AR simulated example Chapter 4

Kathleen M. Gates

Set up the environment:

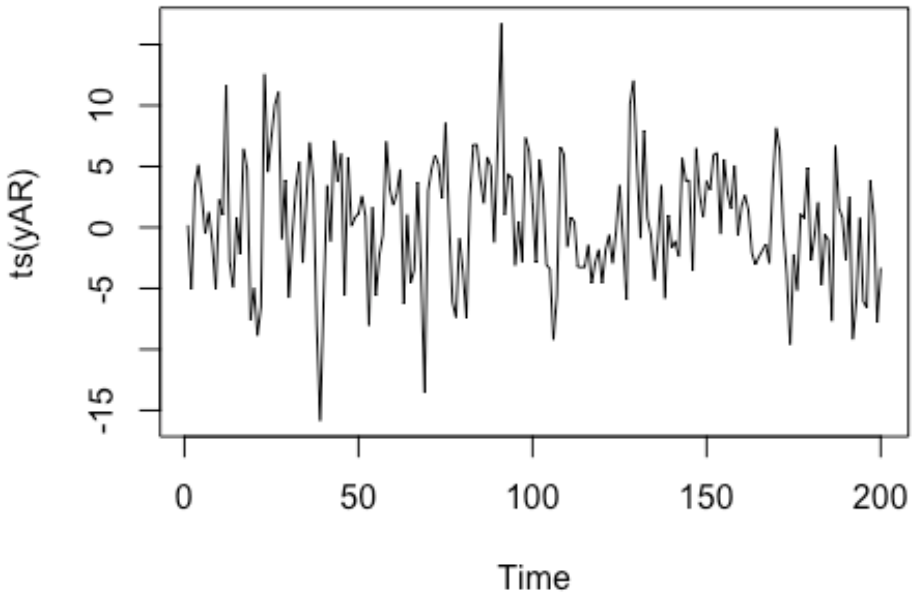
```
library(pracma)

library(signal)

library(fUnitRoots)

library(tseries)
library(portes)
```

Look at trends and order

First let's generate some data with an AR(1) coefficient of 0.20 and no trend:

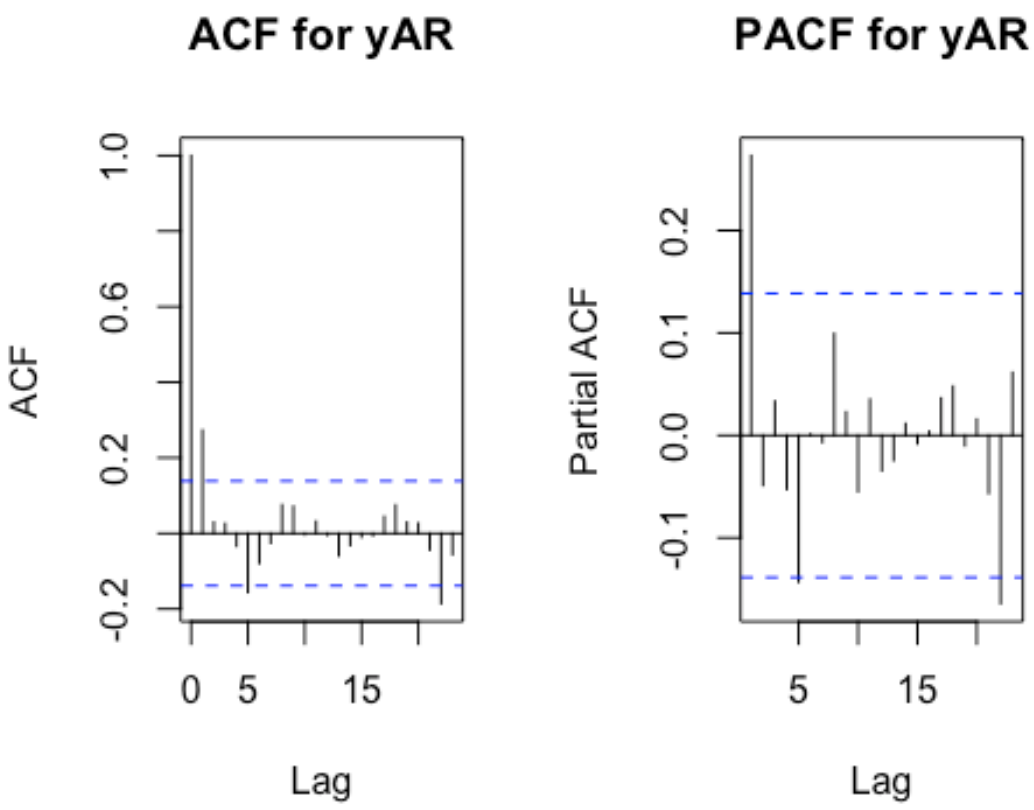
$$y(t) = 0.20 * y(t-1) + \zeta(t)$$

```
#initiate
seed <- 123456
noise <- rnorm(200, 0, 5)

# AR series
yAR <-  rnorm(1, 0, 1)
for (p in 2:200)
yAR[p]     <- 0.20*yAR[p-1] + noise[p]
plot(ts(yAR))
```



