06 ARIMA Models in R

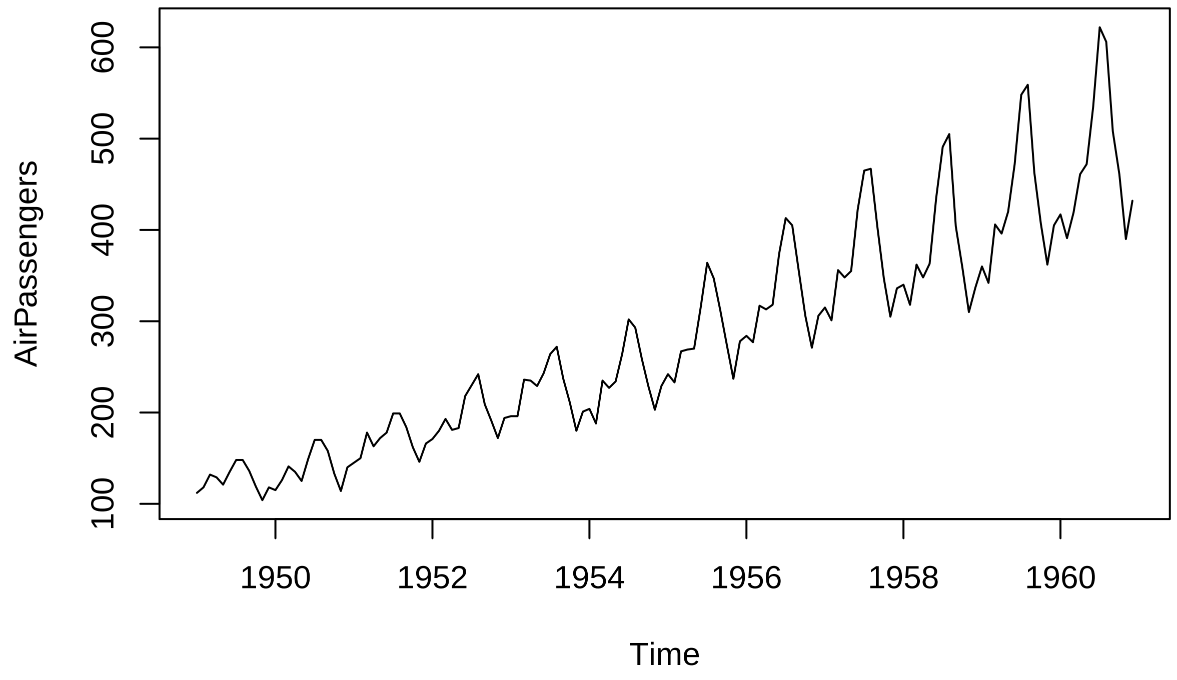
Boni

9/6/2020

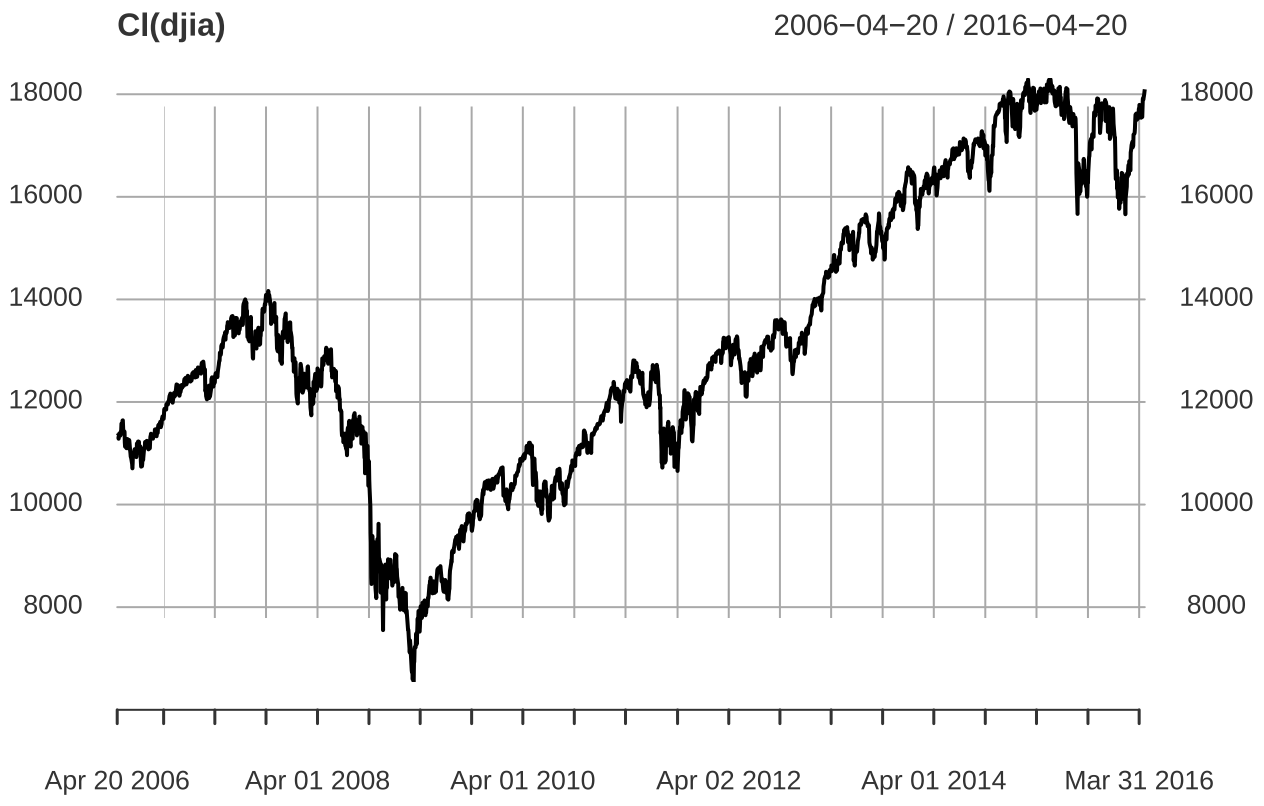
Time series data and models

The AirPassengers data show a handful of important qualities, including seasonality, trend, and heteroscedasticity, which distinguish the data from standard white noise.

