Stat Econ 2 Second

Victor Z. Nygaard

Last compiled on 28. oktober, 2021

Stat Econ 2 Assignment 2

We may start off by setting seed:

```
set.seed(314)
```

Exercise 1

a)

With $\hat{X}_{n+1} = E(X_{n+1}|X_n) = \phi X_n$ being the best approximation of X_{n+1} in the sense mentioned, the surrogate predictor $\tilde{X}_{n+1} = \hat{\phi} X_n$ will necessarily be worse, as $\hat{\phi}$ will in and of itself is fallible to errors in predicting ϕ. It will thus not be the best approximation.

b)

We will initialize, assuming a central student t distribution, and simulate the AR(1) model, with slope 0.9

```
n <- 1000
X <- 0:1000
Z <- rt(n+1,10)
for (i in 2:1001){
  X[i] <- 0.9*X[i-1]+Z[i]
}
```

With this we may start simulation of the YW estimates for later implementation in simulating the surrigate pridictor process, for this we reimplement functions build for the first assignment in StatØ2:

```
gammaf <- function(X,h) {
   n <- length(X)
   gamt <- 0
   for (t in 1:(n-h)) {
      tempt <- (X[t]-mean(X))*(X[t+h]-mean(X))
      gamt <- gamt + tempt
   }
   1/n*gamt
}
```

Such that for

$$\hat{\phi}_n = \frac{\gamma_{n,X}(1)}{\gamma_{n,X}(0)} \equiv \rho_{n,X}(1)$$

we have

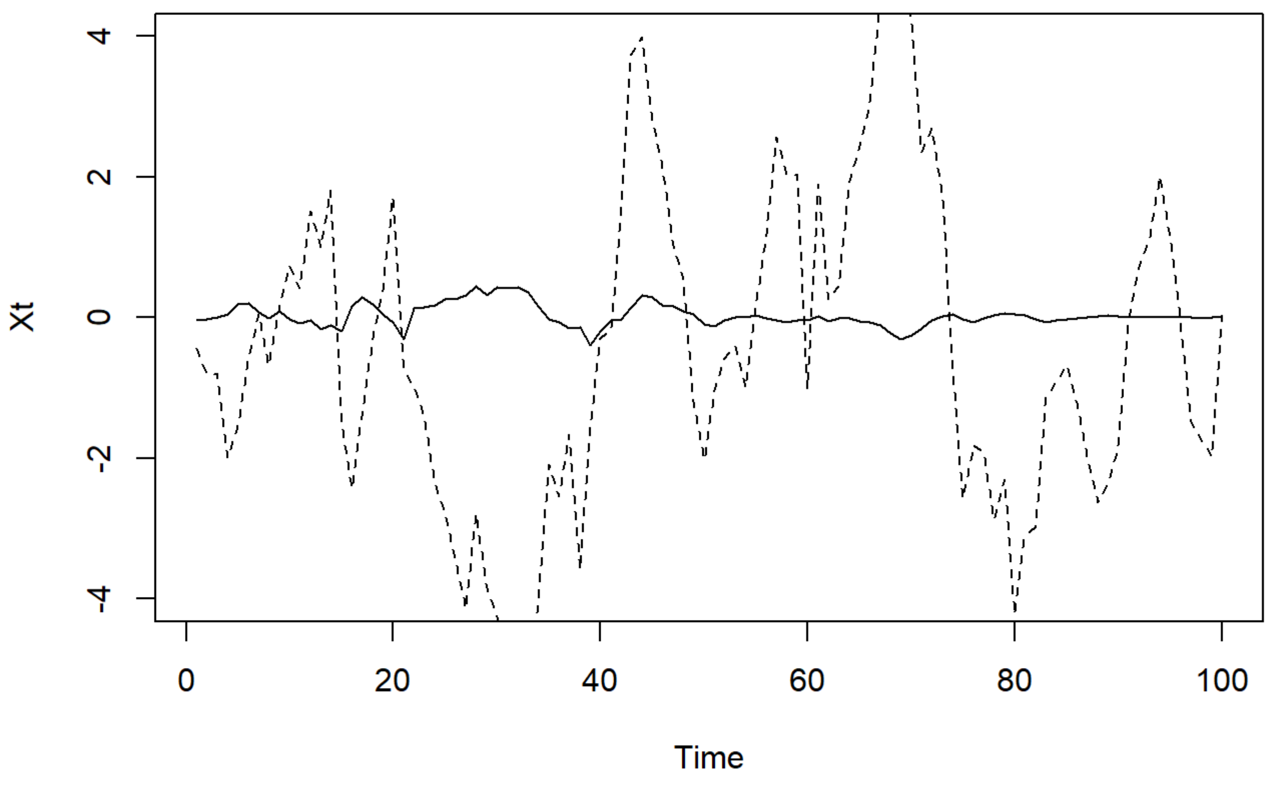
```r
phih <- 1:1000
for (i in phih) {
  phih[i] <- gammaf(X,i)
}
```