Test order and coefficients.

```
par(mfrow = c(1,2))
plot(acf(yAR, plot = F), main = "ACF for yAR")
plot(pacf(yAR, plot = F), main = "PACF for yAR")
```



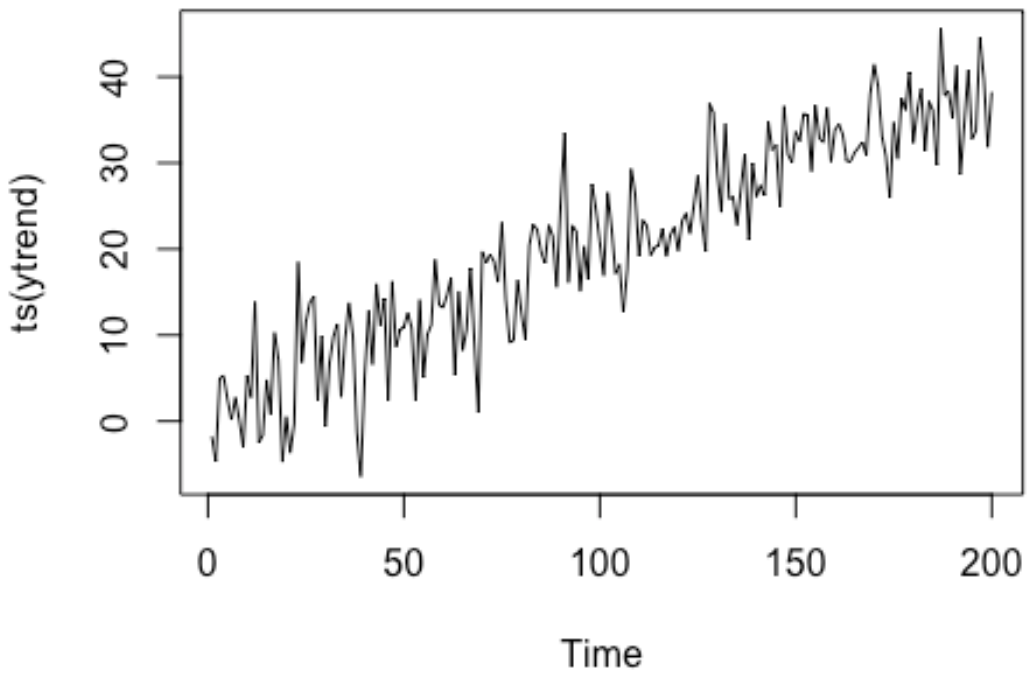

```
fitAR <- ar(yAR)

# examine order
fitAR$order

## [1] 1

# examine coefficient estimates
fitAR$ar

## [1] 0.2730648
```

The lag order and estimates look good.

