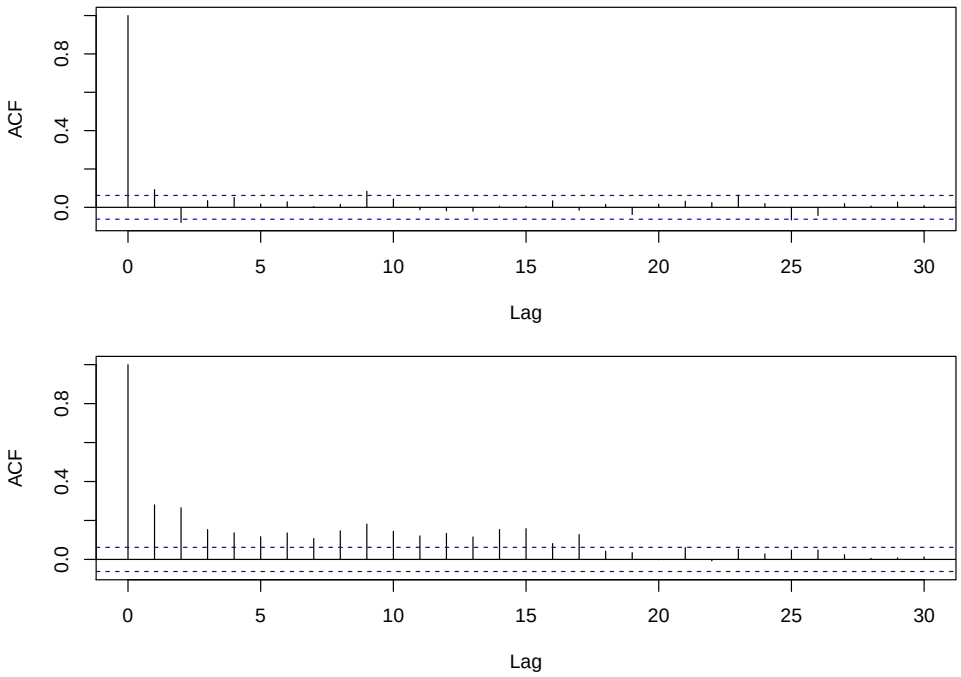
Solution to Series 4

1. **a)** The ACF plot shows that autocorrelation among X_ts is almost inexistent (but this does not imply that they are independent). These findings support our theoretical derivations from part **a)**, namely that $\mathrm{Cov}\,(X_t, X_{t-h}) = 0$. Autocorrelation among X_t^2s is of higher order.

```
> bmwlr <- scan("http://stat.ethz.ch/Teaching/Datasets/bmw.dat")
> par(mfrow = c(2,1))
> acf(bmwlr)
> acf(bmwlr^2)
```



b) First, we sort the values, in order not to get problems with `ksmooth`, which orders the values internally and gives back results corresponding to the ordered values.

```
> x  <- bmwlr[-length(bmwlr)]    ## last value of xt cannot be used for x
> ox <- order(x)
> x  <- x[ox]
> y  <- bmwlr[-1]^2 ## first value of xt cannot be used for y
> y  <- y[ox]
```

Then, we perform a local polynomial regression with degrees of freedom = `dgf` (automatically calculated). Note that the automatic choice of the degrees of freedom in the function `loess` is somewhat arbitrary: the method uses `span = 0.75` as default smoothing parameter. A better way to choose an appropriate smoothing parameter would be to use cross validation (see next exercise series); however, the data in this example is so poorly structured that this approach doesn't help here either.

```
> bmwloess <- loess(y ~ x)
> (dgf     <- bmwloess$trace.hat)
```

```
[1] 6.870869
```

Next, we fit a smoothing spline with the same degrees of freedom.

```
> bmwss <- smooth.spline(x,y,df = dgf)
> n <- length(x)
```

Next we choose an adaptive bandwidth h for `ksmooth` such that the trace of the corresponding hat matrix approximately equals `dgf` obtained from the `loess` fit. This may be acheived using the code provided in the exercise sheet, correctly filling in gaps as shown below. The function `delta.dfg` returns the degrees of freedom for the Nadaraya-Watson kernel regression with bandwidth h minus the degrees of freedom dgf that we want to match (i.e. the degrees of freedom provided by the method `loess`). The resulting h equals 3.54.