Calculating the surrogate predictor for indices 901 through 1000:

```r
Xt <- 901:1000
for (k in 901:1000) {
  Xt[k-900] <- phih[k-1]*X[k-1]
}
```

and plotting the `Xt` as the solid line, together with the dashed line of `X[901:1000]`:

```r
ts.plot(Xt, ylim = c(-4,4));lines(X[901:1000], lty = 2)
```



Exercise 2

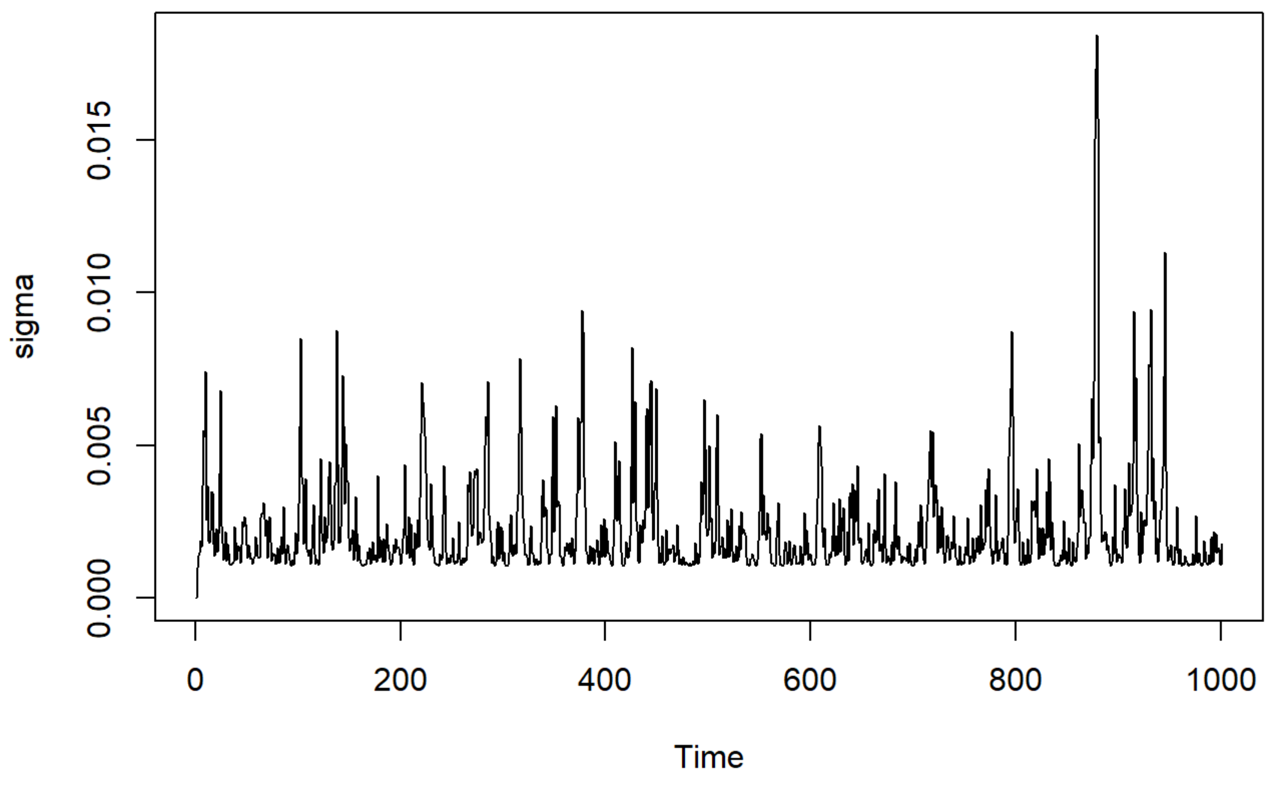
We will attempt to do the simulation of GARCH(1,1) ourselves. Starting off, we may define the coefficients involved:

```r
sigma <- 0:1000
alpha0 <- 10^(-6)
alpha1 <- 0.9
beta1 <- 0.1
n <- 10^3
Z <- rnorm(n+1)
X <- 0:1000
```