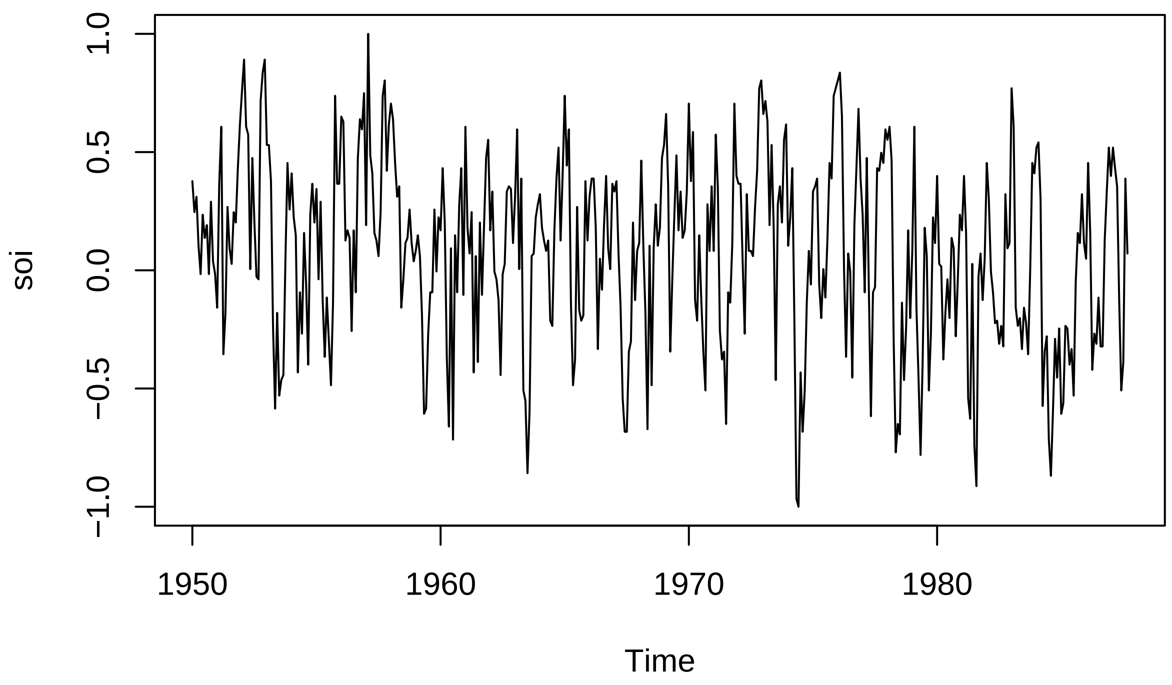
```r
# Plot AirPassengers
plot(AirPassengers)
```



```r
# Plot the DJIA daily closings
plot(Cl(djia))
```



```r
# Plot the Southern Oscillation Index
plot(soi)
```



Stationarity and nonstationarity

A time series is Stationary when it is “stable” meaning:

- the mean is constant over time (no trend)
- the correlation structure is constant over time.

Random walk trend:

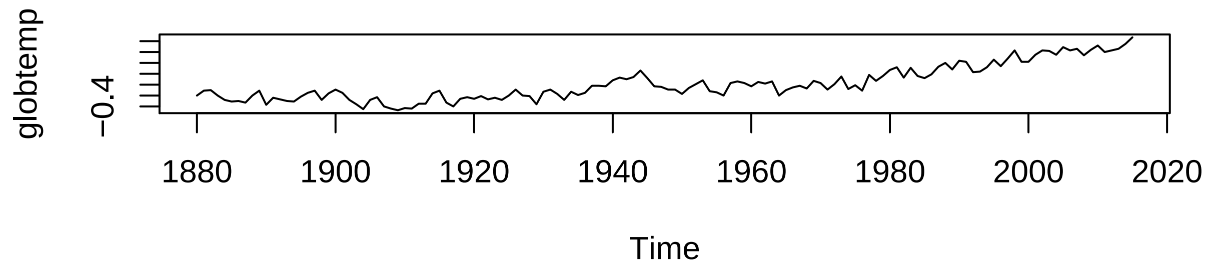
- Not stationary, but differences in lag of data are stationary.

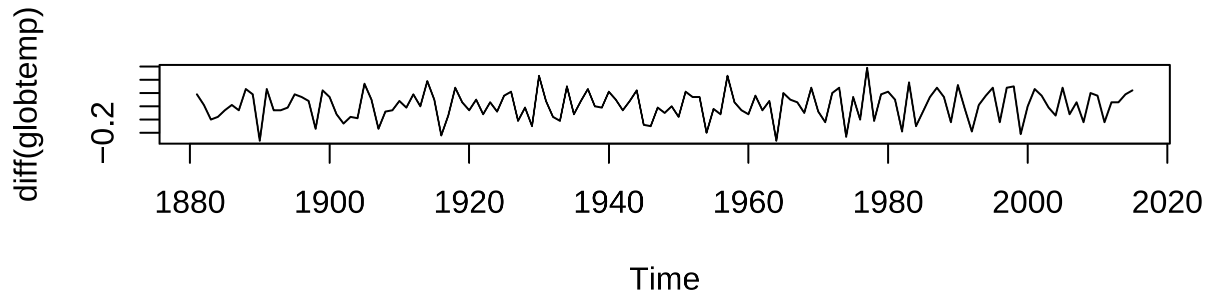
Trend stationary:

- Stationary with trend (increase/decrease)

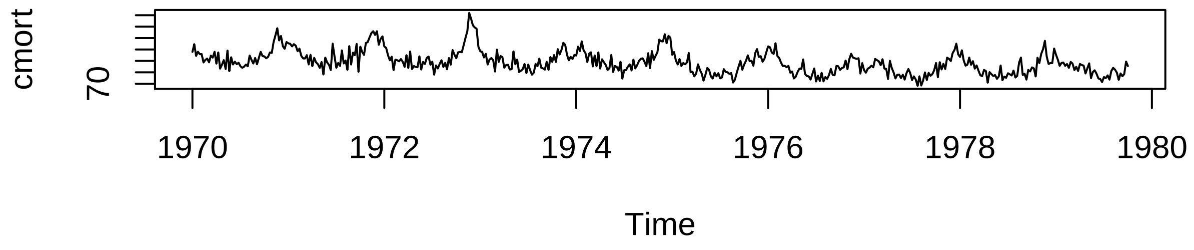
Detrending the data

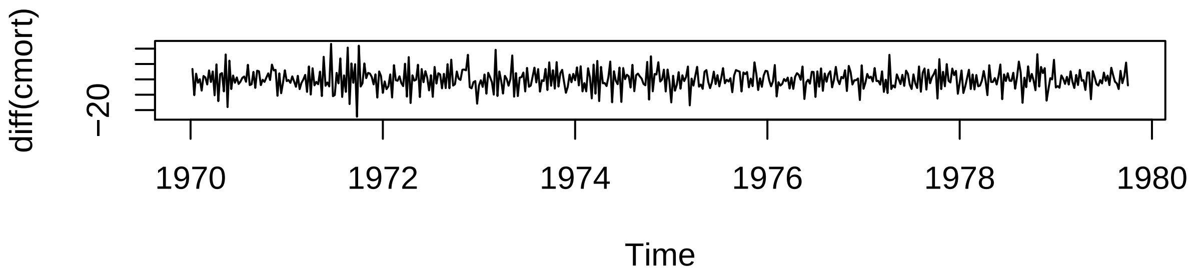
```r
# Plot globtemp and detrended globtemp
par(mfrow = c(2,1))
plot(globtemp)
plot(diff(globtemp))
```





