Lab 1: Nonparametric and quantile regression

In this lab you will need a data object which you can load using the command:

```
load(url("http://www.stats.gla.ac.uk/~claire/aptslab1.RData"))
```

This handout can be downloaded from: `http://www.stats.gla.ac.uk/~claire/aptslab1.pdf`

Data and R objects

Use `ls()` in `R` to explore the objects available in this `RData` file.

The available datasets (some of which we have discussed in the lectures):

Divorces in the US

This data set (called `divorces`) contains the number of divorces per 10,000 women (divorce) per year for most of the 20th century.

Radiocarbon dating.

This is the data (called `radiocarbon`) used in the lectures with true calendar age (cal.age) and radiocarbon dating predicted age (rc.age), in 1000s.

Great barrier reef

This is a univariate version of the data (called `gbr`) used in the lectures. Only two columns are retained longitude and the principal component score summarising the fauna catch (score1).

Engel household and expenditure data.

This data set is called `engel` with two columns containing household income (x) and food expenditure (y).

Mammals.

This dataset is called `Mammals` with 4 columns, the weight and speed for mammals and an indicator for "specials" or "hoppers".

Tasks:

1. For the **Divorces** dataset:

   (a) Produce a scatterplot of the data to explore the relationship between calendar year (x) and number of divorces (y).

   (b) Using the `lm` function in `R`, fit a polynomial regression model of appropriate degree to the data to model the relationship between number of divorces and year. If necessary, change the degree of the polynomial.

   Hint: The formula `y~x+I(x^2)` *or* `y~poly(x,2)` *fits a quadratic regression model in R. Commands like* `plot(x,y); lines(x, predict(model))` *can be used to plot the model.*

   (c) Fit the following model in `R` to the data using a regression spline with a cubic B-spline basis, assuming normally distributed errors with mean 0 and variance σ^2:

   $$\text{Model2} : \mathbf{y} = f(\mathbf{x}) + \varepsilon.$$

   This can be fitted using the following commands suitably adjusted for this context:

```
library(splines)
lm(y~bs(x, df=6))
```

   The number of basis functions can be altered by changing the value for `df`.

   Plot the fitted model using commands like `predict(model2)` from part (b). Are you happy with the level of smoothing here? Explore alternative values for the degrees of freedom.

   (d) Use the `ns()` function (within `lm()`) to fit a natural cubic spline instead of a cubic B-spline. A natural cubic spline is linear beyond the boundary knots.

2. (a) Now for the **Great Barrier Reef** data fit a spline model using the truncated power series basis to investigate the relationship between score1 (y) and longitude (x). Remember the design matrix is

   $$\mathbf{B} = \begin{pmatrix} 1 & x_1 & \dots & x_1^{r-1} & (x_1-\kappa_1)_+^r & \dots & (x_1-\kappa_m)_+^r \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & \dots & x_n^{r-1} & (x_n-\kappa_1)_+^r & \dots & (x_n-\kappa_m)_+^r \end{pmatrix}$$

   for a truncated power series basis of degree r with equally-spaced interior knots in $\kappa_1, \dots, \kappa_m$.

   We first need to formulate `B`. To do this you can use the function `tbase(x,n.knots=10, deg=3)` provided (n.knots contains interior and outer knots).

   We can then form `yhat` using:

```
beta <- solve(crossprod(B), t(B)%*%y)
yhat <- B%*%beta
```

   Plot longitude against score1 and add the fitted line from `yhat`.

   (b) Fit a penalised regression spline model, this can be done using the `mgcv` package in `R`, which can be used inside `ggplot2` as shown below

```
library(ggplot2)
p <- ggplot(gbr, aes(longitude, score1))
p + geom_point(colour = ‘‘darkblue’’) +
geom_smooth(method = ‘‘gam’’, formula=y~s(x), colour=‘‘red’’)
```

3. For the **Radiocarbon data**, launch the function:

   ```
   pspline.cartoon(radiocarbon)
   ```

   The function fits a P-spline model (using a difference penalty) to the data and shows the fitted model together with the basis function. Experiment with the degree of the spline and the number of knots (by changing the $\pm$), and the smoothing parameter λ (by moving the slider). (Note: the panel might appear on the desktop tool bar).

4. For the **Engel** data

   (a) Plot the data to explore the relationship between household income (x) and food expenditure (y).