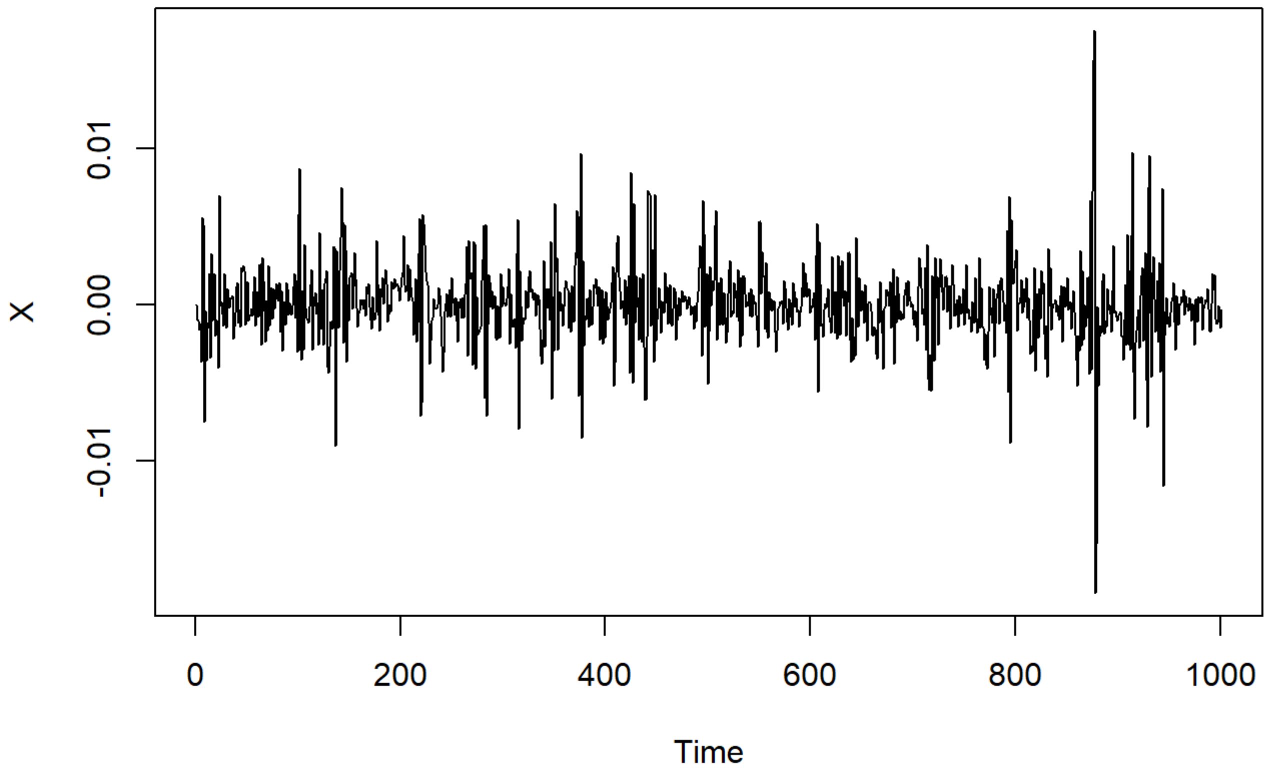
a)

We may then employ the main simulation

```
for (i in 2:(n+1)) {
sigma[i] <- sqrt(alpha0+alpha1*(X[i-1])^2 + beta1*(sigma[i-1])^2)
X[i] <- sigma[i]*Z[i]
}
ts.plot(sigma)
```