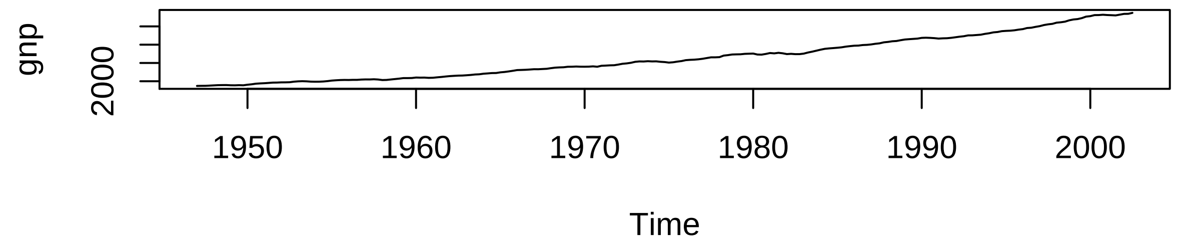
```r
# Plot cmort and detrended cmort
par(mfrow = c(2,1))
plot(cmort)
plot(diff(cmort))
```

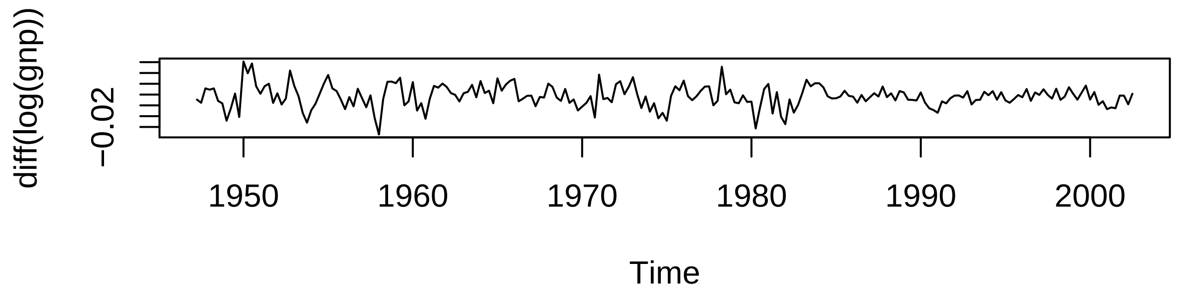




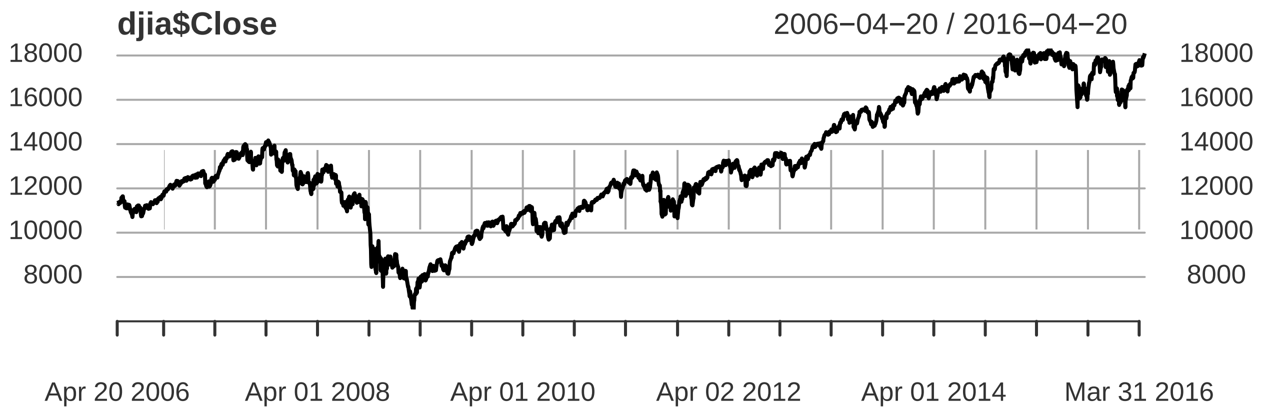
Dealing with trends and heterokedasticity

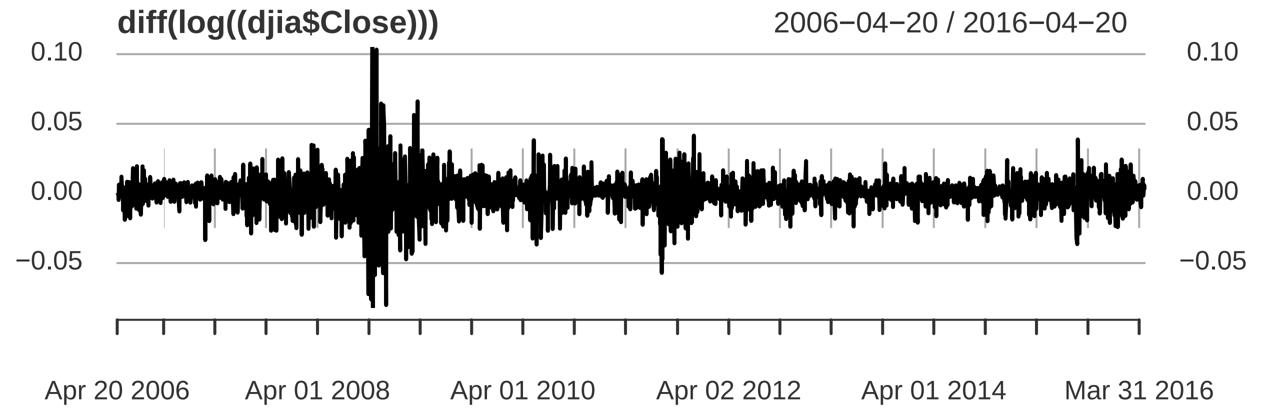
```
# astsa and xts are preloaded

# Plot GNP series (gnp) and its growth rate
par(mfrow = c(2,1))
plot(gnp)
plot(diff(log(gnp)))
```