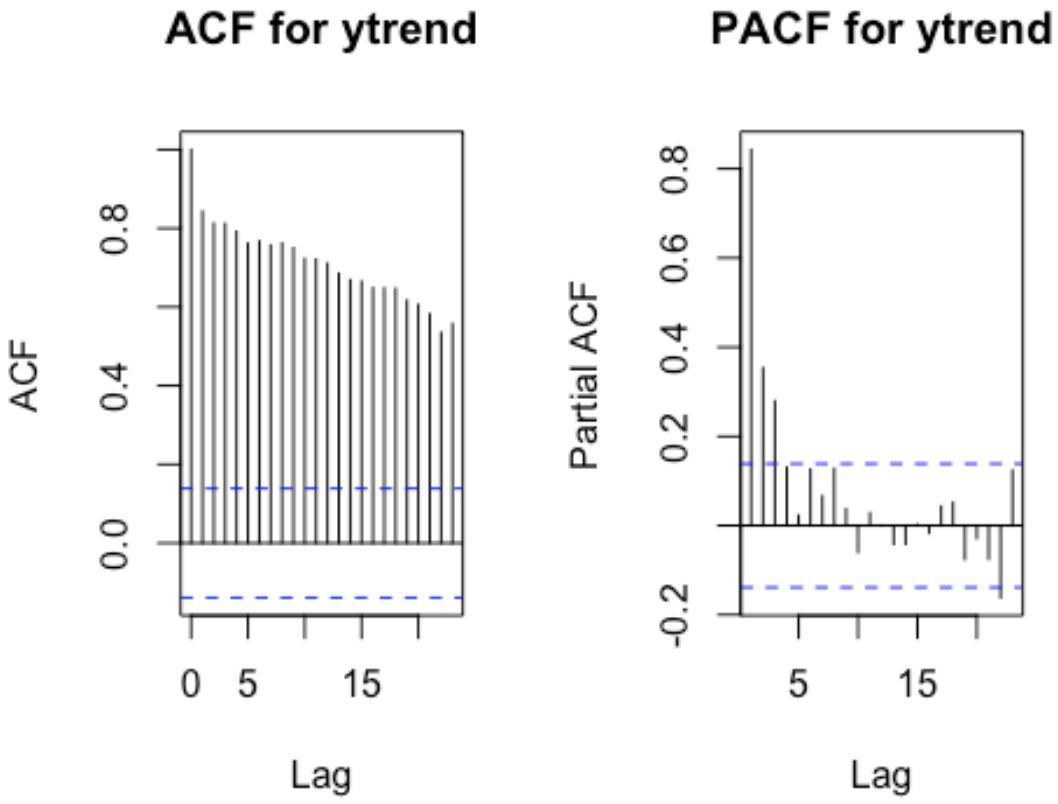
Let's examine a linear trend without any autoregressive relations and see what happens.

$$y(t) = 0.20 * time(t) + \zeta(t)$$

```
# trend series
time  <- 1:200
ytrend <-  rnorm(1, 0, 1)
for (p in 2:200)
ytrend[p]     <-  0.2*time[p] + noise[p]
plot(ts(ytrend))
```




```
par(mfrow = c(1,2))
plot(acf(ytrend, plot = F), main = "ACF for ytrend")
plot(pacf(ytrend, plot = F), main = "PACF for ytrend")
```