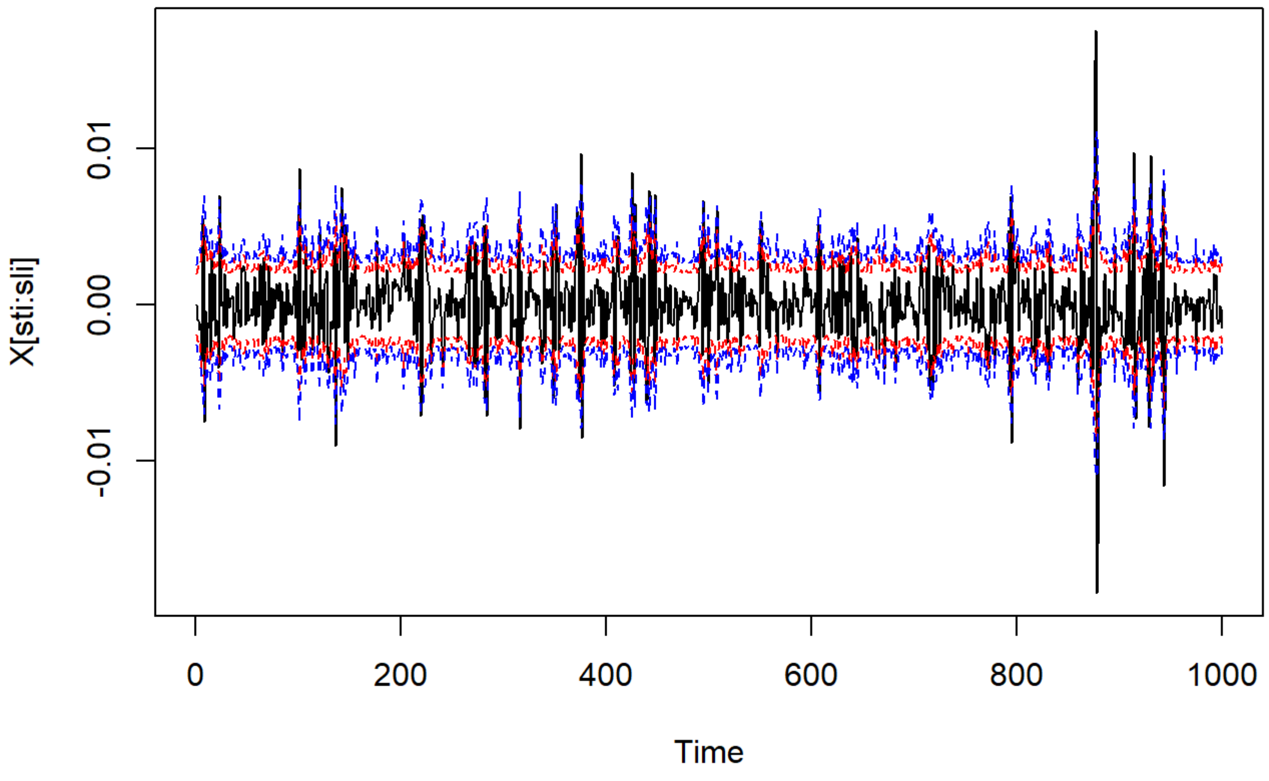
```
ts.plot(X)
```



b)

As we for iid standard normal noise (Z_t) have $X_{t+1}|X_t, X_{t-1}, \ldots \sim \mathcal{N}\left(0, \sigma_t^2\right)$ we may calculate 95% and 99% conditional forecast intervals based on `sigma` from a) as well as plot these, with the 95% interval in red, and the 99% interval in blue.

```
q95 <- qnorm(0.975)
q99 <- qnorm(0.995)
condpred95U <- 0+q95*sqrt(sigma[-1]/n)
condpred95L <- 0-q95*sqrt(sigma[-1]/n)
condpred99U <- 0+q99*sqrt(sigma[-1]/n)
condpred99L <- 0-q99*sqrt(sigma[-1]/n)
sti <- 1
sli <- 1000
ts.plot(X[sti:sli]);lines(condpred95U[sti:sli], lty = 2, col = "red");lines(condpred95L[sti:sli], lty =
2, col = "red");lines(condpred99U[sti:sli], lty = 2, col = "blue");lines(condpred99L[sti:sli], lty = 2,
col = "blue")
```



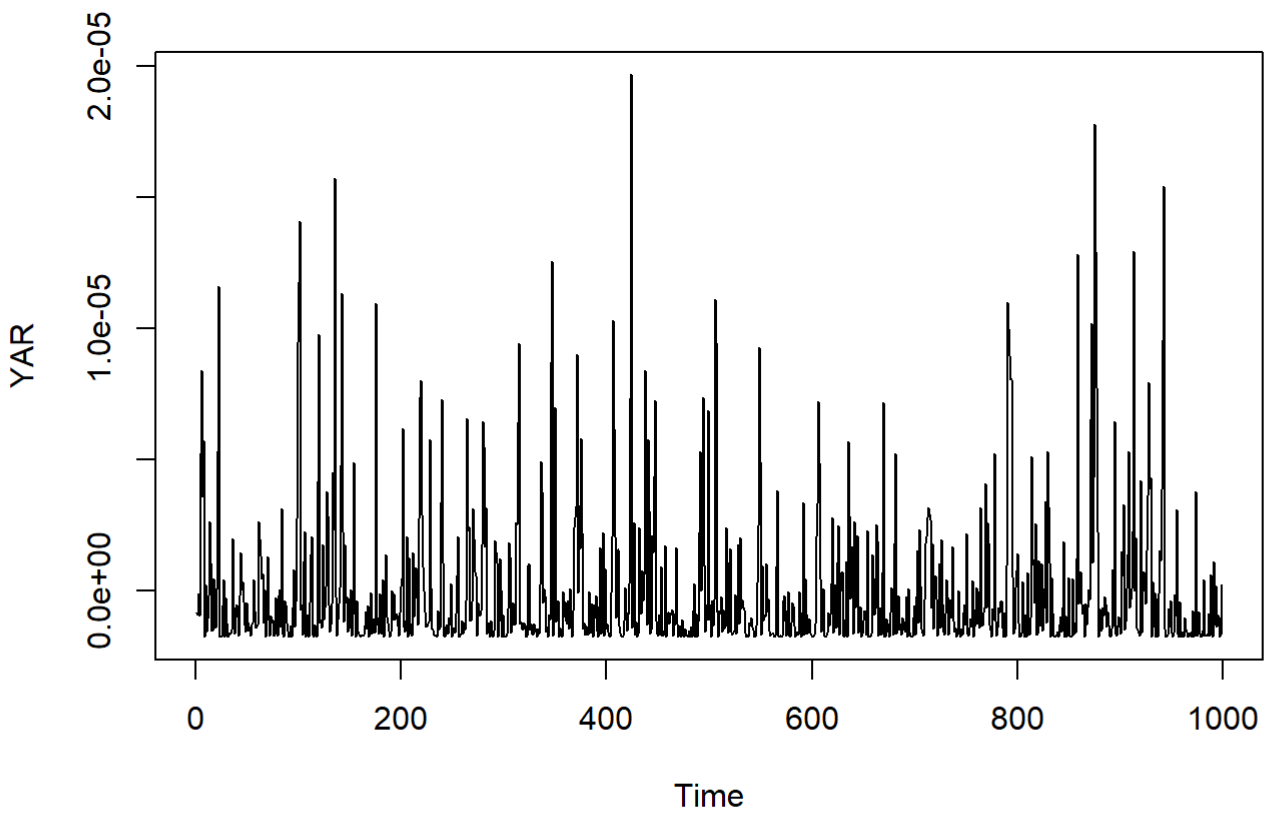
c)