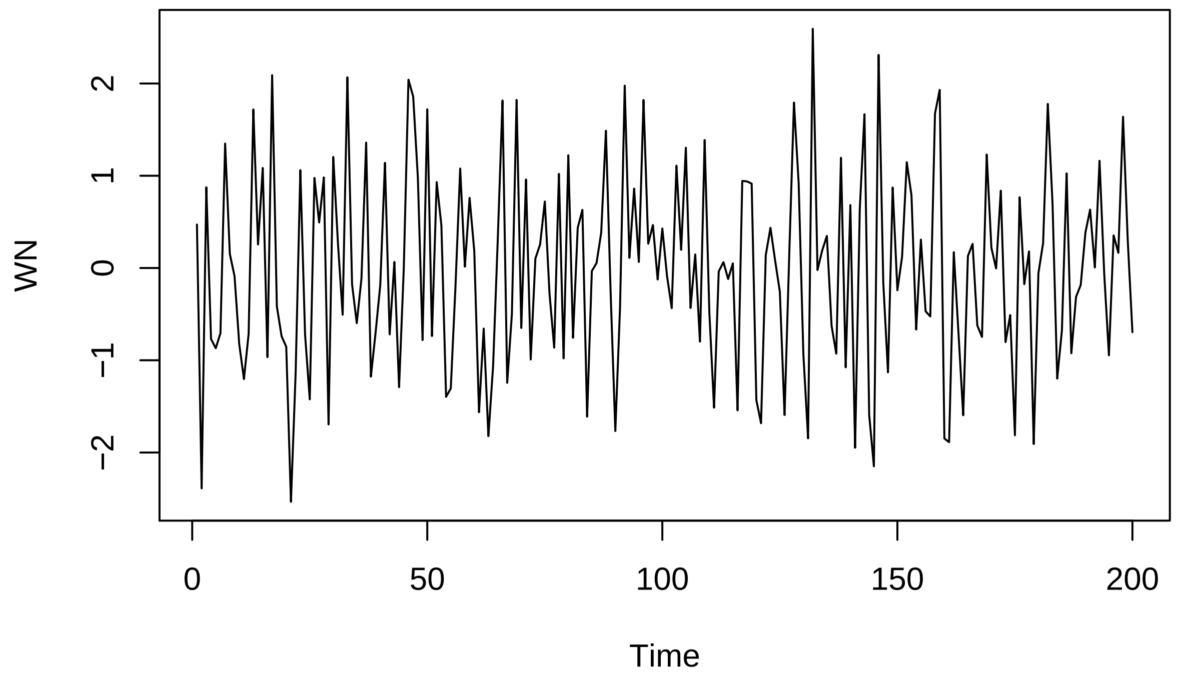
```r
# Plot the DJIA closings (djia$Close) and its returns
par(mfrow = c(2,1))
plot(djia$Close)
plot(diff(log((djia$Close))))
```



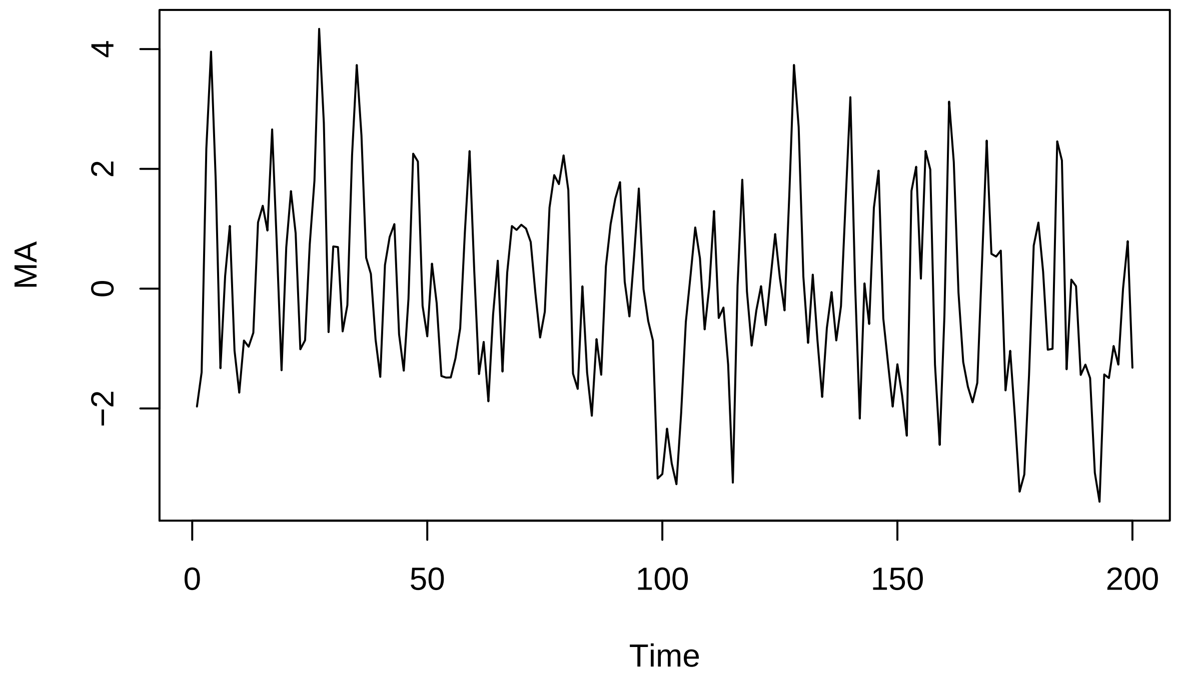


Stationary time series ARMA

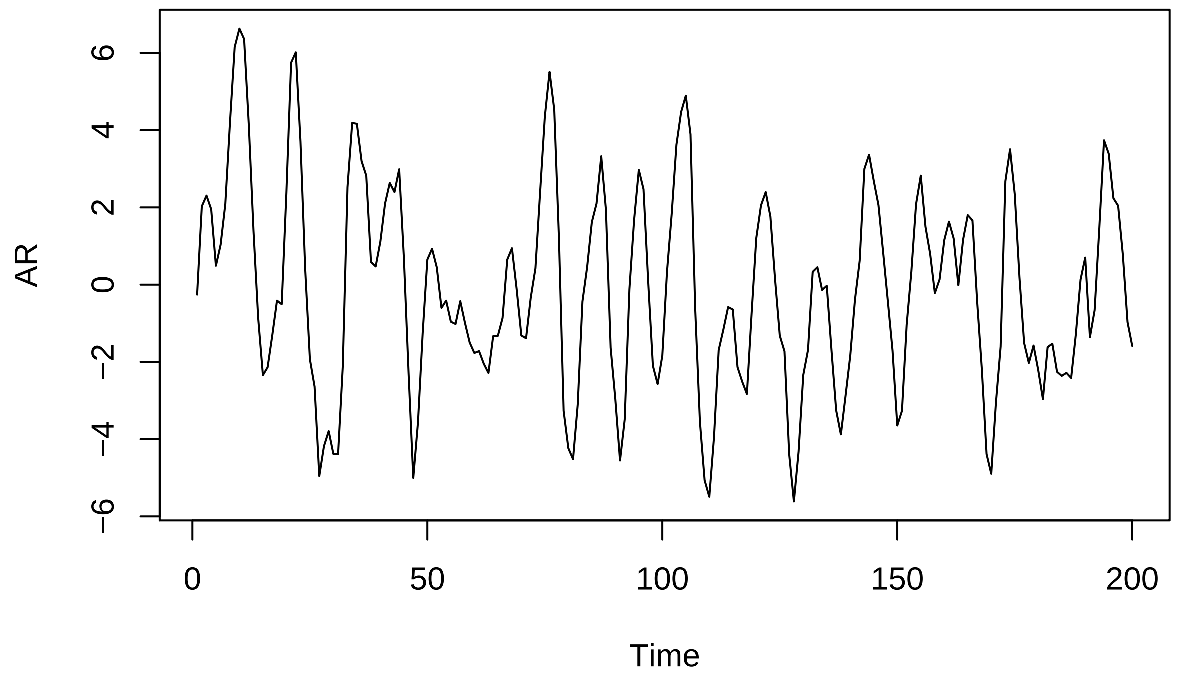
```
# Generate and plot white noise
WN <- arima.sim(model = list(order = c(0, 0, 0)), n = 200)
plot(WN)
```



```r
# Generate and plot an MA(1) with parameter .9 by filtering the noise
MA <- arima.sim(model = list(order = c(0, 0, 1), ma = .9), n = 200)
plot(MA)
```