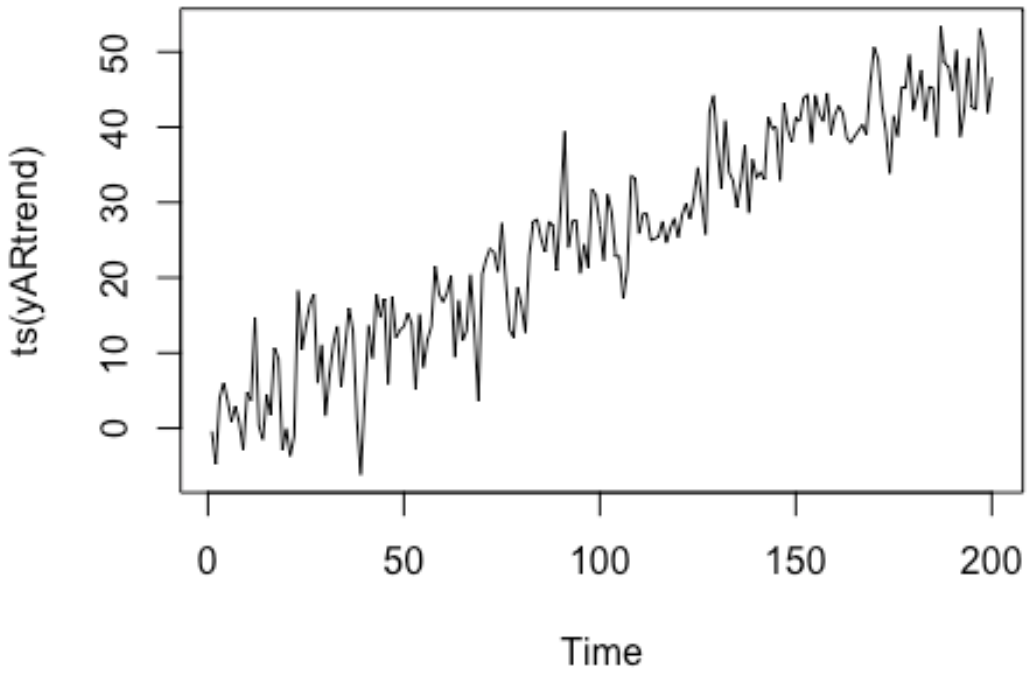

```
fitTrend <- ar(ytrend)

# examine order
fitTrend$order

## [1] 4

# examine coefficient estimates
fitTrend$ar

## [1] 0.4097025 0.1751033 0.2211417 0.1304655
```

We see that a linear trend may show up as AR effects.

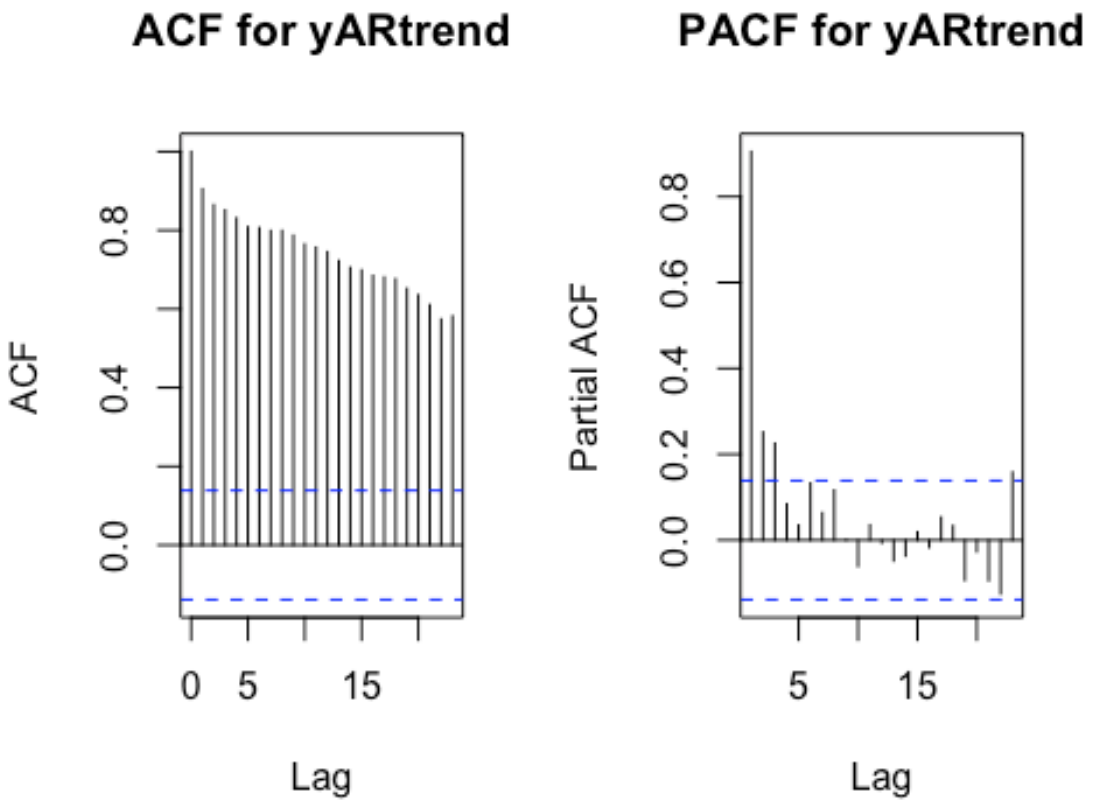
Now let's put them together - a linear trend and an AR(1) relation.

