Exercise 4

1. The dataset `bmw` is a time series of log returns of the BMW stock (business-daily, between June 1986 and March 1990). The log return is defined as follows:

$$X_t = \log\left(\frac{P_t}{P_{t-1}}\right),$$

where P_t is the stock price at time t. Log returns can be modelled by

$$X_t = \sigma_t \epsilon_t, \text{ where } \mathbf{E}[\epsilon_t] = 0, \mathrm{Var}(\epsilon_t) = 1, \tag{1}$$

ϵ_t independent of $\{X_s;\ s < t\}$, $\sigma_t^2 = v(X_{t-1})$, where $v: I\!R \mapsto I\!R^+$ is the so-called "volatility function". Thus, X_t depends on $\{X_s;\ s < t\}$ only through X_{t-1} (Markov-property).

The model can be fitted by nonparametric regression of the function v in

$$Y_t = X_t^2 = v(X_{t-1}) + \eta_t, \text{ where } \eta_t = \sigma_t^2(\epsilon_t^2 - 1)$$

is treated as error term. The data can be downloaded at

`http://stat.ethz.ch/Teaching/Datasets/bmw.dat`

to a file with name `bmw.dat`, say, and read into **R** by

```
bmwlr <- scan("bmw.dat")
```

`bmwlr` should be a vector of 1000 observations.

a) Compute $\mathbf{E}[X_t|X_{t-1}, X_{t-2}, \ldots]$, $\mathrm{Var}(X_t|X_{t-1}, X_{t-2}, \ldots)$, $\mathrm{Cov}(X_t, X_{t-h})$, $h > 0$.
Background about conditional expectations:
For two (possibly multi-dimensional) random variables X and Y, the conditional distribution $P^{Y|X=x}$ can be uniquely defined for $P-$almost all values of X. Define $h(x) = \mathbf{E}[Y|X = x]$ as the expectation of Y under the conditional distribution $P^{Y|X=x}$. For the random variable X, $h(X) = \mathbf{E}[Y|X]$ is a random variable. Here is a very useful equation (the so-called tower property), which will be needed for parts a and b:

$$\mathbf{E}[Y] = \mathbf{E}[\mathbf{E}[Y|X]], \tag{2}$$

the outer expectation taken over the distribution of X. Conditional variances and covariances are defined analogously.

b) Show $\mathbf{E}[\eta_t] = 0$.
Note: Other usual model assumptions on errors, such as independence, are not fulfilled by η_t, but with some effort (don't try!) it can be shown that v can be optimally estimated by the same estimation methods as if the η_t would be independent errors.

c) Model (1) is often chosen for this kind of data because it leads to observations that are not autocorrelated[1] (as shown in part a), but dependent. Dependency can be verified by showing that under the model $\text{Cov}\left(X_t^2, X_{t-h}^2\right) \neq 0,\ h > 0$ (complicated). Plot and interpret the autocorrelation functions of X_t and X_t^2 for the BMW-dataset.

R-hint: Function `acf`. For example, "autocorrelation of lag 1" (in the plot, with 1000 observations, indicated as lag 1 out of 999) means correlation between X_t and X_{t-1}. The plot shows also an acceptance region (at 5%-significance level) for testing the nullhypothesis of uncorrelated observations.

d) Fit the data by a suitable nonparametric regression of the regression function v. Check model assumptions. If you like, comment on the results and/or compare more than one fit.

R-hint: There are numerous functions to perform nonparametric regression. You may choose one or more from the following list. Consider the help-pages for details.

```
locpoly   library(KernSmooth)  local polynomial
dpill     library(KernSmooth)  optimal bandwith estimation
ksmooth   library(stats)       kernel regression
loess     library(stats)       local polynomial
lokerns   library(lokern)      kernel regression with local optimal bandwith
```

Note that for computing residuals it is necessary to know the fitted values at the data points. These can be provided by `ksmooth` (via parameter `x.points`) and `loess` (via `predict`). For the other methods, a very dense grid has to be chosen and the nearest value in the grid could be used.

Warning: The structure in the data is pretty unclear. This is not a very didactic example.

2. Consider again the `bmw`-dataset of exercise 1 and the model

$$Y_t = X_t^2 = v(X_{t-1}) + \eta_t.$$

Because the structure in the dataset is pretty unclear, we might want to know if a complicated nonparametric regression gives us valuable information and which one is the best. The following fits should be compared:

1. the kernel regression fit from `ksmooth`,
2. the `loess`-fit,
3. a smoothing spline fit as obtained by function `smooth.spline` (all from package `stats`), where you have chosen a fixed value for the parameter `df` in advance,
4. a smoothing spline fit as obtained by function `smooth.spline` with the smoothing parameter selected automatically by cross-validation (consider the "Details"-section of `help(smooth.spline)` and the description of parameter `cv` and value `cv.crit`).
5. a constant "fit" by the overall mean of Y_t, simply ignoring the X_{t-1}-values.

a) Compare the methods by leave-one-out cross-validation (note that this is included in function `smooth.spline`, but not in the others). You may also compare the value for `smooth.spline` with your own computed cross-validation value and the GCV-value, which is also included in function `smooth.spline`.

b) The comparison of the CV-value of the method no. 4 with the others is not fair. Can you explain the problem?