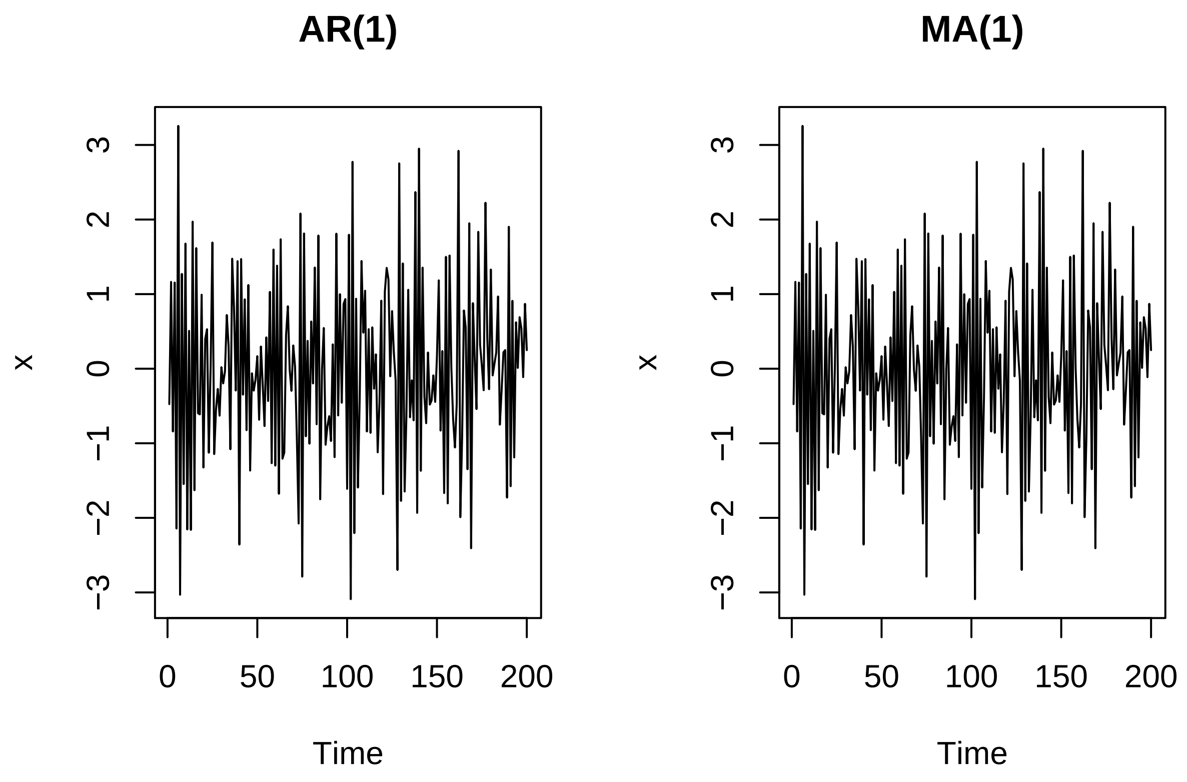
```
# Generate and plot an AR(1) with parameters 1.5 and -.75
AR <- arima.sim(model = list(order = c(2, 0, 0), ar = c(1.5, -.75)), n = 200)
plot(AR)
```



Fitting ARMA Model

AR and MA Models

```
x <- arima.sim(list(order = c(1, 0, 0), ar = -.7), n = 200)
y <- arima.sim(list(order = c(0, 0, 1), ma = -.7), n = 200)
par(mfrow = c(1, 2))
plot(x, main = "AR(1)")
plot(x, main = "MA(1)")
```



By looking at the graph we can not identify the model, we need tools like ACF (Auto Correlation Function) and PACF (Partial Auto Correlation Function) in R.

Fitting an AR(1) model

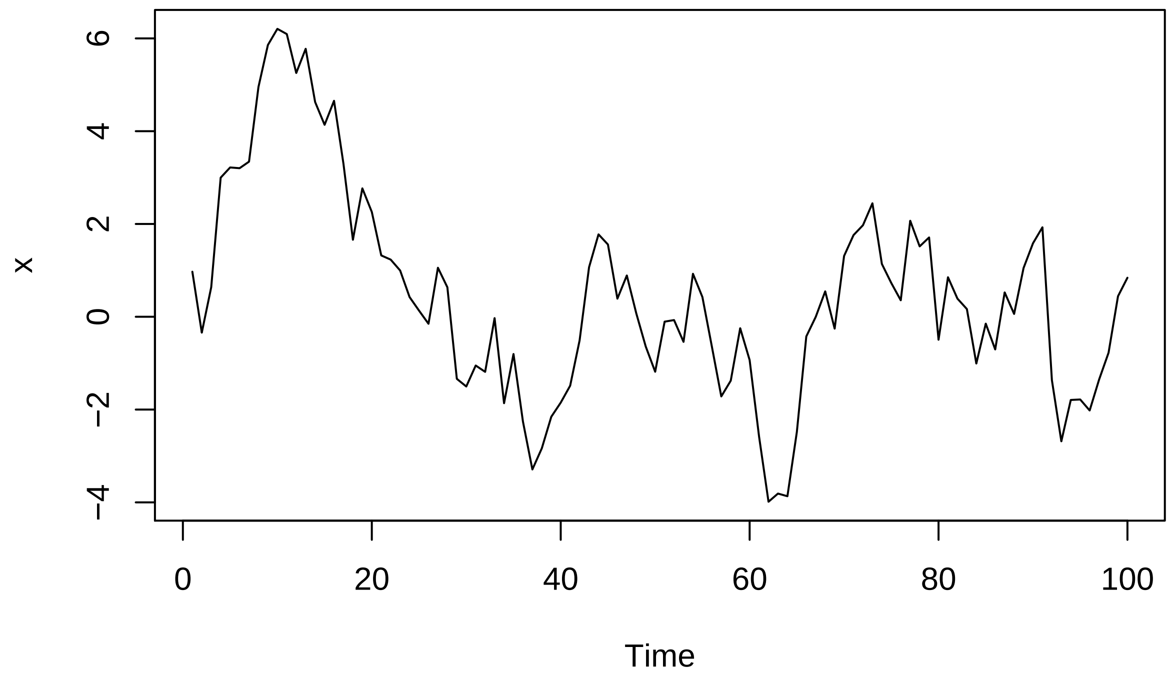
In this exercise, you will generate data from the AR(1) model,

