of uncertainty**

  Standard approach is to fix hyperparameters ($\bar{\theta}$) by fit to (learning from) training data. However, standard expression for pred. variance σ_*^2 does not account for uncertainty in our $\bar{\theta}$ estimate! $\Longrightarrow$ [systematic underest. of uncert., visible when extrapolating.]

  [See arxiv: 1606.03865]

- **Regularisation of covariance matrix**

  Even in problems where there is zero data noise, the GP results can be better by introducing a small "nugget"/noise term along diagonal in cov. matrix. Makes it easier to compute inverse Σ^{-1} without loss of numerical precision.

  - Keep an eye on the condition number $K(\Sigma) = \frac{\lambda_{max}}{\lambda_{min}}$. When $K \to \infty$, $\Sigma \to$ singular (not invertible)
  - Typically gets worse with many (and very close) training points.

How to deal with the large-data limit ?

- Basic problem : Computation of Σ^{-1} too expensive when n gets very large [$> O(10^4)$]
- Many approaches, most are based on splitting up data / making Σ more sparse. Some examples:
  - Distributed Gaussian Processes, arxiv : 1502.02843
  - Generalized Robust Bayesian Committee Machine, arxiv : 1806.00720
  - Sparse Gaussian Processes (replace large training set with a smaller set of "inducing points")
  - Deep Kernel Learning, arxiv : 1511.02222 (use large datasets to train a neural network which can act as kernel function for a GP)

  + much more !