Bruce Campell NCSU ST 534 HW 3

Problems 3.10, 3.18, and 3.21

Shumway, Robert H.; Stoffer, David S. Time Series Analysis and Its Applications: With R Examples (Springer Texts in Statistics)

23 October, 2017

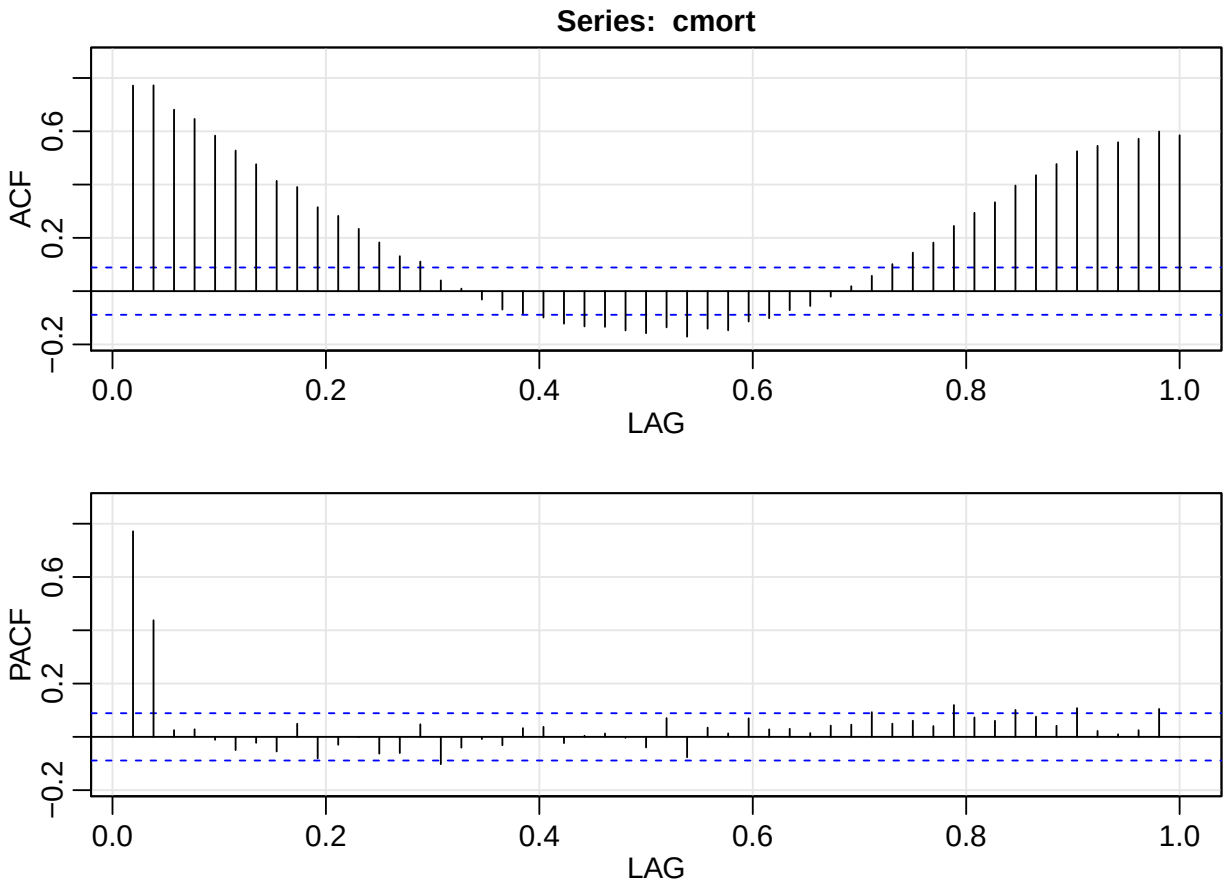
3.10 cmort analysis - prediction

Let x_t represent the cardiovascular mortality series (cmort) discussed in Chapter 2, Example 2.2.

(a) Fit an $AR(2)$ to x_t using linear regression as in Example 3.17.

Since the cmort data is weekly for 10 years - we choose a lag of 52 to investigate.

```
rm(list = ls())
library(astsa)
data(cmort, package = "astsa")
max.lag <- 52
invisible(acf2(cmort, max.lag = max.lag))
```




```
(cmort.ar2 = ar.ols(cmort, order = 2, demean = FALSE, intercept = TRUE))
```

```
##
## Call:
## ar.ols(x = cmort, order.max = 2, demean = FALSE, intercept = TRUE)
##
## Coefficients:
##      1       2
## 0.4286  0.4418
##
## Intercept: 11.45 (2.394)
##
## Order selected 2  sigma^2 estimated as  32.32
```

```
pander(data.frame(mean = cmort.ar2$asy.se.coef$x.mean), caption = "mean")
```

Table 1: mean

mean
2.394

```
pander(data.frame(se.estimates = cmort.ar2$asy.se.coef$ar), caption = "standard errors o
```

