Time Series Practical 1 **Hilary Term 2010** Prof. Reinert

In this practical we shall have a go at the `lh` time series from lectures. The series is compiled of 48 observations at 10-minute intervals on luteinizing hormone levels for a human female. You need the package `library(MASS)`.

Look at the data: type `lh` and plot the data: `plot(lh)` or `ts.plot(lh)`. In the data plot you can check how the sample mean compares to the time series by using for example

```
lh.mean<-mean(lh)
abline(lh.mean, 0, col="red")
```

Does the series have a trend, is it stationary? We can plot the running means and running variances using

```
RunningMean<-NULL
RunningVariance<-NULL
for (Dummy in 1:length(lh))
{ RunningMeanDummy<-mean(lh[1:Dummy])
RunningMean<-c(RunningMean,RunningMeanDummy)
RunningVarianceDummy<-var(lh[1:Dummy])
RunningVariance<-c(RunningVariance,RunningVarianceDummy) }
par(mfrow=c(1,2))
plot.ts(RunningMean)
plot.ts(RunningVariance)
```

Does the series look stationary? One way to compute the linear trend of a series is via `lm`, which fits a linear model; try `lh.fit<- lm(lh ~ c(1:48))`. You can plot the trend via `lines(fitted(lm(lh ~ c(1:48))))`. What do you think about the fit?

Remove the linear trend from the time series. For example, you could use

```
Intercept<-lh.fit$coef[1]
Slope<-lh.fit$coef[2]
lh.fit.detrend<-lh.fit$residual
plot.ts(lh.fit.detrend)
```

Does this series look stationary? You can try the running mean and the running standard deviation again. You could stabilise the variance by taking logs, or you could difference the series, using `diff`.

There is a command called `stl` which decomposes the time series into a trend, a seasonal component, and a remainder term. For this command to

work we need to introduce a period in the time series. Here a natural choice is 6, as 6 times 10 min gives 60 min. For example you could use
`lh.Frequency.6<-ts(as.vector(lh.diff),start=1,,frequency=6)`

To perform the decomposition, try
`lh.Frequency.6.stl<-stl(lh.Frequency.6,s.window="periodic")`

Then plot using `plot(lh.Frequency.6.stl)`. The output of the decomposition are three series; we would like to model the remainder series, `lh.rem<-lh.Frequency.6.stl$time.series[,3]`.

Estimate the autocorrelation function using `acf`, and estimate the partial autocorrelation function, using `pacf`. Use the plots to suggest some AR, or MA, or perhaps ARIMA models to fit the data.

Use `arima` to fit ARIMA models to the data. The arima function by default finds starting values by conditional sum of squares and then uses maximum likelihood for the fit. For example, `arima(x=lh.rem, order=c(1,0,1))` fits an ARMA(1,1) model to the data. The option `n.cond.` specifies on how many observations is being conditioned on How do you use the AIC for model selection?

If you wanted to write a loop for fitting arima models, you could try

```
TS<-lh.rem
ARIMA.AIC<-matrix(,5,5)
Dummy1List<-NULL
Dummy2List<-NULL
for (Dummy1 in 1:5){
for (Dummy2 in 1:5){
Dummy1List<-c(Dummy1List,Dummy1)
Dummy2List<-c(Dummy2List,Dummy2)
ARIMA.AIC[Dummy1, Dummy2]<-arima(TS, order=c(Dummy1,0,Dummy2))$aic
}
ARIMA.AIC }
```

What is your best fit? Plot time series diagnostics using `tsdiag`. For example, if `fit 1<- arima(x=lh.rem, order=c(1,0,1))` then `tsdiag(fit1)` gives diagnostic plots. You can isolate the remainder of the fit using `fit1.res<-fit1$residuals`

Is this series consistent with white noise? You can use `cpgram(fit2.res)` for a cumulative periodogram.