$$y(t) = 0.20 * y(t-1) + 0.20 * time(t) + \zeta(t)$$

```
# AR series + trend
yARtrend <-  rnorm(1, 0, 1)
for (p in 2:200)
yARtrend[p]     <- 0.20*yARtrend[p-1] +  0.2*time[p]+ noise[p]
plot(ts(yARtrend))
```




```
par(mfrow = c(1,2))
plot(acf(yARtrend, plot = F), main = "ACF for yARtrend")
plot(pacf(yARtrend, plot = F), main = "PACF for yARtrend")
```




```
fitARtrend <- ar(yARtrend)

# examine order
fitARtrend$order

## [1] 3

# examine coefficient estimates
fitARtrend$ar

## [1] 0.62051167 0.09927645 0.22509653
```

Again, we see inflated AR coefficients.

Fortunately, we can remove trends pretty easily. We can either include it during the model (much like in our data-generating equations above) or we can detrend prior to analysis. Let's detrend.

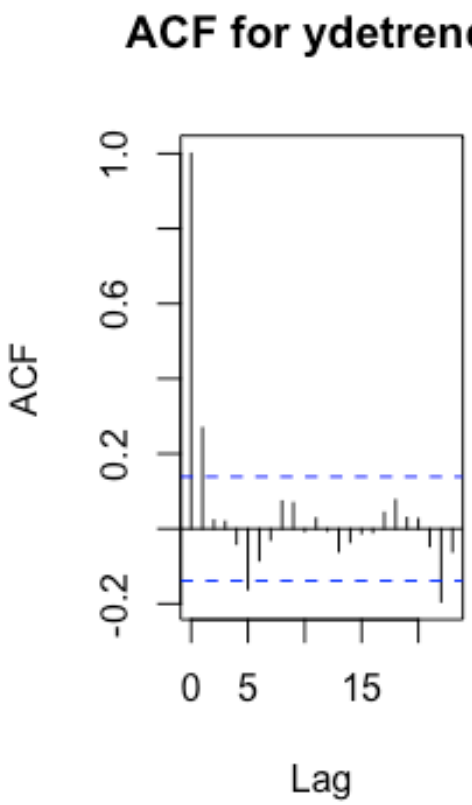
```
## estimating trends ##

fitc<- lm(yARtrend ~ time)  #run a simple regression predicting time
summary(fitc)
```

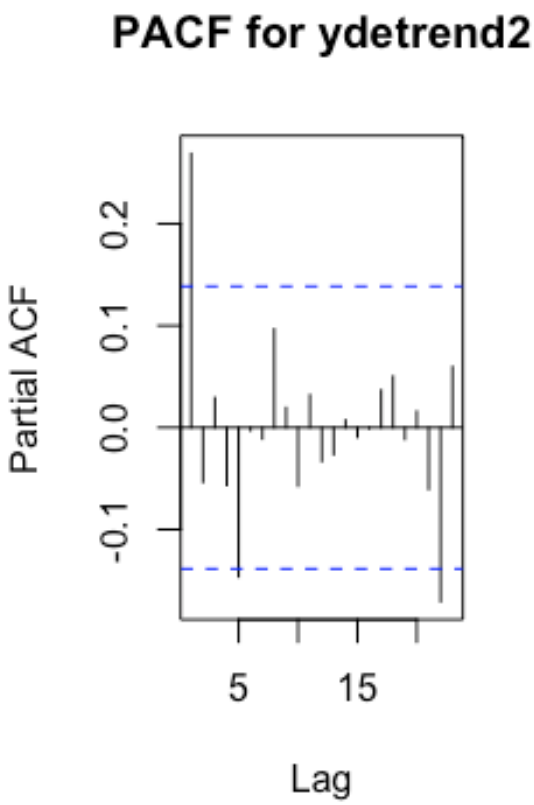
```
##
## Call:
## lm(formula = yARtrend ~ time)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -16.7784  -3.3305   0.0633   3.5687  16.1571
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 1.158418   0.712559   1.626    0.106
## time        0.242667   0.006148  39.472   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5.02 on 198 degrees of freedom
## Multiple R-squared:  0.8872, Adjusted R-squared:  0.8867
## F-statistic:  1558 on 1 and 198 DF,  p-value: < 2.2e-16
```

We can see here that trends exist for *yARtrend*. You can check on your own to ensure that the coefficient for the trend variable *time* was not signfiicant for the *yAR* data.