Table 2: standard errors of the estimates

se.estimates
0.03979
0.03976

From the documentation for ar.ols,model we fit is given by

$$(x[t] - m) = a[0] + a[1] * (x[t-1] - m) + . + a[p] * (x[t-p] - m) + e[t]$$

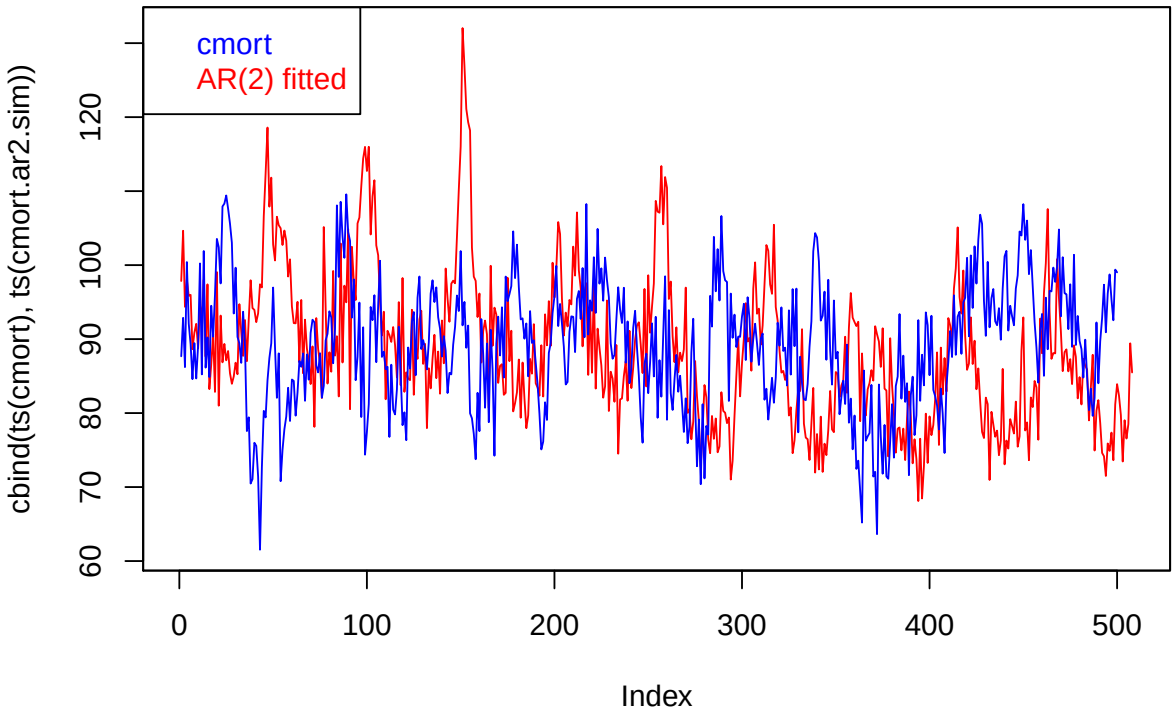
where $a[0]$ is zero unless intercept is true, and m is the sample mean if demean is true, zero otherwise.

Let's plot our model and the original series. We also check the coefficient values are consistent with those provided by stats::arima.

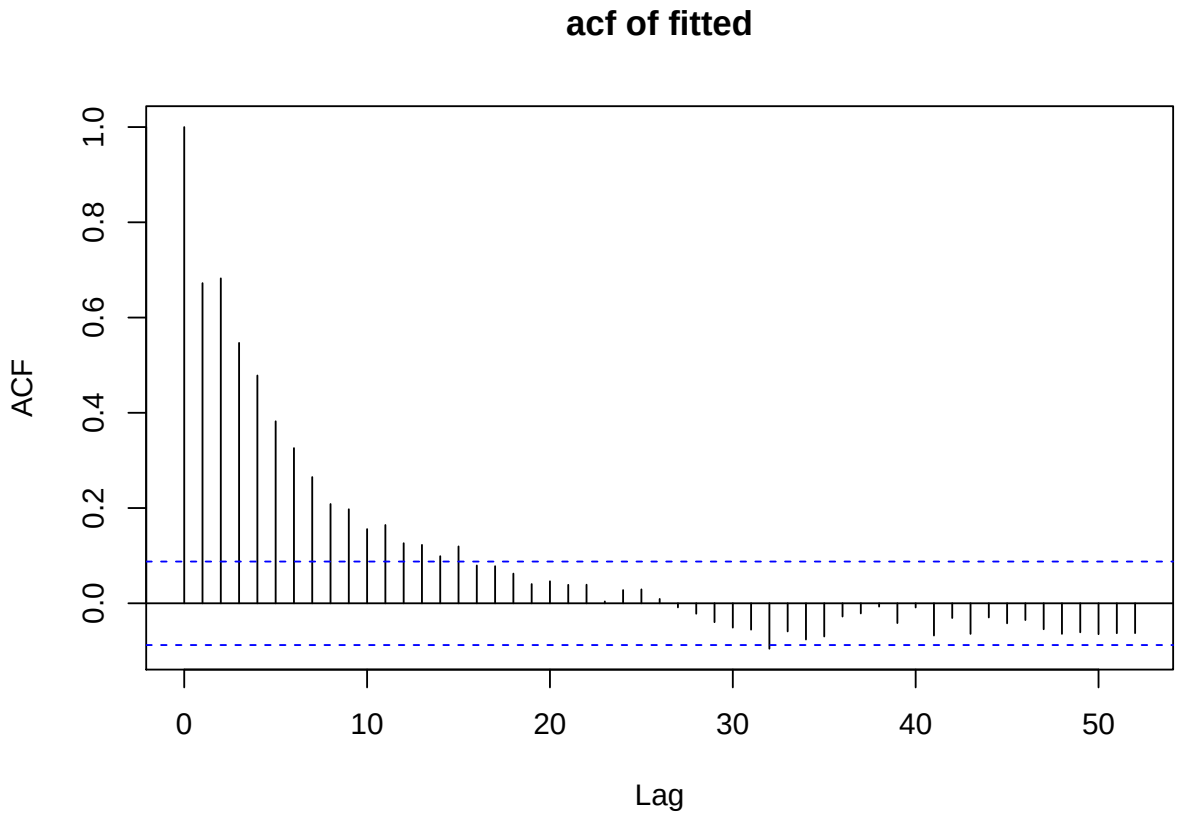
```
cmort.arima.fit <- arima(cmort, order = c(2, 0, 0))
cmort.arima.fit
```

```
##
## Call:
## arima(x = cmort, order = c(2, 0, 0))
##
## Coefficients:
##          ar1     ar2  intercept
##       0.4301  0.4424    88.8538
## s.e.  0.0397  0.0398     1.9407
##
## sigma^2 estimated as 32.37:  log likelihood = -1604.71,  aic = 3217.43
```

