Series 10

1. Consider again the ozone dataset. Now we focus on **projection pursuit regression (PPR)**. Take the log of the response **uop3** and remove the outlier. It is also useful to standardize the predictors: The α-matrix is much more easily interpretable because your explanatory variables are now on the same scale. This is achieved by the following R-code:

```
data(ozone, package = "gss")
d.ozone <- subset(transform(ozone, logupo3 = log(upo3)), select = -upo3)
d.ozone.e <- d.ozone[-which.max(d.ozone[,"wdsp"]),]
d.ozone.es <- d.ozone.e
d.ozone.es[,-10] <- scale(d.ozone.e[,-10])
```

Explain why in a few short sentences.

We want to compare models with different number of ridge functions and different smoothers. Use leave-one-out cross-validation to compare models.

a) You can choose three different smoothers (argument `sm.method`). The default is `supsmu`, Friedmans "super smoother". Other possibilities are `spline` which uses splines with a specified (equivalent) degree of freedom for each ridge function and `gcvspline` which chooses the smoothness by GCV.
For each of these smoothers vary the numbers of ridge functions (`nterms`). If your computer power allows, you can also try different degrees of freedom (`df`) for `spline`.
You may also want to use `max.terms` to get better results. The following description is taken from the details paragraph of the help-file of `ppr` (i.e. `?ppr`):

The algorithm first adds up to 'max.terms' ridge terms one at a time; it will use less if it is unable to find a term to add that makes sufficient difference. It then removes the least "important" term at each step until 'nterms' terms are left.

Compare your PPR-model to MARS with an interaction degree of 1,2 and 3. You can use the cv-function: `cv(earth, degree = ?)`.
R-Hints: The following function is a general function for cross validation. You can use it in this excercise, but also later on for other excercises or your own projects. Therefore, it might be worth trying to understand the functions `model.frame` and `model.response`. Look at the help pages of these functions.

```
cv <- function(fitfn, formula = logupo3 ~ . , data = d.ozone.es, ...,
               trace = TRUE)
 {
   modFrame <- model.frame(formula, data = data)
   nc <- nrow(data)
   ssr <- 0
   if(trace) cat("  j = ")
   for(j in 1:nc) {
     if(trace) cat(if(j %% (nc %/% 10) == 1) paste(j, "") else ".")
     ## Fit without 'j' :
     fit <- fitfn(formula=formula, data = data[-j ,], ...)
     ## Evaluate at 'j' :
     ssr <- ssr + (model.response(modFrame)[j] - predict(fit, modFrame[j,]))^2
   }
```

```
  if(trace) cat("\n")
  ssr
}
```

Remarks:

1. The first argument is a **function**. "`...`" are multiple arguments which are passed to `fitfn`.
2. The default values for formula and data in `cv()` are specified to work with the transformed and standardized ozone dataset `d.ozone.es`:
3. Example for a call:

```
cv.gcv.2 <- cv(ppr, sm.method="gcvspline", nterms = 2, max.terms = 5)
```

You can get the code of the function `cv` and the dataframe `d.ozone.es` with:

```
source("http://stat.ethz.ch/teaching/lectures/FS_2010/CompStat/cv-dozonees.R").
```

b) Choose the best model from a) and visualize the ridge functions as in the lecture notes on page 71.

c) Interpret the α-matrix (`$alpha`) for your model. (What does for example a big value of an element of α mean?)
You can use `round()` to get a better overview. If you have another model which performs nearly as well as the best but has a much nicer interpretation you may want to prefer it to the best model.

Preliminary discussion: Friday, May 20.

Deadline: Friday, May 27.