We may reuse the initialisation above, and start the simulation of the ARCH(1) and of the Y_t's

```
XARCH <- 1:1000
sigmaARCH <- 1:1000
alpha1ARCH <- 0.5
sigmaARCH[1] <- alpha0+alpha1ARCH*0
XARCH[1] <- sigmaARCH[1]*Z[1]
for (i in 2:n) {
  sigmaARCH[i] <- sqrt(alpha0+alpha1ARCH*(XARCH[i-1])^2)
  XARCH[i] <- sigmaARCH[i]*Z[i]
}
(estX0Squared <- gammaf(XARCH, 0))
```

```
## [1] 1.762877e-06
```

```
YAR <- XARCH[-1]^2-estX0Squared
ts.plot(YAR)
```



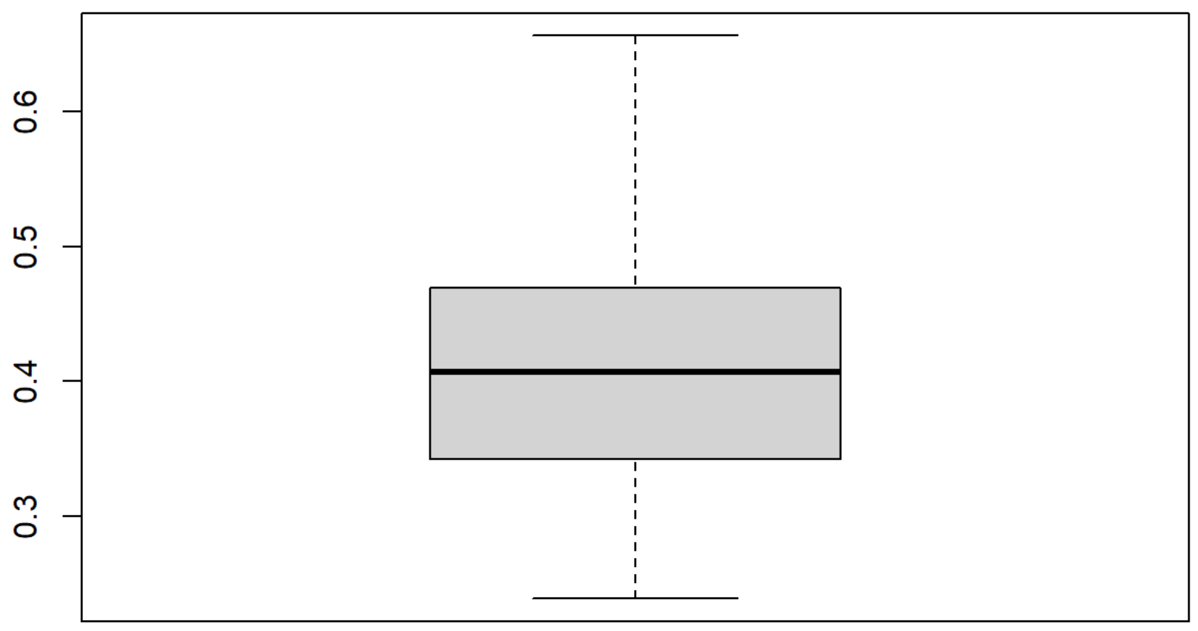
We will estimate α_1 from data, akin to how it was done in problem 1:

```
(phihYAR <- gammaf(YAR,1)/gammaf(YAR,0))
```

```
## [1] 0.3825345
```