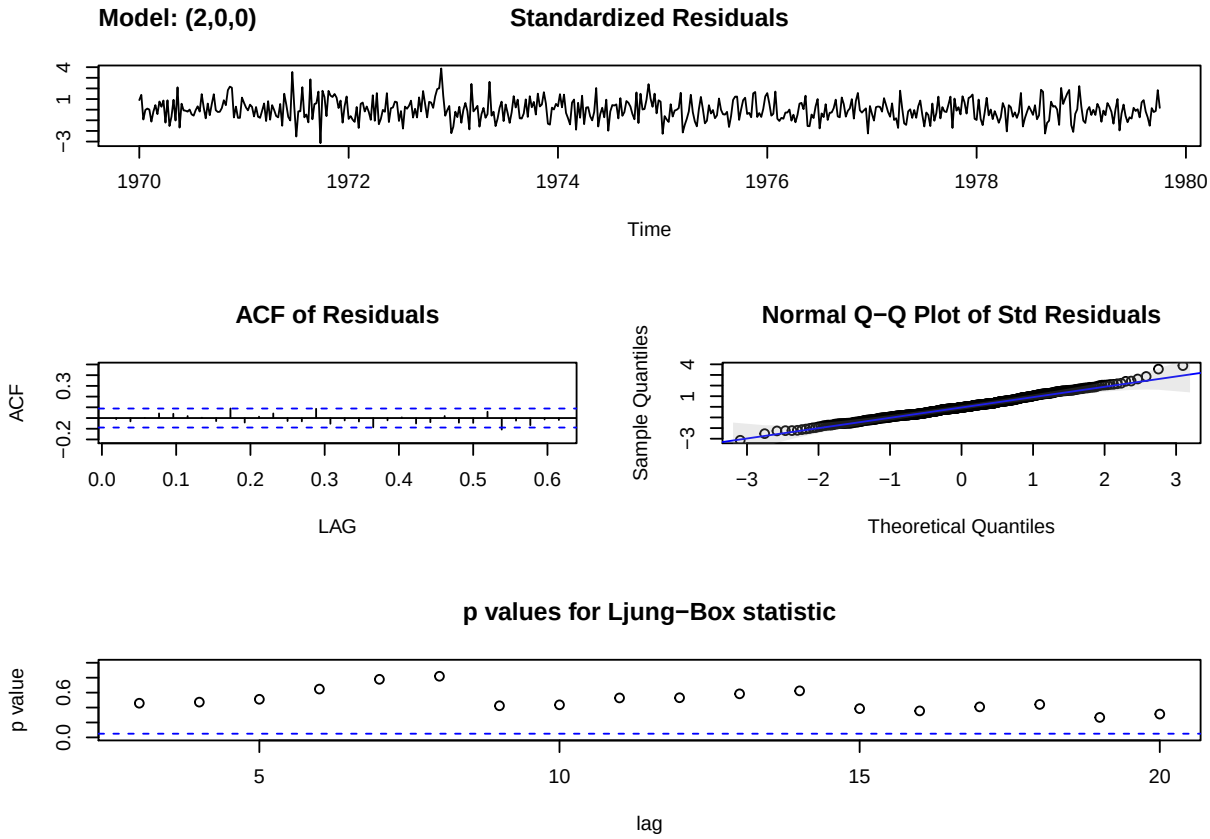
```
cmort.ar2.sim <- arima.sim(list(order = c(2, 0, 0), ar = c(0.4286, 0.4418)),
    sd = sqrt(32.32), n.start = 52 * 5, n = 500) + mean(cmort)
library(zoo)
plot.zoo(cbind(ts(cmort), ts(cmort.ar2.sim)), plot.type = "single", col = c("red",
    "blue"))
legend("topleft", title.col = "black", c("cmort", "AR(2) fitted"), text.col = c("blue",
    "red"), text.font = 1, cex = 1)
```




```
acf(cmort.ar2.sim, lag.max = max.lag, main = "acf of fitted")
```



