1 Gaussian Processes Coursework

The intention of this coursework is for you to get a better understanding of Gaussian processes by implementing Gaussian process regression.

Provided to you are **IDAPI_GP_Coursework.py**, a skeleton file in which you will provide solutions, and **yacht_hydrodynamics.data**, a file containing a dataset on yacht hydrodynamics we will be using to test your solutions. More information about the dataset can be found here: `https://archive.ics.uci.edu/ml/datasets/Yacht+Hydrodynamics`. You are given the function **loadData** which returns the yacht hydrodynamics dataset partitioned into a training set, $\mathcal{D}_{\text{train}} = \{X, \mathbf{y}\}$, and a test set, $\mathcal{D}_{\text{test}} = \{X_*, \mathbf{y}_*\}$.

Submit your final version to CATe via the LabTS system.

If at any point you are experiencing trouble with numerical stability, the mathematical appendix in Rasmussen & Williams may give some helpful pointers:
`http://www.gaussianprocess.org/gpml/`.
Generally, we consider the regression setting

$$y = f(\boldsymbol{x}) + \epsilon, \quad \epsilon \sim \mathcal{N}\left(0, \sigma_n^2\right)$$

We will place a GP prior on f with mean function $m \equiv 0$ and a covariance function k.

Task 1: 10 marks

Complete the definition of the function **multivariateGaussianSample**. This function takes a mean vector, μ, and covariance matrix, Σ, and returns a sample drawn from $\mathcal{N}(\mu, \Sigma)$. This will be useful if you want to visualise draws from a GP prior or posterior.

Task 2: 20 marks

Complete the definition of the function **covMatrix**. In this coursework, we are only considering a single kernel / covariance function: the squared exponential (Gaussian / radial-basis-function) kernel plus a White-Noise kernel to account for the Gaussian likelihood (noise model):

$$k(\mathbf{x}_p, \mathbf{x}_q) = \sigma_f^2 \exp\left(-\frac{1}{2\ell^2}\|\mathbf{x}_p - \mathbf{x}_q\|^2\right) + \sigma_n^2 \delta_{pq} \qquad \delta_{pq} = \begin{cases} 1 & \text{if } p = q \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

We included the contribution of the Gaussian likelihood in the kernel definition ($\sigma_n^2 \delta_{pq}$) as this will simplify the implementation of the model.
For reasons explained in Task 5, we will not be considering the parameters $\sigma_f^2, \ell, \sigma_n^2$ directly. **Instead** we will be considering the log parameters $\ln \sigma_f, \ln \ell, \ln \sigma_n$ given

by the identities in eq. (3). This is why the class **RadialBasisFunction** is initialised using the log parameters, but computes the value of the parameters via the identities as well.
The function **covMatrix** should return the kernel matrix K, where $K = K(A, A)$ is the kernel matrix computed for a set of points, A, using eq. (1).

Task 3: 25 marks

Complete the definition of the function **predict**, which takes a set of test points X_* and computes the posterior mean, $\bar{\mathbf{f}}_*$, and covariance, $\text{cov}(\mathbf{f}_*)$, of the GP regression for X_*.

Task 4: 5 marks

Complete the definition of the function **logMarginalLikelihood**, which computes the *negative log marginal likelihood* of the training set. Note: our optimiser, provided for you in the function **optimize**, minimises the target function, so please return the negative log marginal likelihood:

$$-\log p(\mathbf{y}|X) = \frac{1}{2}\mathbf{y}^\top K^{-1}\mathbf{y} + \frac{1}{2}\log|K| + \frac{n}{2}\log 2\pi \tag{2}$$

Task 5: 25 marks

Complete the definition of the function **gradLogMarginalLikelihood** which computes the gradients of the negative log marginal likelihood you found in Task 4. The function **optimize** will minimise the negative log marginal likelihood on the training set using these gradients via the BFGS algorithm.
Note: we could optimise the parameters of the GP using the constraints $\sigma_f^2, \ell, \sigma_n^2 > 0$, but a simpler method would be to solve the unconstrained optimisation problem for the log parameters using the identities:

$$\begin{aligned} \sigma_f^2 &= e^{2\ln\sigma_f} \\ \ell &= e^{\ln\ell} \\ \sigma_n^2 &= e^{2\ln\sigma_n} \end{aligned} \tag{3}$$

Optimisation is then accomplished by replacing each instance of $\sigma_f^2, \ell, \sigma_n^2$ in eq. (2) with the corresponding identity in eq. (3) and differentiating the rewritten negative marginal log likelihood with respect to the log parameters, $\ln\sigma_f, \ln\ell, \ln\sigma_n$.

Task 6: 5 marks

Using **optimize** and initial parameter values $\{\sigma_f^2 = 1.0, \ell = 0.1, \sigma_n^2 = 0.5\}$, (so, initial log parameter values are $\{\log\sigma_f = 0.5\log(1.0), \log\ell = \log(0.1), \log\sigma_n = 0.5\log(0.5)\}$) find the optimal parameters for the GP regression.

Task 7: 10 marks

Complete the definitions of the test statistics functions **mse** and **msll** and compute the MSE and MSLL for the test set using your trained GP regression.
The function **mse** computes the *mean squared error* on the test set $\{X_*, \mathbf{y}_*\}$ using the observed values $\mathbf{y}_*$ and the predictions, $\bar{\mathbf{f}}_*$, for the test input values, X_*.

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}\left(y_*^{(i)} - \bar{f}(\mathbf{x}_*^{(i)})\right)^2 \tag{4}$$

The function **msll** computes the *mean standardised log loss* on the test set $\{X_*, \mathbf{y}_*\}$ using the observed values $\mathbf{y}_*$ and the predictions, $\bar{\mathbf{f}}_*$, for the test input values, X_*, and $\text{cov}(\mathbf{y}_*)$, the covariance of the predictive distribution of the noisy test data.

$$\text{MSLL} = \frac{1}{n}\sum_{i=1}^{n} -\log p(y_*^{(i)}|\mathcal{D}_{\text{train}}, \mathbf{x}_*^{(i)}) = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{2}\log(2\pi\sigma^2(\mathbf{x}_*^{(i)})) + \frac{\left(y_*^{(i)} - \bar{f}(\mathbf{x}_*^{(i)})\right)^2}{2\sigma^2(\mathbf{x}_*^{(i)})} \tag{5}$$

$\sigma^2(\mathbf{x}_*^{(i)})$ is the predictive variance given by $\sigma^2(\mathbf{x}_*^{(i)}) = \mathbb{V}(f_*^{(i)}) + \sigma_n^2$. $\mathbb{V}(f_*^{(i)})$ denotes the predictive variance of the function value for test case i.