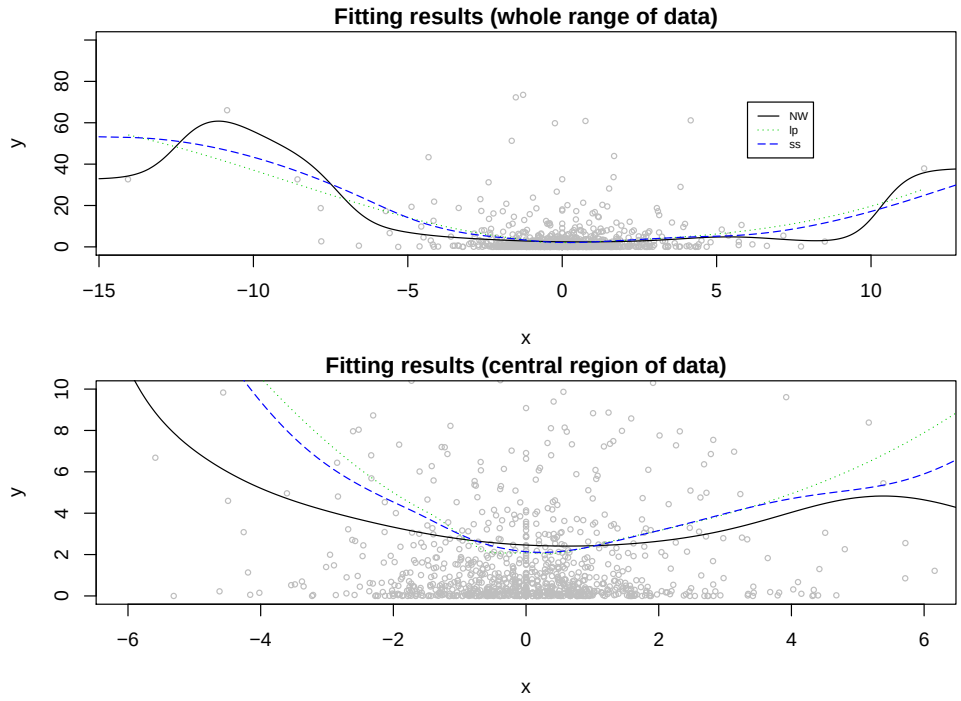
```
> # Function delta.dgf returns the degrees of freedom for the Nadaraya-Watson kernel
> # regression with bandwidth 'h' minus the degrees of freedom 'dgf' to be matched
> delta.dgf <- function (h, x, dgf){
        hatMat(x, trace = TRUE,
        pred.sm = function(x,y,...) ksmooth(sort(x), y, "normal", x.points=x, ...)$y,
        bandwidth = h) - dgf
 }
> bandwidth.nw <- uniroot(delta.dfg, c(3,4), x = x, dgf = 6.870869, tol = 0.01)$root #3.54
```

Fitting a kernel smoother is done in the following. Here, h is chosen as aforementioned. Furthermore, we select an equidistant grid of design points for plotting. In doing so, we are able to investigate the interpolation quality of our smoothers.

```
> h <- 3.54
> bmwks <- ksmooth(x,y,kernel="normal",bandwidth=h,x.points=x)
> x.out <- seq(-15,15,length = n)
> ## NW evaluated at x.out
> bmwksp <- ksmooth(x,y,kernel = "normal",bandwidth=h,x.points = x.out)
```