   (b) Re-plot the data adding an OLS regression line and a quantile regression line at the median. For this you will need to load `library(quantreg)` and use the following commands:

   ```
   abline(rq(y~x,tau=.5),col=‘‘blue’’)
   abline(lm(y~x),lty=2,col=‘‘red’’)
   ```

   (c) Lines at additional quantiles can be added by using:

   ```
   taus <- c(.05,.1,.25,.75,.90,.95)
   f <- rq(y ~ x, tau = taus)
   for( i in 1:length(taus)){
   abline(coef(f)[,i],col=‘‘grey’’)
   }
   ```

   (d) If we wanted to see all the distinct quantile regression solutions for this example we could specify a `tau` outside the range [0,1], e.g.

   ```
    z <- rq(y~x,tau=-1)
   ```

   The primal solution is in `{z$sol}`, and the dual solution is in `{z$dsol}`.

   (e) If you want to estimate the conditional quantile function of y at a specific value of x and plot it you can do something like this:

   ‘Poor’ is defined as at the .1th quantile of the sample distribution.

   ```
   x.poor <- quantile(x,.1)
   ps <- z$sol[1,]
   qs.poor <- c(c(1,x.poor)%*%z$sol[4:5,])
   plot(ps,qs.poor,type=”n”,
   xlab=expression(tau),ylab=”quantile”)
   plot(stepfun(ps,c(qs.poor[1],qs.poor)),do.points=FALSE,add=TRUE)
   ```

   Explore the quantiles for the rich i.e. the 0.9th quantile.

   (f) Testing can be done using the `summary(model1)` and the `anova(model1,model2)` commands as usual. Fit two quantile regression models at different quantiles, look at their model summaries and compare them formally i.e. using

   `fit.25 <- rq(y~x,tau=.25)` to fit a particular quantile.

   (g) The QR testing has revealed a strong tendency for the dispersion of food expenditure to increase with household income. This is a particularly common form of heteroscedasticity. One common remedy for symptoms like this would be to reformulate the model in log linear terms. Repeat part (b) taking a log10 transform for y and x.

5. The **Mammals** data give the maximum running speed and body weight for a sample of 107 terrestrial mammals. Two groups are of particular interest, "hoppers", such as the kangaroo, and "specials" such as the sloth and the hippopotamus whose lifestyles do not feature speed as an important factor.

   (a) For the `Mammals` dataset explore the use of the function `rqss`, in the package `quantreg`, which enables nonparametric quantile regression fitting with a total variation roughness penalty. For these data estimate a model for the conditional median ($\tau = .5$) of `log(speed)` as a function of `log(weight)`. Note that the default value of the penalty is $\lambda = 1$ - - you can also experiment with different values.

```
x <- log(weight)
y <- log(speed)
plot(x,y, xlab=‘‘Weight in log(Kg)’’, ylab=‘‘Speed in log(Km/hour)’’,type=‘‘n’’)
points(x[hoppers],y[hoppers],pch = ‘‘h’’, col=‘‘red’’)
points(x[specials],y[specials],pch = ‘‘s’’, col=‘‘blue’’)
others <- (!hoppers & !specials)
points(x[others],y[others], col=‘‘black’’,cex = .75)
fit <- rqss(y ~ qss(x, lambda = 1),tau = .5)
plot(fit, add = TRUE)
```

   (b) Repeat the analysis for $\tau = .9$.

   (c) Now fit a model to the data excluding "specials". Notice that the quantile fit is robust to these outlying observations. Also fit regression spline models for the mean with and without "specials" and compare the effect of these outliers on the fit.

6. Explore the following code for plotting quantiles using `ggplot`. An example is given using the `Mammals` data:

```
library(ggplot2)
m <- ggplot(Mammals, aes(x=weight, y=speed)) + geom_point() +
                   scale_x_log10() + scale_y_log10()
m + geom_quantile()
m + geom_quantile(quantiles = 0.5)
q10 <- seq(0.05, 0.95, by = 0.2)
m + geom_quantile(quantiles = q10)
```

   You can also use `rqss` to fit smooth quantiles. You'd need to provide a value for `lambda` yourself (default is `lambda=1`).

```
m + geom_quantile(method = ‘‘rqss’’)
```

If you would like to check your answers or would prefer to work through a preprepared script, then this is available from:

http://www.stats.gla.ac.uk/~claire/aptslab1.R