Series 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 $v : \mathbb{R} \mapsto \mathbb{R}^+$ 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.

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 were 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, $\mathrm{Cov}\left(X_t^2, X_{t-h}^2\right) \neq 0,\ h > 0$ (complicated). Plot and interpret the autocorrelation functions of X_t and X_t^2 for the BMW-dataset.
The data can be read into R by

```
> bmwlr <- scan("http://stat.ethz.ch/Teaching/Datasets/bmw.dat")
```

`bmwlr` should be a vector of 1000 observations.
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 null hypothesis of uncorrelated observations.

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

d) Fit the data using the nonparametric regression methods Nadaraya-Watson, local polynomial and smoothing splines for the regression function v.

Comment on the results and compare the fits obtained using the mentioned nonparametric estimators.

R-hint: There are numerous functions to perform nonparametric regression and here are some of them. Consider the help-pages for details.

Function	**Library**	**Description**
`loess`	`stats`	local polynomial
`smooth.spline`	`stats`	smoothing splines
`ksmooth`	`stats`	Nadaraya-Watson kernel regression
`glkerns`	`lokern`	kernel regression with global optimal bandwidth
`lokerns`	`lokern`	kernel regression with local optimal bandwith

Start with `bmwloess <- loess(y~x)`. This yields an estimated degree of freedom (`bmwloess$trace.hat`). Use this degree of freedom for the `smooth.spline(x,y,df=...)` function. Theoretically it is possible to adaptively choose the bandwidth for the `ksmooth` function, so that the trace of the hat matrix (=df) equals the degrees of freedom calculated above (see Series 3). As this will take up too much time on slow machines, we provide you with the result. The corresponding bandwidth for `ksmooth` is 3.53.

Check model assumptions, but don't spend too much time on this since the structure of the data is pretty unclear. Note that for computing residuals it is necessary to know the fitted values at the data points. For `ksmooth` they are provided via argument `x.points` and for `loess` and `smooth.spline` via `fitted()`.

e) Fit the data using the functions `glkerns` and `lokerns`. Compare the fits and check the model assumptions as in part d). In addition plot the local bandwidths from `lokerns` and compare them to the global bandwidth of the function `glkerns`.

Preliminary discussion: Friday, March 23.

Deadline: Friday, March 30.