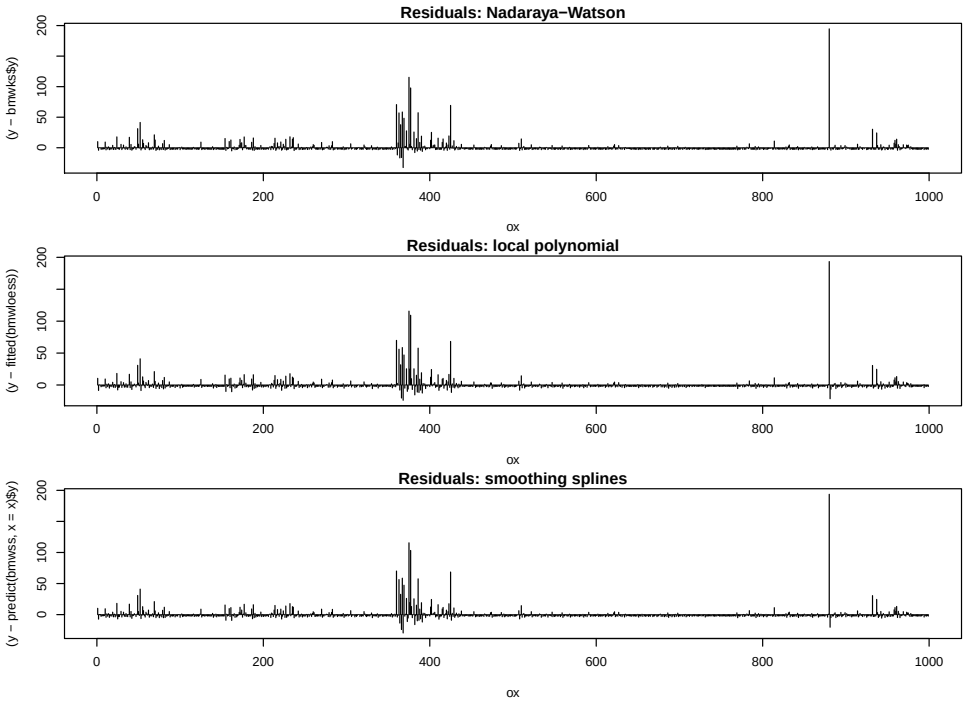
We compare three smoothers taking care that they have approximately the same degrees of freedom. From the figure below no clear functional pattern in the data is visible, although one might conclude that a slight U-shape is noticeable. Therefore, (in absolute values) large returns on a preceding date lead to large volalities the next day. In the extreme areas of `x`, the `ksmooth`-fit is determined by few points, hence the poor fit. `smooth.spline` and `loess` look smoother.

```
> ## plot 1: whole range of data first
> par(mfrow = c(2,1))
> plot(x,y, main="Fitting results (whole range of data)",
      cex = 0.6, col = "gray", ylim = c(0, 100))
> lines(x.out,bmwksp$y,col = 1) ## plot ksmooth fit
> lines(x.out, predict(bmwloess, newdata = data.frame(x = x.out)),
       lty = 3, col = 3, cex = 0.5) ## plot loess fit
> lines(x.out, predict(bmwss, x = x.out)$y, lty = 5,col = 4) ## plot smoothing splines fit
> legend(6,70,legend = c("NW","lp","ss"),
        lty = c(1,3,5),col = c(1,3,4),cex = 0.6)
> ## plot 2: fits for central region of data
> plot(x,y, main="Fitting results (central region of data)", cex = 0.6, col = "gray", xlim = c
> lines(x.out,bmwksp$y,col = 1)
> lines(x.out, predict(bmwloess, newdata = data.frame(x = x.out)),
       lty = 3, col = 3, cex = 0.5)
> lines(x.out, predict(bmwss, x = x.out)$y, lty = 5,col = 4)
```



