STA 414/2104, Spring 2006 — Assignment #4

Due at ***start*** *of class on April 13. Note that this assignment is to be done by each student individually. You may discuss it in general terms with other students, but the work you hand in should be your own.*

In this assignment you will implement and try out Gaussian process regression models, with hyperparameters found by maximizing the probability of the training data.

You should try two models. In the "additive" model, the covariance function should have the following form:

$$C(x_{i_1}, x_{i_2}) \;=\; 10^2 \;+\; \eta^2 \sum_{j=1}^{p} \Big[\exp\big(-\, ((x_{i_1,j} - x_{i_2,j})/\rho_j)^2 \,\big) \Big] \;+\; \delta_{i_1,i_2}\, \sigma_\epsilon$$

in the non-additive model, the form of the covariance function should be as follows:

$$C(x_{i_1}, x_{i_2}) \;=\; 10^2 \;+\; \eta^2 \exp\Big(-\sum_{j=1}^{p} ((x_{i_1,j} - x_{i_2,j})/\rho_j)^2 \Big) \;+\; \delta_{i_1,i_2}\, \sigma_\epsilon$$

Here, p is the number of input variables, and x_{i_1} and x_{i_2} are the values of the inputs for two cases. $\delta_{a,b}$ is one if $a = b$ and zero otherwise.

In both forms of the covariance function, the 10^2 term is equivalent to an intercept in a linear model, allowing the function to be shifted up or down (from zero) by an amount for which we assign a prior distribution with standard deviation 10. The hyperparameter η controls the overall vertical scale of the function. The ρ_j hyperparameters control the horizontal scales with respect to the input variables. A large value for ρ_j means that changes in input variable j do not have much effect; a small value for ρ_j means that the function changes rapidly as the j'th input changes. The σ_ϵ hyperparameter represents the standard deviation for the noise (residuals) in the model.

Given a set of N training cases, in which you know both the inputs x_i for each case i, you can compute the covariance matrix, $\mathbf{C}$, of the responses, y_i, for these training cases, provided you know what values of the hyperparameters to use. You can then use this covariance matrix, and the actual responses in the training cases, to predict the response in a test cases. For this assignment, your prediction should just be the mean of the distribution for the response in the test case give the responses in the training cases (ie, you don't need to compute its variance).

You should find suitable values for the hyperparameters (σ_ϵ, η, and the ρ_j) by maximizing the probability of the training data, as explained in the lecture slides. This has to be done numerically, using the `nlm` function in R, or the `fminsearch` function in Matlab. To avoid overflow/underflow problems, you should actually minimize minus the log of the probability of the training data, which is obviously inversely related to the probability itself. Another potential problem is that the hyperparameters must be positive — negative values would make no sense — and very small positive values might also produce problems. To ensure this, rather than minimize with respect to a hyperparameter α, you should minimize with respect to $\gamma = \sqrt{\alpha - 0.01}$. You translate back to $\alpha = \gamma^2 + 0.01$ to actually use the hyperparameter.

You should write the following functions (and perhaps others if you find it useful):

```
gp.predict (x.train, y.train, x.test, covf, hypers)
```

This takes as arguments a matrix of training inputs, a vector of training responses, a matrix of test inputs, a covariance function, and a vector hyperparameter values. It returns a vector of mean predictions for responses in test cases. The covariance function passed should take three arguments: two input vectors and a vector of values for the hyperparameters, in the order σ_ϵ, η, and the ρ_j. It should ignore σ_ϵ, returning the covariance between cases with those inputs, excluding the final term involving σ_ϵ. (This term needs to be added in to the diagonal of the matrix $\mathbf{C}$, but it must not be added to the covariance between a training and test case.)

```
gp.log.prob (x.train, y.train, covf, hypers)
```

This takes arguments like `gp.predict`, except that no test cases are involved. It returns the log of the probability of the training data according to the covariance function and hyperparameter values passed.

```
gp.find.hypers (x.train, y.train, covf, initial.hypers)
```

This takes as arguments the training inputs, the training responses, the covariance function (as for `gp.predict`), and a vector of initial values for hyperparameters. It should use the built-in R function `nlm` to find the hyperparameters that maximize the log of the probability of the training data, starting its search with the initial values passed. It returns the vector of hyperparameters found.

I'll put some hints on how to do these things in R on the web page soon.

You should try out your functions on two artificial datasets, both of which have two input variables. Both datasets are on the course web page. The first is the same as for Assignment 3, and is available as before under the entry for Assignment 3 on the web page. The second dataset has the same number of training and test cases, and the same number of inputs, and is available under the entry for Assignment 4 on the web page.

You should do separate experiments on the two datasets. For each dataset, you should make predictions for the test cases based on the training data, using both the additive and the non-additive forms of the covariance function. For both forms of covariance function, you should set the hyperparameters to maximize the probability of the training data. For both forms of covariance function, you should report the average squared error of your predictions for the test cases, and also the log probability of the training data with the hyperparameter values you found.

The web page also has input values for a 51×51 grid of points (the same for the two datasets). You should make predictions for these points as well, using the two forms of covariance function, with the hyperparameters you found using the training cases, and produce contour plots of these predictions.

You should discuss what your results indicate about the properties of these datasets, and compare the results you obtained with the Gaussian process models to the results you obtained with the additive linear spline model in Assignment 3.