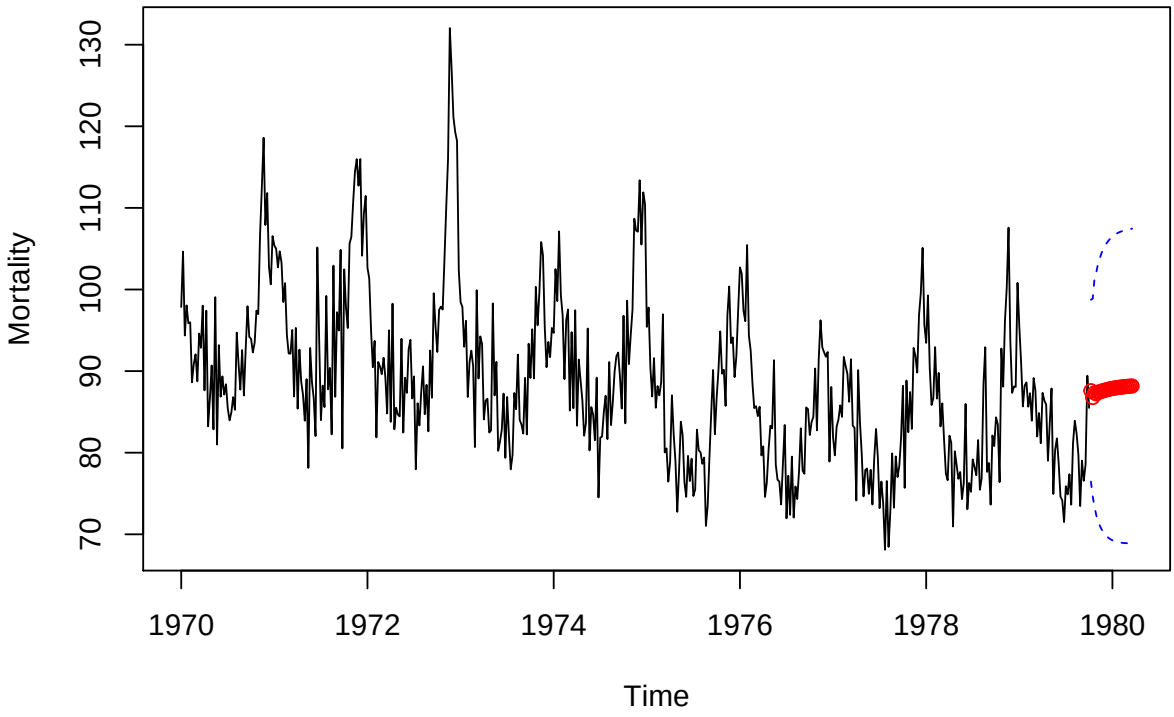
Now we use the textbook author's function astsa::sarima to fit the model.



(b) Assuming the fitted model in (a) is the true model, find the forecasts over a four-week horizon, x^n_{nn+m}, for $m = 1, 2, 3, 4$ and the corresponding 95% prediction intervals.

We plot the prediction first. The 95% prediction interval is displayed as dashed lines.

```
ols.fit = ar.ols(cmort, order = 2, demean = FALSE, intercept = TRUE)
fore = predict(ols.fit, n.ahead = 24, interval = "prediction")
ts.plot(cmort, fore$pred, col = 1:2, ylab = "Mortality")
lines(fore$pred, type = "p", col = 2)
lines(fore$pred + 1.96 * fore$se, lty = "dashed", col = 4)
lines(fore$pred - 1.96 * fore$se, lty = "dashed", col = 4)
```