The residuals vs. time plots for the three smoothers depict a clear violation of the independence assumption of the residuals. The Tukey-Anscombe plots (not shown) exhibit large outliers. Futhermore, the heteroskedacity of the underlying process is noticeable.

```
> par(mfrow = c(3,1))
> plot(ox,(y-bmwks$y), type = "h",main = "Residuals: Nadaraya-Watson")
> plot(ox,(y-fitted(bmwloess)), type = "h",main = "Residuals: local polynomial")
> plot(ox,(y-predict(bmwss,x = x)$y),type = "h",
      main="Residuals: smoothing splines")
```



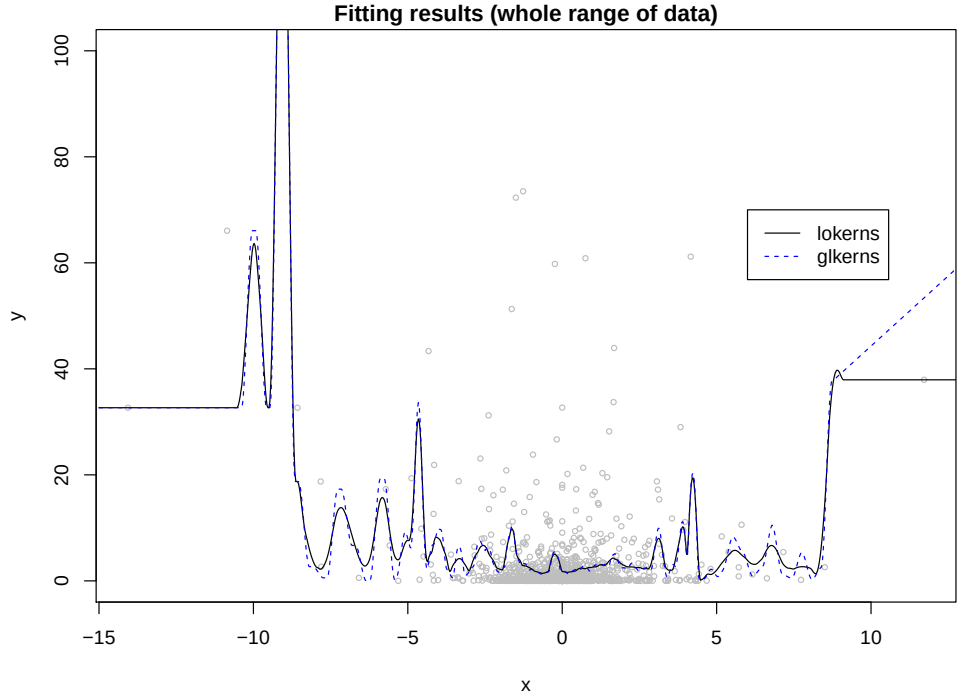
c) Finally, we perform a kernel regression but in contrast to the kernel smoother, a global and local optimal bandwidth is selected, respectively.

```
> library(lokern)
> glk.r <- glkerns(x,y,x.out=x)
> lk.r  <- lokerns(x,y,x.out=x)
```

```
> glk <- glkerns(x,y,x.out=x.out)
> lk  <- lokerns(x,y,x.out=x.out)
```