which is fairly close to the sought after 0.5. Repeating the simulation and estimation 100 more times, we will now have to simulate new Z's;

```
phihYARv <- 1:100
for (j in 1:100) {
  Z <- rnorm(n)
  XARCH <- 1:1000
  sigmaARCH <- 1:1000
  sigmaARCH[1] <- alpha0+alpha1ARCH*0
  XARCH[1] <- sigmaARCH[1]*Z[1]
  for (i in 2:n) {
  sigmaARCH[i] <- sqrt(alpha0+alpha1ARCH*(XARCH[i-1])^2)
  XARCH[i] <- sigmaARCH[i]*Z[i]
}
  estX0Squared <- gammaf(XARCH, 0)
  YAR <- XARCH[-1]^2-estX0Squared
  phihYARv[j] <- gammaf(YAR,1)/gammaf(YAR,0)
}
boxplot(phihYARv)
```



We see that the parameter is seemingly being underestimated.

d)

Note that with the coefficients from a) with $\alpha_1 = 0.9,\ \beta_1 = 0.1$ that for $Z_i \sim \mathcal{N}(0,1)$ for $i = 1, \dots, n$ we have for $\kappa = 2$:

$$E\Big[\big(\alpha_1 Z_0^2 + \beta_1\big)^{\frac{\kappa}{2}}\Big] = E\big[\big(0.9Z_0^2 + 0.1\big)\big] = 0.9EZ_0^2 + 0.1 = 1.$$

e)

We will do the simulation of the σ_t's from the model in a)

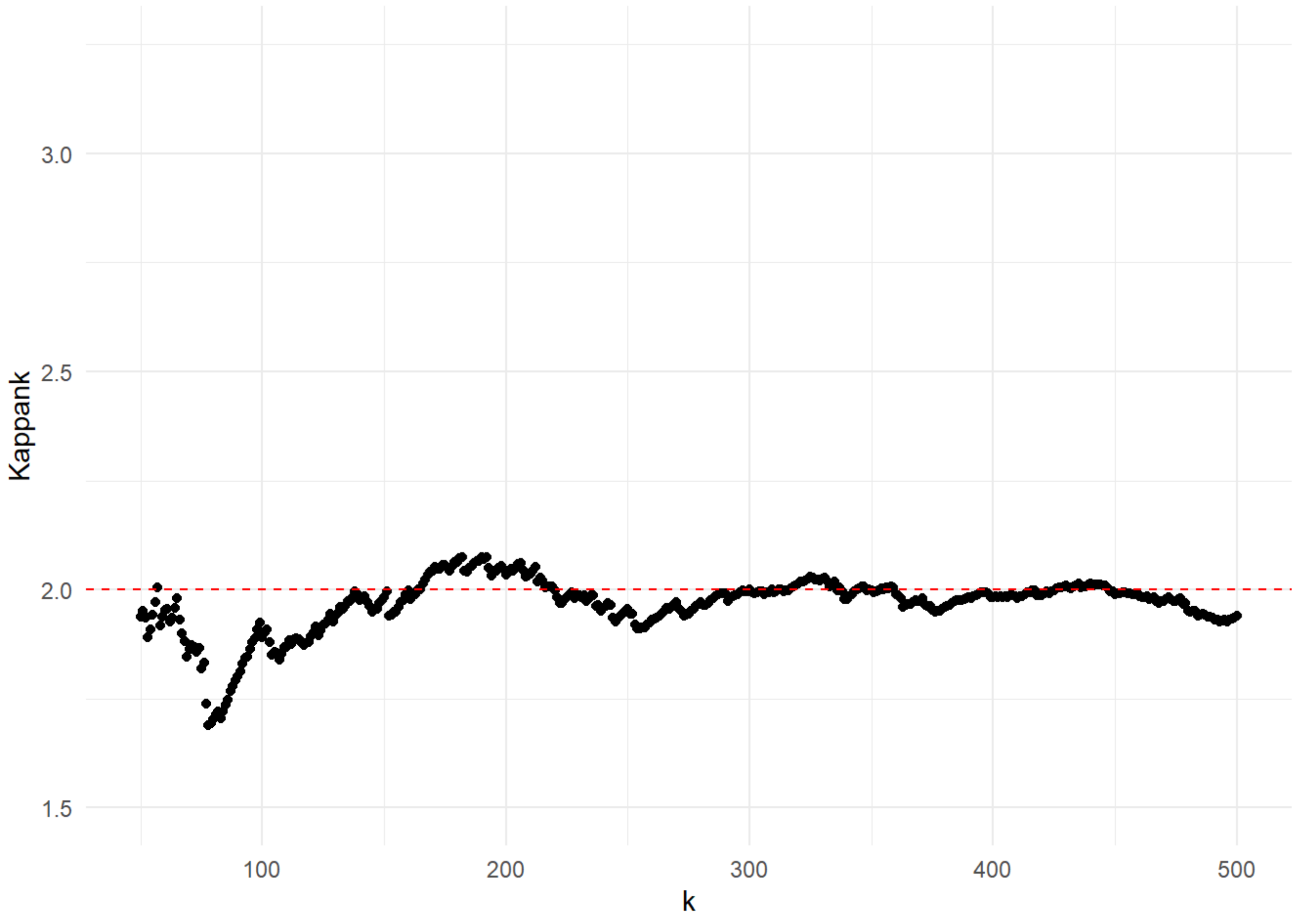
```r
n2e <- 10^4
Z2e <- rnorm(n2e+1)
X2e <- 0:n2e
sigma2e <- 0:n2e
for (i in 2:(n2e+1)) {
  sigma2e[i] <- sqrt(alpha0+alpha1*(X2e[i-1])^2 + beta1*(sigma2e[i-1])^2)
  X2e[i] <- sigma2e[i]*Z2e[i]
}
sigma2e <- sigma2e[-1]
```