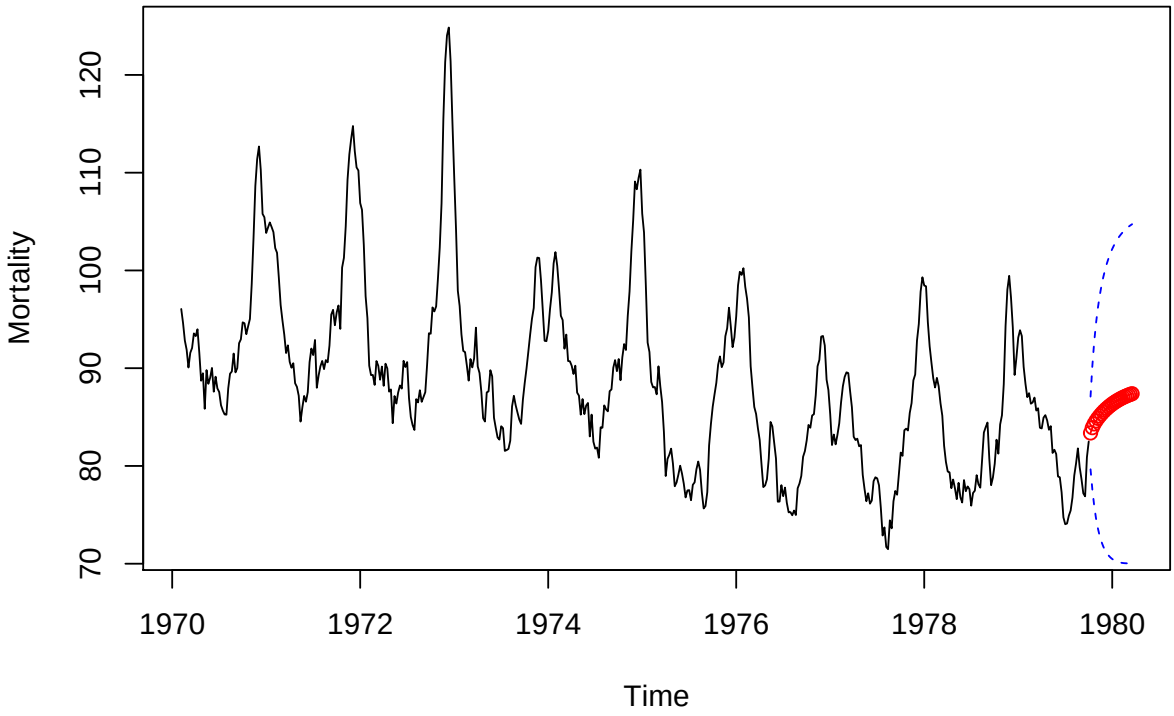

```
fore = predict(ols.fit, n.ahead = 4, interval = "prediction")
predicted <- fore$pred
right.pi <- fore$pred + 1.96 * fore$se
left.pi <- fore$pred - 1.96 * fore$se
pi.width <- 1.96 * fore$se
pander(data.frame(left.pi = left.pi, predicted = predicted, right.pi = right.pi,
    pi.width = pi.width), caption = "Predicted with limits of prediction interval.")
```

Table 3: Predicted with limits of prediction interval.

left.pi	predicted	right.pi	pi.width
76.46	87.60	98.74	11.14
74.64	86.76	98.89	12.12
73.35	87.34	101.32	13.98
72.33	87.21	102.10	14.88

For fun we apply an MA(5) smoothing filter to the cmort data and see how that affects the prediction intervals.

```
cmort.sm <- filter(na.exclude(cmort), rep(1/4, 4), sides = 1)
cmort.sm.trim <- window(cmort.sm, start = start(cmort.sm) + 4 * deltat(cmort.sm))
ols.fit = ar.ols(cmort.sm.trim, order = 2, demean = FALSE, intercept = TRUE)
fore = predict(ols.fit, n.ahead = 24, interval = "prediction")
ts.plot(cmort.sm.trim, fore$pred, col = 1:2, ylab = "Mortality")
lines(fore$pred, type = "p", col = 2)
lines(fore$pred + 1.96 * fore$se, lty = "dashed", col = 4)
lines(fore$pred - 1.96 * fore$se, lty = "dashed", col = 4)
```



3.18 cmort analyis - estimation

Fit an AR(2) model to the cardiovascular mortality series (cmort) discussed in Chapter 2, Example 2.2. using linear regression and using Yule- Walker.

```
ols.fit = ar.ols(cmort, order = 2, demean = FALSE, intercept = TRUE)
pander(data.frame(ar.ols = ols.fit$ar), caption = "AR(2) coefficicients fit by OLS")
```

Table 4: AR(2) coefficicients fit by OLS

ar.ols
0.4286
0.4418

```
cmort.yw = ar.yw(cmort, order = 2)
pander(data.frame(ar.yw = cmort.yw$ar), caption = "AR(2) coefficicients fit by Yule-Walk
```

Table 5: AR(2) coefficicients fit by Yule-Walker method

ar.yw
0.4339
0.4376

(a) Compare the parameter estimates obtained by the two methods.

```
pander(data.frame(var.pred = ols.fit$var.pred), caption = "AR(2) prediction variance fit
```

Table 6: AR(2) prediction variance fit by OLS

var.pred
32.32

```
pander(data.frame(var.pred = cmort.yw$var.pred), caption = "AR(2) prediction variance fi
```

Table 7: AR(2) prediction variance fit by Yule-Walker method

var.pred
32.84

The models are comparable in terms of the predicted variance. Predicted variance is reported by both fitting methods and is and estimate of the portion of variance not explained by the fitted model.