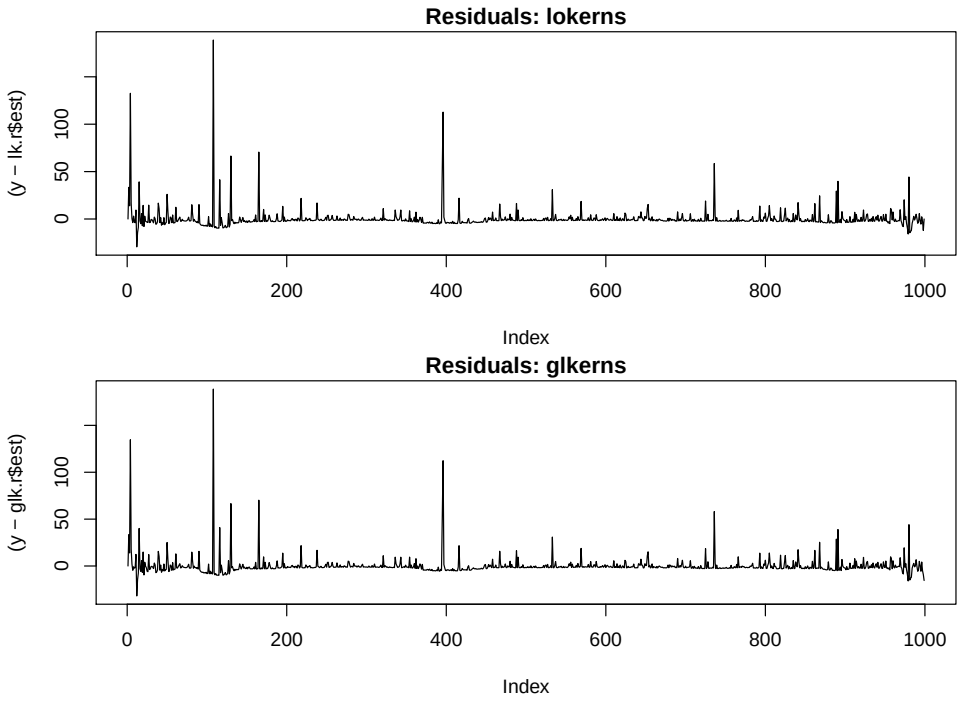
We fit both smoothers (`lokerns` and `glkerns`) to the data. Here, `lokerns` looks a bit smoother. The variability of `glkerns` is slightly higher. At the highest positive x-values `glkerns` shows some artifacts.

```
> plot(x,y, main="Fitting results (whole range of data)",
      cex = 0.6, col = "gray", ylim = c(0, 100))
> lines(x.out,lk$est,col = 1) ## plot lokerns fit
> lines(x.out,glk$est,col = "blue",lty = 2) ## plot glkerns fit
> legend(6,70,legend = c("lokerns","glkerns"),
        lty = c(1,2),col = c(1,"blue"))
```



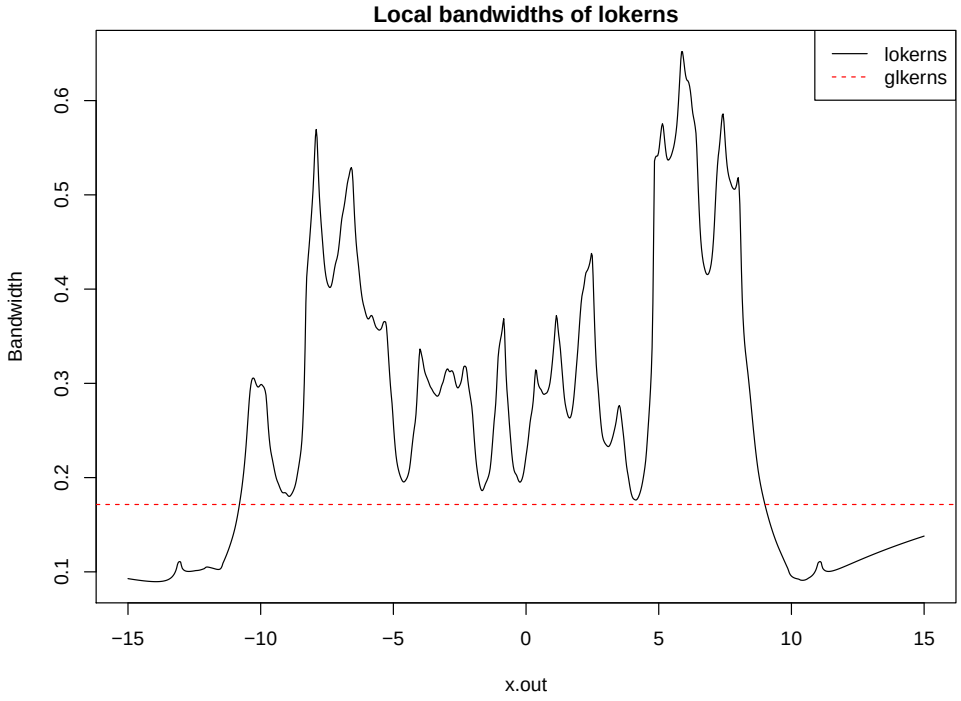
The residuals vs. time plots for the three smoothers depict (again) a clear structure. The indepedence assumption seems to be violated.

```
> par(mfrow = c(2,1))
> plot((y-lk.r$est), type = "l",main = "Residuals: lokerns")
> plot((y-glk.r$est), type = "l",main = "Residuals: glkerns")
```