Now that we have identified the presence of a linear trend we can remove it.

```
##### detrend the yARtrend data - the easy way ####
ydetrend1 <- detrend(yARtrend, tt = 'linear')

# If we do the below we can see exactly what is being done.
# Remember that fitc was the regression of the yARtrend variable on the time
vector.
ypredict <- fitc$coefficients[1] + fitc$coefficients[2]*time # predict Y usin
g coefficients
ydetrend2 <- yARtrend - ypredict # subtract the predicted values from observe
d to obtain the residuals. This is the detrended data.

# This is what is done in the detrend function, and is perfectly correlated w
ith ydetrend1

# now look at results:
par(mfrow = c(1,2))
plot(acf(ydetrend2, plot = F), main = "ACF for ydetrend2")
plot(pacf(ydetrend2, plot = F), main = "PACF for ydetrend2")
```






```
ar(ydetrend2) # AR(1) selected
```

```
##
## Call:
## ar(x = ydetrend2)
##
## Coefficients:
##      1
## 0.2688
##
## Order selected 1  sigma^2 estimated as  23.38
```

Now we obtain the AR(1) coefficient used to generate the data.

Testing for stability and stationarity.

It is common to also test for stability and stationarity. The former helps to identify if the data will "blow up" if taken to T = infinity.

```
adf.test(yARtrend,  alternative = "stationary")
```

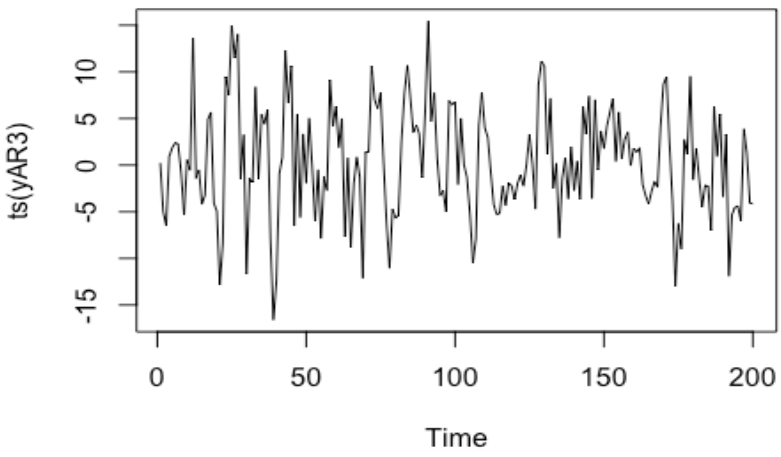
```
## Warning in adf.test(yARtrend, alternative = "stationary"): p-value smaller
## than printed p-value

##
##  Augmented Dickey-Fuller Test
##
## data:  yARtrend
## Dickey-Fuller = -6.1662, Lag order = 5, p-value = 0.01
## alternative hypothesis: stationary
```

The ADF test found that the null hypothesis that the data are not stable is rejected. The alternative hypothesis, that the data are stationary, was supported. This highlights the need to explore linear trends in the data where extreme values are unlikely.