(b) Compare the estimated standard errors of the coefficients obtained by linear regression with their corresponding asymptotic approximations, as given in Property 3.10.

```
pander(data.frame(coeff.se = ols.fit$asy.se.coef$ar), caption = "AR(2) coefficient stand
```

Table 8: AR(2) coefficient standard error fit by OLS

coeff.se
0.03979
0.03976

From the very important large sample result for the distribution of estimators of ARMA processes we know that

$$\hat{\phi} \underset{d}{\sim} N(\phi, \frac{\sigma^2 \Gamma^{-1}}{n})$$

We can substitute our estimates $\hat{\sigma}$ and $\hat{\Gamma^{-1}}$ into this expression. The textbook has worked out the expression for $\sigma_2 \Gamma^{-1}_{AR(2)}$

$$\begin{pmatrix} \hat{\phi}_1 \\ \hat{\phi}_2 \end{pmatrix} \underset{d}{\sim} N(\begin{pmatrix} \hat{\phi}_1 \\ \hat{\phi}_2 \end{pmatrix}, \frac{1}{n} \begin{pmatrix} 1-\phi_2^2 & -\phi_1(1+\phi_2) \\ -\phi_1(1+\phi_2) & 1-\phi_2^2 \end{pmatrix})$$

+1 point for $LaTeX$

Now calculating the approx asymptotic SE using the estimates of the coefficients from our OLS fit we have

```
se.phi <- sqrt(1/length(cmort) * (1 - ols.fit$asy.se.coef$ar[1]^2))

pander(data.frame(se.asymptotic = se.phi), caption = "Asyptotic SE")
```

Table 9: Asyptotic SE This is in agreement with the result from the ols function.

se.asymptotic
0.04433

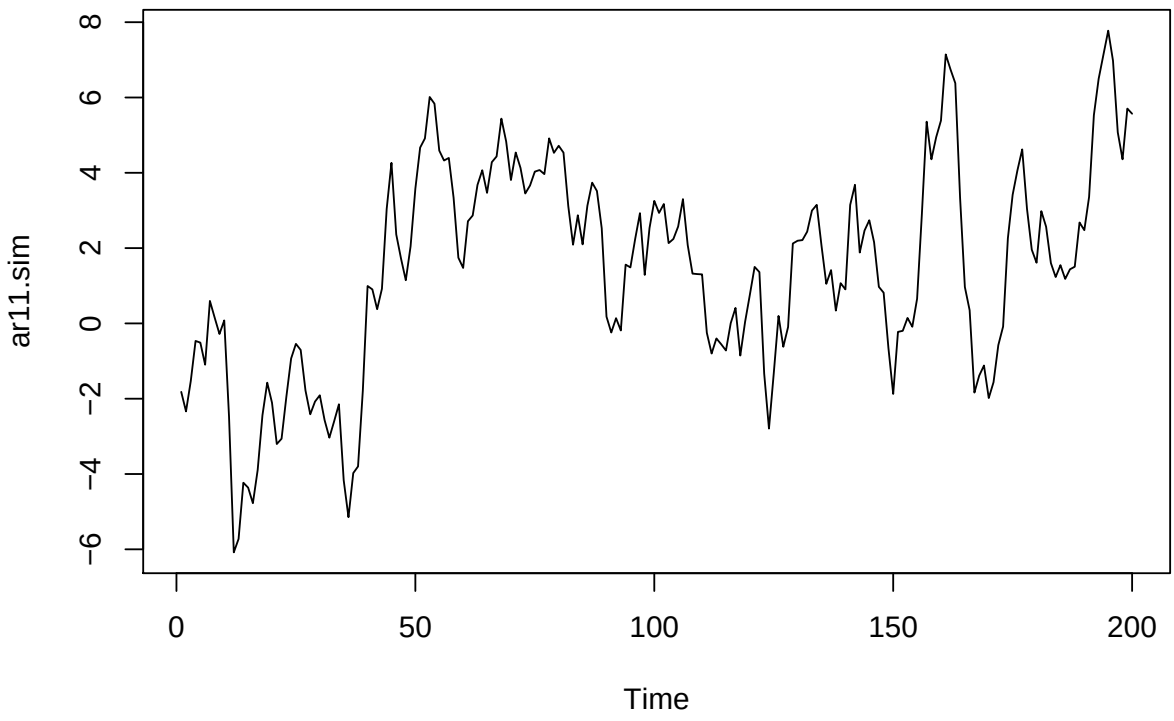
3.21 MLE of ARMA(1,1)

Generate 10 realizations of length $n = 200$ each of an $ARMA(1,1)$ process with $\phi = .9$, $\theta = .5$ and $\sigma^2 = 1$. Find the MLEs of the three parameters in each case and compare the estimators to the true values.

First we generate and plot one realization to get an idea of what this model looks like.

```
ar11.sim <- arima.sim(list(order = c(1, 0, 1), ar = c(0.9), ma = c(0.5)), sd = 1,
    n.start = 200, n = 200)
```