[1] "Autocorrelated" refers to correlation over time, i.e., correlation between X_t and $X_{t-h},\ h > 0$.

c) Because of the time series structure of the data, a "naive" leave-one-out cross-validation may not be valid. A safer leave-one-out approach is as follows: If the leave-one-out loss function is evaluated for a data point (X_t^2, X_{t-1}), not only (X_t^2, X_{t-1}) is excluded from the computation of the fit, but also (X_{t+1}^2, X_t) and (X_{t-1}^2, X_{t-2}). Repeat part a) with the safer approach and compare.

R-hints: You may begin as follows:

```
bmwlr <- scan("bmw.dat")
library(stats)
library(KernSmooth)
x <- bmwlr[1:999]                  # last values of xt cannot be used for x
y <- bmwlr[2:1000]^2               # first value of xt cannot be used for y
reg <- data.frame(x=x,y=y)
```

Then, `cdat <- reg[-i,]` is `reg` without point i, and `cdat <- reg[-c((i-1),i,(i+1)),]` is `reg` without the points $i-1, i, i+1$. You may run into trouble because some functions, e.g., `ksmooth`, `dpill` and `loess` yield sometimes values `NA` ("missing") or `NaN` ("not a number"). This means that the computations did not work properly because, e.g., a point outside the data range was to be predicted (this happens if an extreme point has been left out for the cross-validation). The function `is.na` tests if a value is `NA` or `NaN` and can be used to predict in these cases the mean of the `y`-values (or something else) instead.

Note that the parameter `newdata` of `predict.loess` has to be a `data.frame`. How you make a data frame out of a single point: `newdata=reg[i,"x",drop=FALSE]`.

The computation of leave-one-out CV may take some time.

3. The leave-one-out CV-score can be written in such a way that it depends only on the estimator $\hat{m}(\cdot)$ which is computed from the *full* dataset. To obtain the CV-score, it is therefore not necessary to calculate the leave-one-out estimators $\hat{m}_{n-1}^{(-i)}(\cdot)$. From the manuscript we learn:

$$n^{-1}\sum_{i=1}^{n}\left(Y_i - \hat{m}_{n-1}^{(-i)}(X_i)\right)^2 = n^{-1}\sum_{i=1}^{n}\left(\frac{Y_i - \hat{m}(X_i)}{1 - S_{ii}}\right)^2,$$

where S is the *hat-matrix* of the linear estimator $\hat{m}(\cdot)$. In this exercise we are going to prove this formula step by step in the case of multiple-linear-regression $y_i = \mathbf{x_i^T}\theta + \epsilon_i$.

a) Show that for an invertible $p \times p$-matrix A and two p-vectors $\mathbf{a}$ and $\mathbf{b}$ with $\mathbf{b^T}A^{-1}\mathbf{a} \neq 0$ the matrix $A - \mathbf{ab^T}$ is invertible too and that its inverse can be computed as follows:

$$(A - \mathbf{ab^T})^{-1} = A^{-1} + \frac{1}{1 - \mathbf{b^T}A^{-1}\mathbf{a}} \cdot A^{-1}\mathbf{ab^T}A^{-1}.$$

b) Show the following formula which describes the influence of omitting the i.th observation for the multiple-linear-regression estimator:

$$\hat{\theta}^{(-\mathbf{i})} - \hat{\theta} = -\frac{y_i - \mathbf{x_i^T}\hat{\theta}}{1 - S_{ii}}(X^T X)^{-1}\mathbf{x_i}.$$

Hints: Let $A := X^T X = \sum_{i=1}^{n} \mathbf{x_i x_i^T}$, $\mathbf{c} := X^T y = \sum_{i=1}^{n} y_i \mathbf{x_i}$.
Now you might start as follows: $\hat{\theta}^{(-\mathbf{i})} = (A - \mathbf{x_i x_i^T})^{-1}(\mathbf{c} - y_i\mathbf{x_i})$, then use **a)**.

c) From **b)** you can finally conclude the desired result:

$$y_i - \mathbf{x_i^T}\hat{\theta}^{(-\mathbf{i})} = \frac{1}{1 - S_{ii}}(y_i - \mathbf{x_i^T}\hat{\theta}).$$

Deadline: Friday, Mai 13, 2005, at the beginning of the lecture.

Advice: Thursdays from 12.00-13.00, LEO C12.1, Leonhardstr. 27. Or contact either Maik Berchtold, `berchtld@stat.math.ethz.ch`, or Lukas Meier, `meier@stat.math.ethz.ch`.