$X_t = .9X_{(t-1)} + w_t$

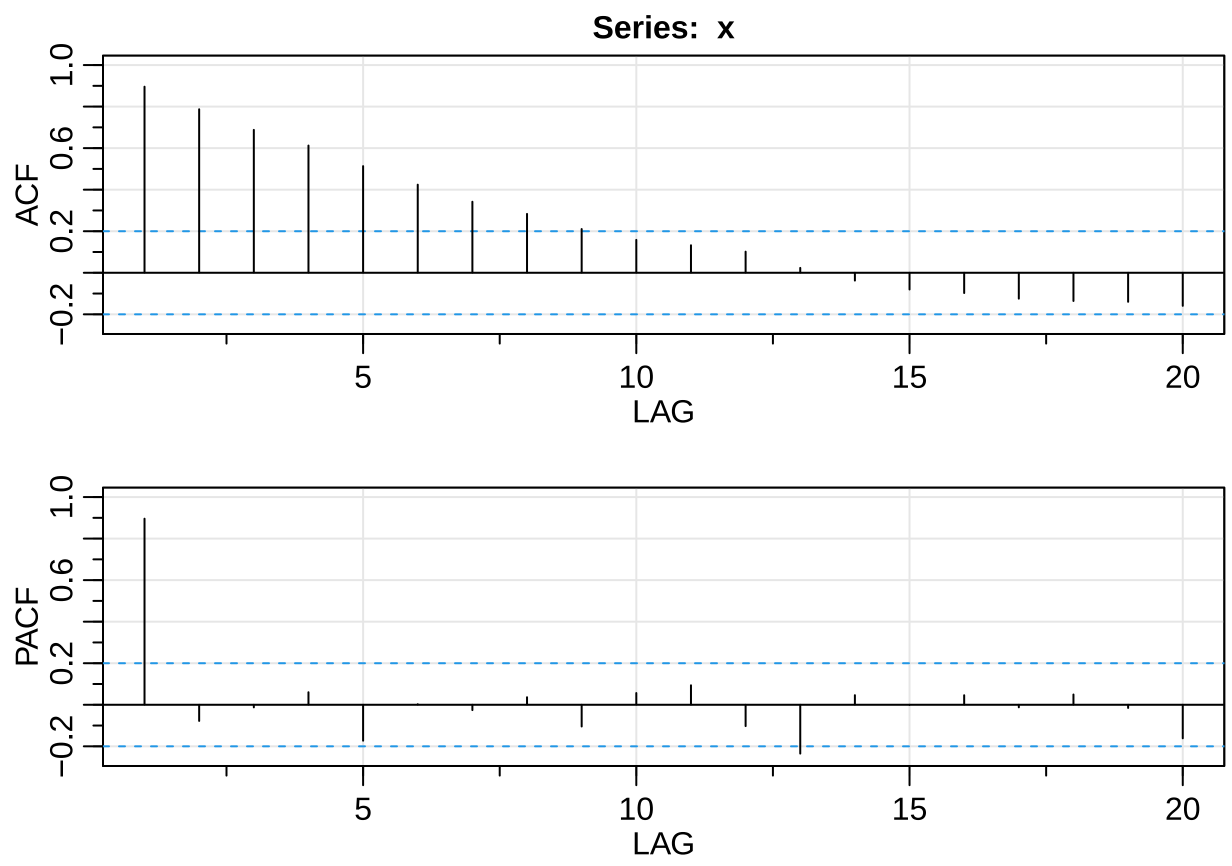
look at the simulated data and the sample ACF and PACF pair to determine the order. Then, you will fit the model and compare the estimated parameters to the true parameters.

Throughout this course, you will be using sarima() from the astsa package to easily fit models to data. The command produces a residual diagnostic graphic that can be ignored until diagnostics is discussed later in the chapter.

```r
# Generate 100 observations from the AR(1) model
x <- arima.sim(model = list(order = c(1, 0, 0), ar = .9), n = 100)

# Plot the generated data
plot(x)
```



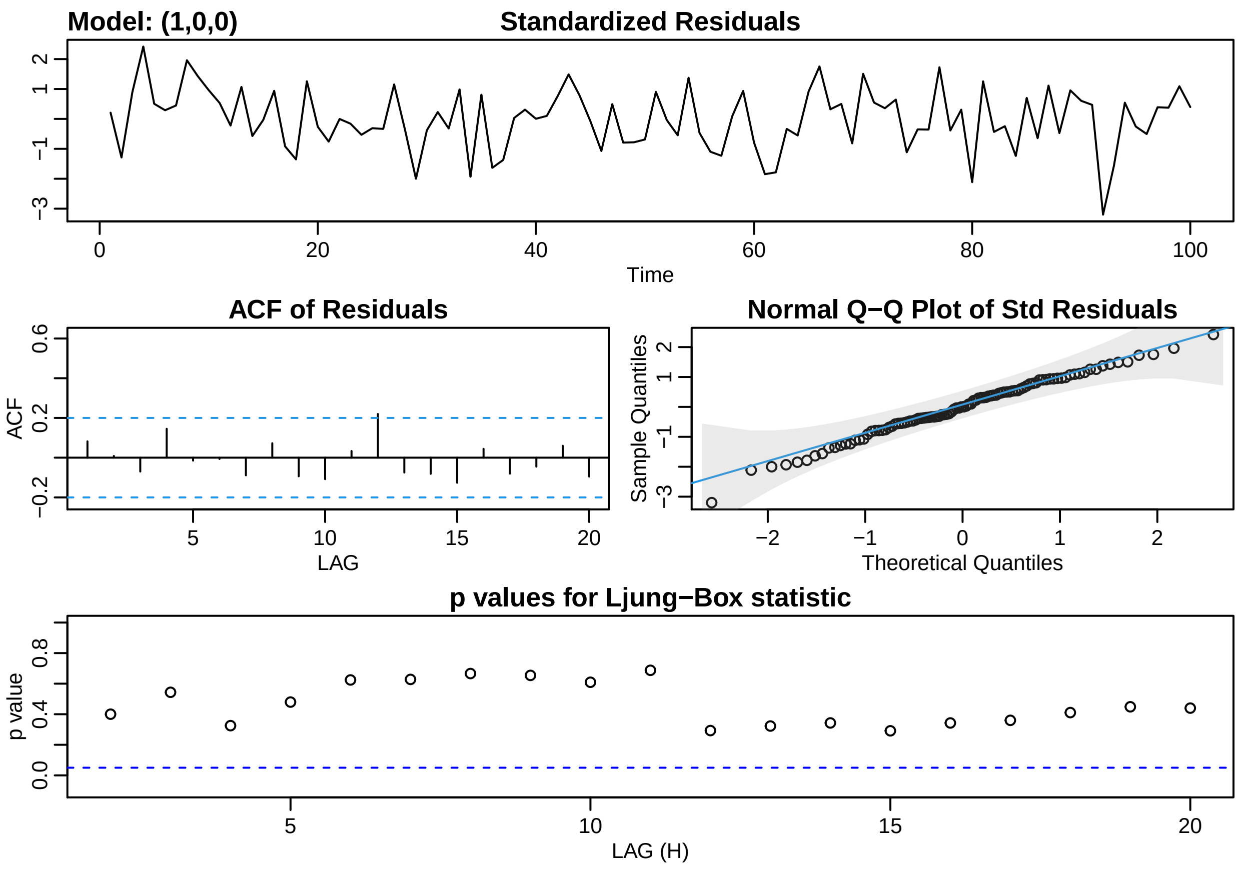
```r
# Plot the sample P/ACF pair
acf2(x)
```



```
##      [,1]  [,2]  [,3] [,4]  [,5] [,6]  [,7] [,8]  [,9] [,10] [,11] [,12] [,13]
## ACF   0.9  0.79  0.69 0.61  0.51 0.42  0.34 0.28  0.21  0.16  0.13   0.1  0.02
## PACF  0.9 -0.08 -0.01 0.06 -0.17 0.00 -0.03 0.04 -0.10  0.06  0.09  -0.1 -0.24
##      [,14] [,15] [,16] [,17] [,18] [,19] [,20]
## ACF  -0.04 -0.08 -0.10 -0.12 -0.14 -0.14 -0.16
## PACF  0.05  0.00  0.05 -0.01  0.05 -0.02 -0.16
```

```
# Fit an AR(1) to the data and examine the t-table
sarima(x, p = 1, d = 0, q = 0)
```

```
## initial  value 0.798201
## iter   2 value -0.014715
## iter   3 value -0.014715
## iter   4 value -0.014715
## iter   5 value -0.014716
## iter   6 value -0.014716
## iter   6 value -0.014716
## final  value -0.014716
## converged
## initial  value -0.011337
## iter   2 value -0.011407
## iter   3 value -0.011530
## iter   4 value -0.011534
## iter   5 value -0.011538
## iter   6 value -0.011546
## iter   7 value -0.011547
## iter   8 value -0.011548
## iter   9 value -0.011548
## iter  10 value -0.011549
## iter  11 value -0.011549
## iter  11 value -0.011549
## final  value -0.011549
## converged
```



```
## $fit
##
## Call:
## stats::arima(x = xdata, order = c(p, d, q), seasonal = list(order = c(P, D,
##     Q), period = S), xreg = xmean, include.mean = FALSE, transform.pars = trans,
##     fixed = fixed, optim.control = list(trace = trc, REPORT = 1, reltol = tol))
##
## Coefficients:
##          ar1   xmean
##       0.8883  0.5262
## s.e.  0.0427  0.8160
##
## sigma^2 estimated as 0.9621:  log likelihood = -140.74,  aic = 287.48
##
## $degrees_of_freedom
## [1] 98
```

```
##
## $ttable
##       Estimate     SE t.value p.value
## ar1     0.8883 0.0427 20.8068  0.0000
## xmean   0.5262 0.8160  0.6448  0.5206
##
## $AIC
## [1] 2.87478
##
## $AICc
## [1] 2.876017
##
## $BIC
## [1] 2.952935
```