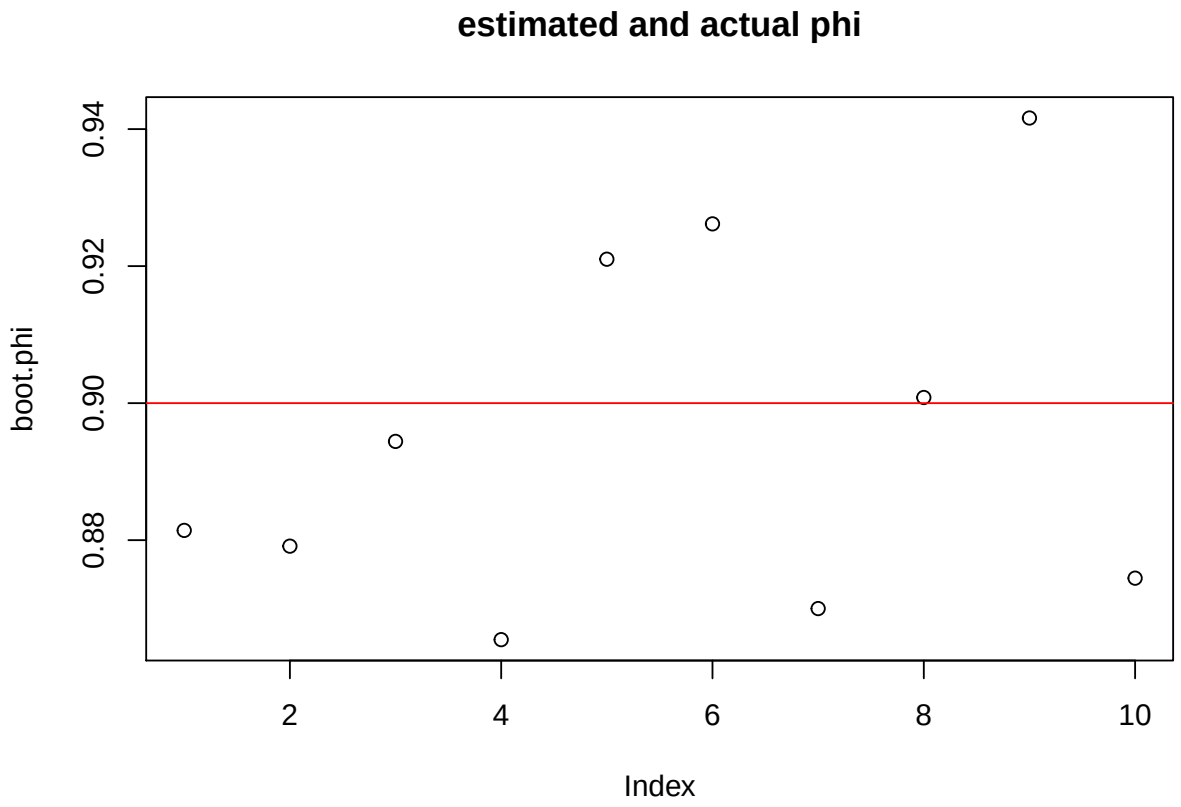
```r
plot(ar11.sim)
```



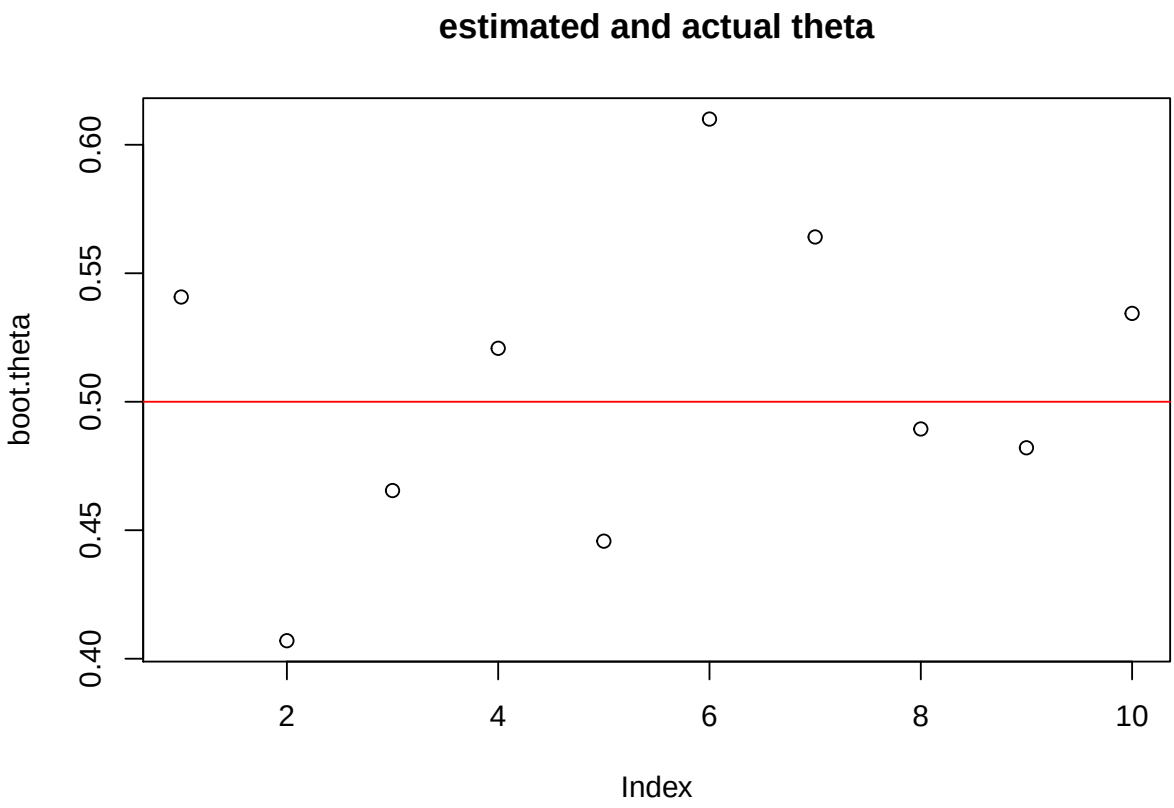
We'll perform the calculations with the `stats::arima` function. We need to set the method parameter to fit using maximum likelihood fitting. Note that the book uses ar.mle for an AR process. TOTO - revisit the relationship between these - they're both in the core stats package.

```r
phi <- 0.9
theta <- 0.5
sigma.sq <- 1

simulationCount <- 10
boot.phi <- matrix(0, nrow = simulationCount, ncol = 1)
boot.theta <- matrix(0, nrow = simulationCount, ncol = 1)
boot.sigma.sq <- matrix(0, nrow = simulationCount, ncol = 1)


for (i in 1:simulationCount) {
    ar11.sim <- arima.sim(list(order = c(1, 0, 1), ar = c(phi), ma = c(theta)),
        sd = sigma.sq, n.start = 200, n = 200)
    ar11.est <- arima(ar11.sim, order = c(1, 0, 1), method = "ML")
```