The last figure illustrates the bandwidth h as a function of x for both `lokerns` and `glkerns`. In regions with many data points the local optimal bandwidth is decreased. However, it is higher than the global optimal h from `glkerns`. Contrarily, in regions where data is sparse, the global optimal h is higher than the local optimal one.

```
> plot(x.out,lk$bandwidth, type = "l",main = "Local bandwidths of lokerns",ylab="Bandwidth")
> abline(h=glk$bandwidth,col=2,lty=2)
> legend("topright",legend = c("lokerns","glkerns"),
        lty = c(1,2),col = c(1,2))
```



d) (optional) Using the facts: ϵ_t independent of $\{X_{t-1}, X_{t-2}, \ldots\}$, and the tower property (and σ_t,

σ_t^2 measurable w.r.t. $\sigma((X_s)_{s<t})$), we get:

$$\begin{aligned}
\mathbf{E}\left[X_t|X_{t-1}, X_{t-2}, \ldots\right] &= \mathbf{E}\left[\sigma_t\epsilon_t|X_{t-1}, X_{t-2}, \ldots\right] \\
&= \sigma_t \mathbf{E}\left[\epsilon_t|X_{t-1}, X_{t-2}, \ldots\right] \\
&= \sigma_t \mathbf{E}\left[\epsilon_t\right] = 0. \\
\mathrm{Var}\left(X_t|X_{t-1}, X_{t-2}, \ldots\right) &= \mathbf{E}\left[X_t^2|X_{t-1}, X_{t-2}, \ldots\right] \\
&= \mathbf{E}\left[\sigma_t^2\epsilon_t^2|X_{t-1}, X_{t-2}, \ldots\right] \\
&= \sigma_t^2 \mathbf{E}\left[\epsilon_t^2|X_{t-1}, X_{t-2}, \ldots\right] \\
&= \sigma_t^2 \mathbf{E}\left[\epsilon_t^2\right] = \sigma_t^2. \\
\mathrm{Cov}\left(X_t, X_{t-h}\right) &= \mathbf{E}\left[X_t X_{t-h}\right] = \mathbf{E}\left[\mathbf{E}\left[X_t X_{t-h}|X_{t-1}, X_{t-2}, \ldots\right]\right] \\
&= \mathbf{E}\left[X_{t-h} \mathbf{E}\left[X_t|X_{t-1}, X_{t-2}, \ldots\right]\right] = 0.
\end{aligned}$$

e) (optional) Using the tower property:

$$\begin{aligned}
\mathbf{E}\left[\eta_t\right] &= \mathbf{E}\left[\mathbf{E}\left[\eta_t|X_{t-1}, X_{t-2}, \ldots\right]\right] \\
&= \mathbf{E}\left[\mathbf{E}\left[\sigma_t^2(\epsilon_t^2 - 1)|X_{t-1}, X_{t-2}, \ldots\right]\right] \\
&= \mathbf{E}\left[\sigma_t^2 \mathbf{E}\left[\epsilon_t^2 - 1|X_{t-1}, X_{t-2}, \ldots\right]\right] \\
&= \mathbf{E}\left[\sigma_t^2 \mathbf{E}\left[\epsilon_t^2 - 1\right]\right] = 0.
\end{aligned}$$