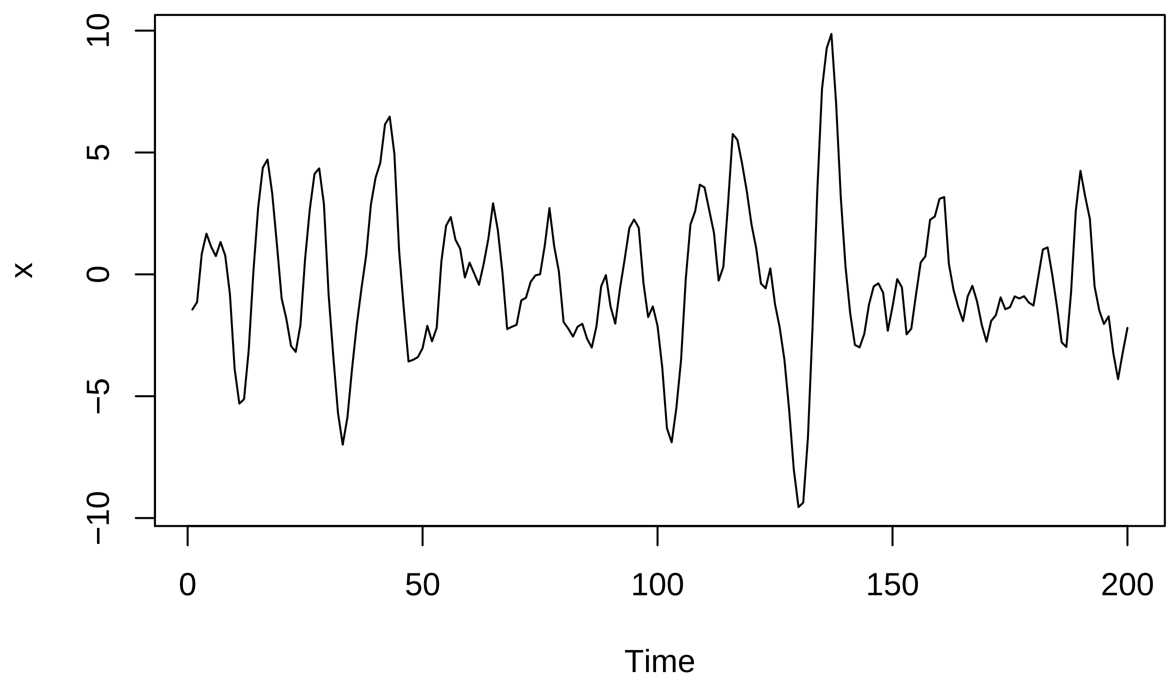
Fitting an AR(2) model

For this exercise, we generated data from the AR(2) model,

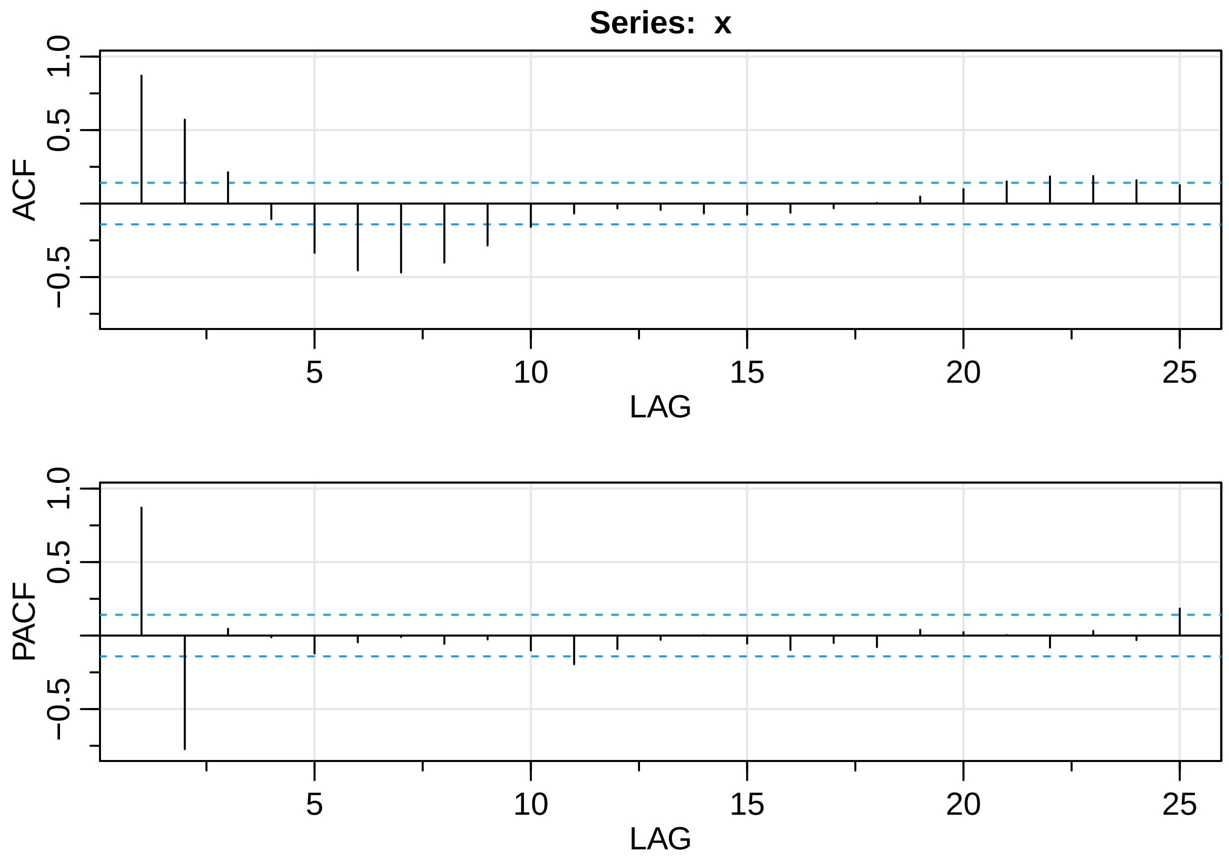
$X_{(t=1)} = 1.5X_{(t-1)} - 1.75X_{(t-2)} + W_t$