We may then prepare to calculate and plot the Hill estimators, by creating the following functions:

```r
Hill <- function(Y,k) {
   n <- length(Y)
   Y <- sort(Y)
   kapk <- 0
   for (i in 1:k) {
      tempi <- log(Y[n-i+1]/(Y[n-k]))
      kapk <- kapk + tempi
   }
   1/(1/k*kapk)
}
HillVec <- function(Y,range) {
  n <- length(Y)
  Y <- sort(Y)
  kapkVec <- length(range)
  kmin <- min(range)
  for (k in range) {
    kapkVec[k-kmin+1] <- Hill(Y,k)
  }
  kapkVec
}
HillVecPlot <- function(Y, range) {
  kapkVec <- HillVec(Y,range)
  ggplot() + geom_point(mapping = aes(x=range, y=kapkVec)) + labs(x="k", y="Kappank")
}
```

We may thus plot:

```r
kRange <- 50:500
HillVecPlot(sigma2e, kRange) + geom_hline(yintercept = 2, colour = "red", lty = 2) + ylim(1.5,3.25)
```



From the plot we see a long run of ‘stability’ from $k = 300$ to $k = 500$ at approximately $\kappa_n^{(k)} = 2$ in agreement with $\kappa = 2$.

f)

We may import S&P 500 Data, filter for year after 2000, create a returns column, and choosing to remove `NA` rows.

```
Data <- read_csv("Data.csv",col_types =
                       cols(col_date(),
                            col_double(),
                            col_double(),
                            col_double(),
                            col_double(),
                            col_double(),
                            col_integer())) %>% filter(year(Date)>=2000) %>%  mutate(Returns = (Close -
lag(Close))/lag(Close)) %>% drop_na()
```

```
## Warning: 876 parsing failures.
## row       col expected actual       file
##  32 Open      a double   null 'Data.csv'
##  32 High      a double   null 'Data.csv'
##  32 Low       a double   null 'Data.csv'
##  32 Close     a double   null 'Data.csv'
##  32 Adj Close a double   null 'Data.csv'
## ... ......... ........ ...... ..........
## See problems(...) for more details.
```

```
head(Data, 10)
```

```
## # A tibble: 10 x 8
##    Date        Open  High   Low Close `Adj Close`   Volume  Returns
##    <date>     <dbl> <dbl> <dbl> <dbl>       <dbl>    <int>    <dbl>
##  1 2000-01-04 6747. 6755. 6510. 6587.       6587. 46678400 -0.0243
##  2 2000-01-05 6586. 6586. 6389. 6502.       6502. 52682800 -0.0129
##  3 2000-01-06 6501. 6539. 6403. 6475.       6475. 41180600 -0.00418
##  4 2000-01-07 6490. 6792. 6470. 6781.       6781. 56058900  0.0473
##  5 2000-01-10 6785. 6975. 6785. 6926.       6926. 42006200  0.0213
##  6 2000-01-11 6926. 6944. 6821. 6891.       6891. 43890000 -0.00495
##  7 2000-01-12 6879. 6914. 6783. 6913.       6913. 34683100  0.00313
##  8 2000-01-13 6913. 7106. 6899. 6956.       6956. 48116200  0.00624
##  9 2000-01-14 6960. 7207. 6960. 7173.       7173. 47040900  0.0312
## 10 2000-01-17 7177. 7297. 7113. 7259.       7259. 44872800  0.0119
```