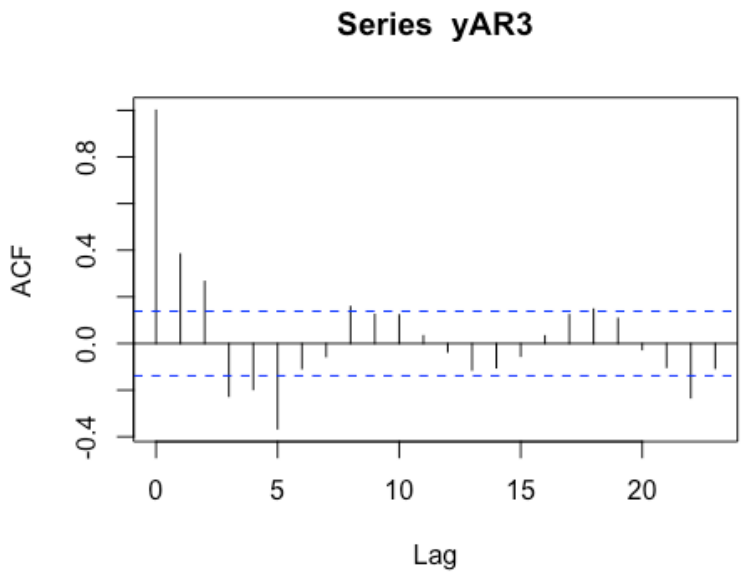
Testing residuals.

Now let's simulate a new series with greater lags - an AR(3) process. We'll fit an

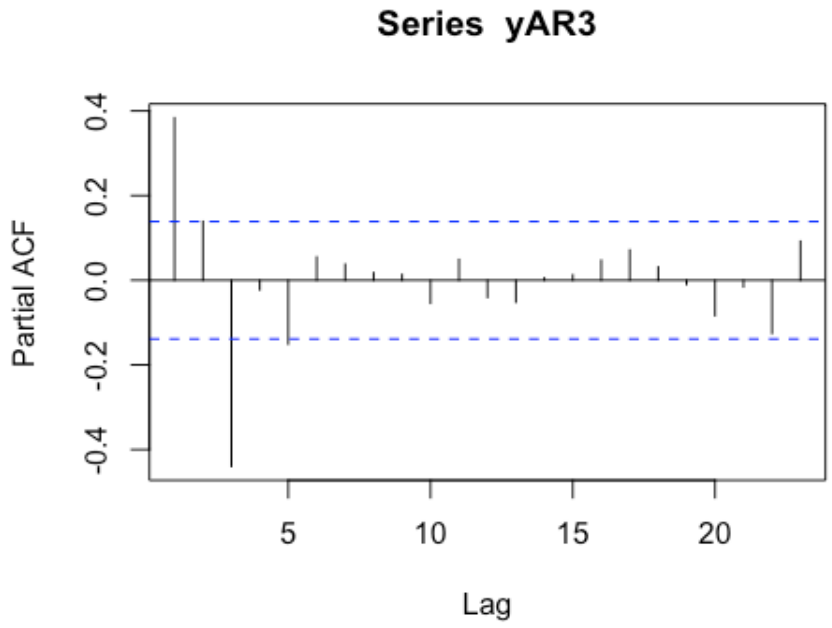
```
# AR(3) series
yAR3 <-  rnorm(1, 0, 1)
yAR3[2] <- 0.3*yAR3[1]+noise[2]
yAR3[3] <- 0.3*yAR3[1] + 0.3*yAR3[2] +noise[2]
for (p in 4:200)
yAR3[p]     <- 0.30*yAR3[p-1] + 0.30*yAR3[p-2] - 0.40*yAR3[p-3] + noise[p]
plot(ts(yAR3))
```



```r
acf(yAR3)
```




```r
pacf(yAR3)
```




```r
fitAR3 <- ar(yAR3)

Box.test(fitAR3$resid, lag = 5, type = c("Box-Pierce"), fitdf = 2)
```

```
##
##  Box-Pierce test
```

```
##
## data:  fitAR3$resid
## X-squared = 0.32229, df = 3, p-value = 0.9558
```

When the correct lag order is used in the model the residuals pass the Box-Pierce test. They are not correlated across time.

Let's see what happens when we only estimate an AR(2).

```
fitAR2 <- ar(yAR3, order.max = 2)
Box.test(fitAR2$resid, lag = 5, type = c("Box-Pierce"), fitdf = 2)

##
##  Box-Pierce test
##
## data:  fitAR2$resid
## X-squared = 53.725, df = 3, p-value = 1.284e-11

Box.test(fitAR2$resid, lag=20,type = "Ljung-Box") # portmanteu test

##
##  Box-Ljung test
##
## data:  fitAR2$resid
## X-squared = 79.037, df = 20, p-value = 5.717e-09
```

We see that the tests were significant - the errors are correlated.