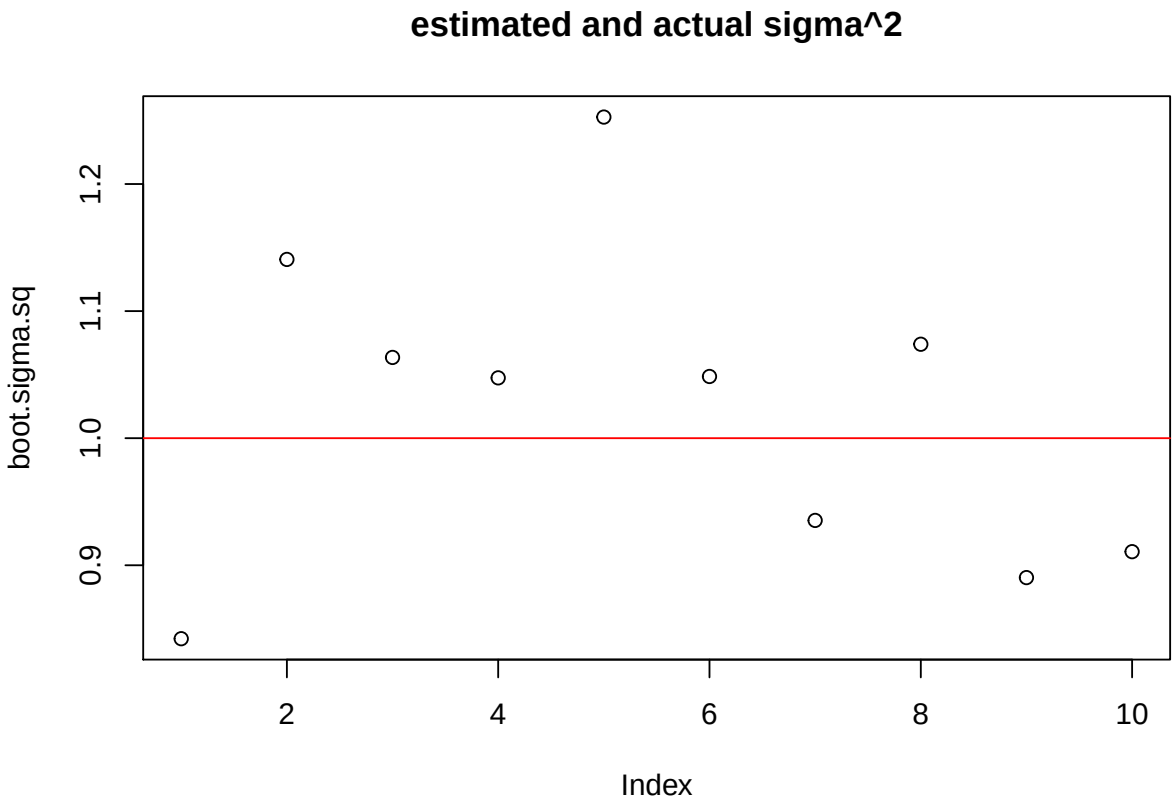
```
    boot.phi[i] = ar11.est$coef[1]
    boot.theta[i] = ar11.est$coef[2]
    boot.sigma.sq[i] = ar11.est$sigma2
}

plot(boot.phi, main = "estimated and actual phi")
abline(h = phi, col = "red")
```




```
plot(boot.theta, main = "estimated and actual theta")
abline(h = theta, col = "red")
```




```
plot(boot.sigma.sq, main = "estimated and actual sigma^2")
abline(h = sigma.sq, col = "red")
```


estimated and actual sigma^2
boot.sigma.sq
0.9
1.0
1.1
1.2
2
4
6
8
10
Index