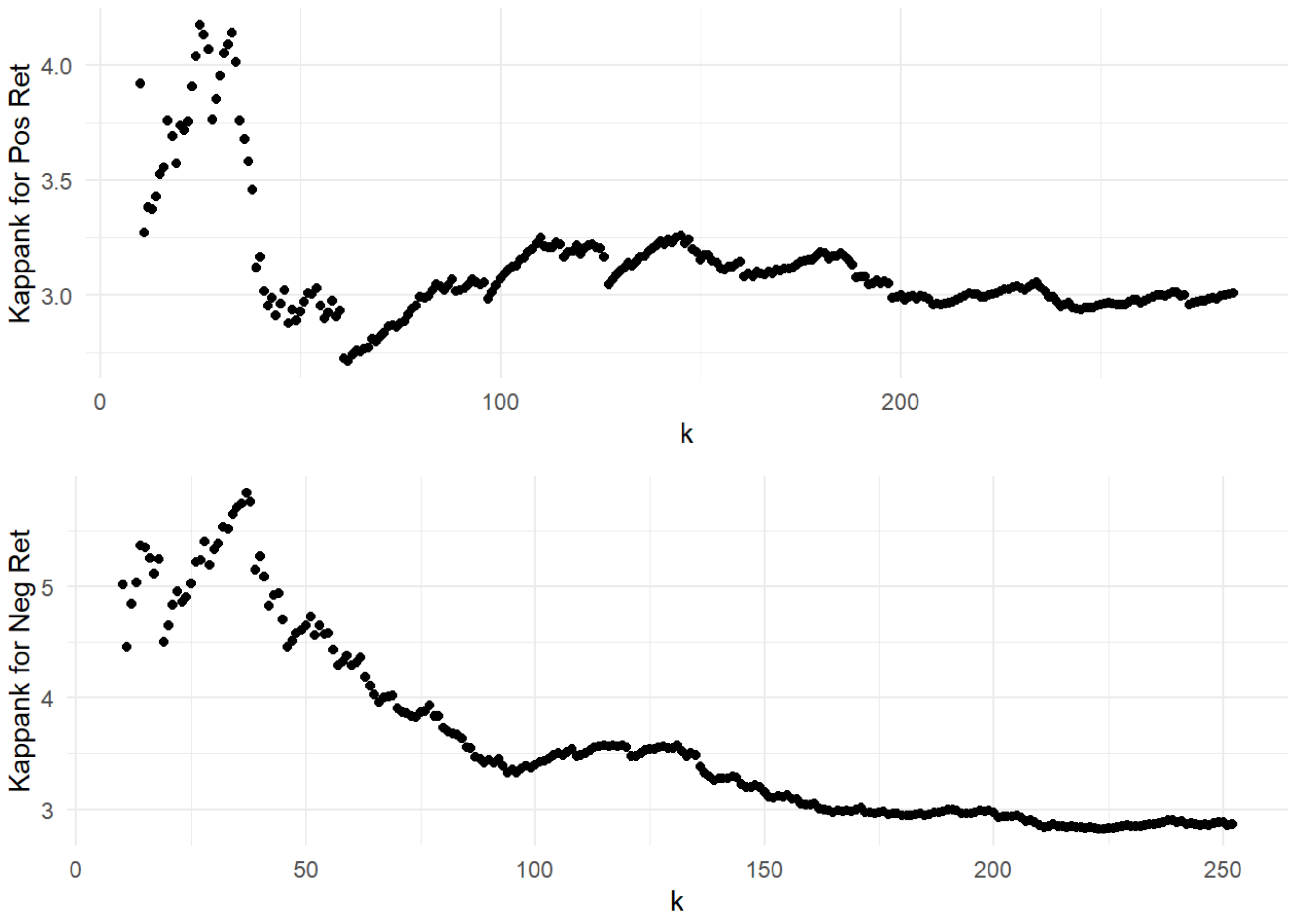
```
Returnvals <- Data[,8]
```

We may subset the positive and negative returns of the data, and provide further setup

```
posReturnsSORTED <- c(Returnvals %>% filter(Returns>0) %>% arrange(Returns))[[1]]
negReturnsSORTED <- c(Returnvals %>% filter(Returns<0) %>% mutate(Returns = Returns * (-1)) %>% arrange
(Returns))[[1]]
kPosRetRange <- 10:(floor(length(posReturnsSORTED)/10))
kNegRetRange <- 10:(floor(length(negReturnsSORTED)/10))
```

Using the previously built `HillVecPlot` function, we may thus generate the Hill estimates, and plot them:

```
p0 <- HillVecPlot(posReturnsSORTED, kPosRetRange) + labs(y = "Kappank for Pos Ret")
p1 <- HillVecPlot(negReturnsSORTED, kNegRetRange) + labs(y = "Kappank for Neg Ret")
grid.arrange(p0,p1, nrow = 2)
```



We note that both seem to stabilise around $\kappa_n^{(k)} = 3$.