using x <- arima.sim(model = list(order = c(2, 0, 0), ar = c(1.5, -.75)), n = 200). Look at the simulated data and the sample ACF and PACF pair to determine the model order. Then fit the model and compare the estimated parameters to the true parameters.

```r
# astsa is preloaded
x <- arima.sim(model = list(order = c(2, 0, 0), ar = c(1.5, -.75)), n = 200)

# Plot x
plot(x)
```



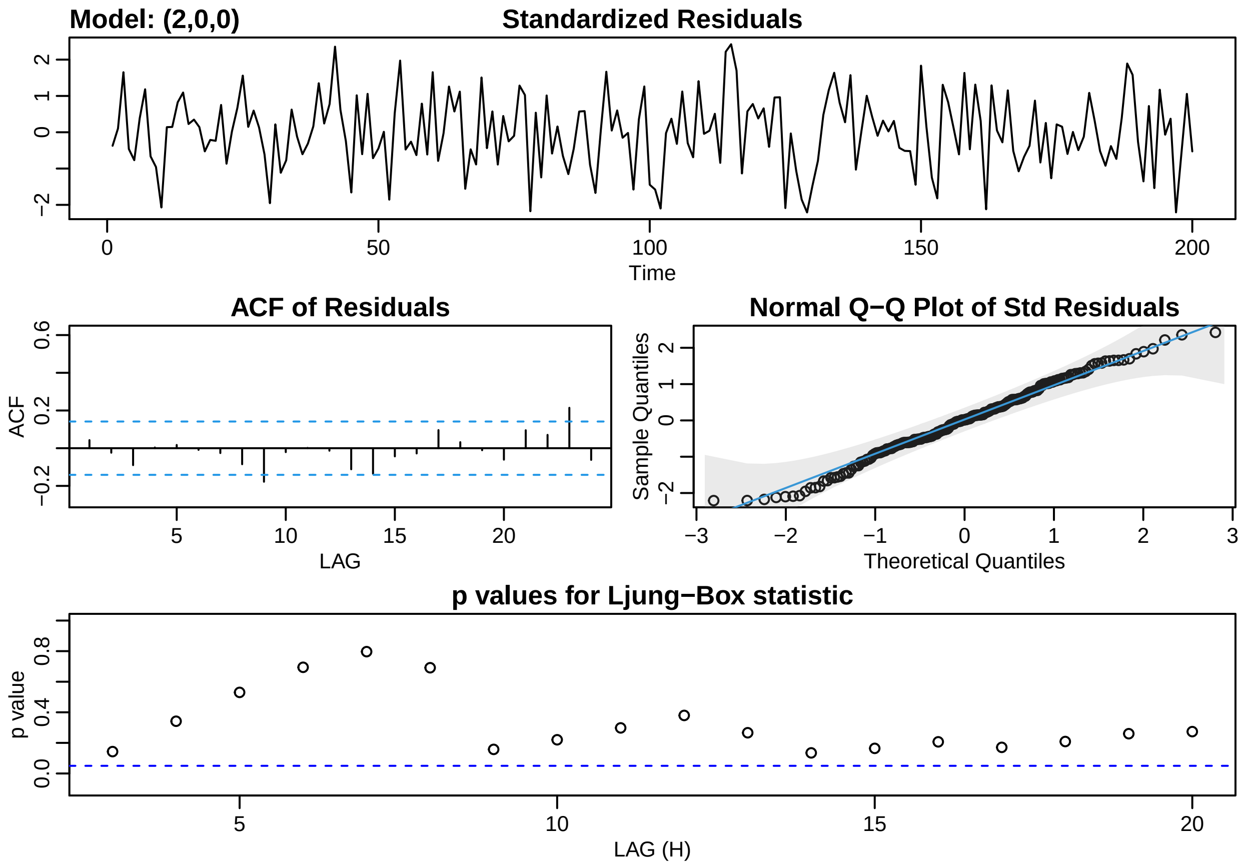
```
# Plot the sample P/ACF of x
acf2(x)
```



```
##       [,1]  [,2] [,3]  [,4]  [,5]  [,6]  [,7]  [,8]  [,9] [,10] [,11] [,12]
## ACF  0.87  0.57 0.21 -0.11 -0.34 -0.46 -0.47 -0.40 -0.29 -0.16 -0.07 -0.03
## PACF 0.87 -0.77 0.05 -0.01 -0.12 -0.05 -0.01 -0.06 -0.03 -0.10 -0.20 -0.09
##      [,13] [,14] [,15] [,16] [,17] [,18] [,19] [,20] [,21] [,22] [,23] [,24]
## ACF  -0.04 -0.07 -0.08 -0.06 -0.03  0.00  0.05  0.10  0.15  0.19  0.19  0.16
## PACF -0.03  0.00 -0.05 -0.10 -0.05 -0.08  0.04  0.02  0.00 -0.08  0.03 -0.03
##      [,25]
## ACF   0.13
## PACF  0.18
```

```
# Fit an AR(2) to the data and examine the t-table
sarima(x, p = 2, d = 0, q = 0)
```

```
## initial  value 1.126446
## iter   2 value 1.000218
## iter   3 value 0.551978
## iter   4 value 0.295617
## iter   5 value 0.155151
## iter   6 value -0.049989
## iter   7 value -0.050991
## iter   8 value -0.051258
## iter   9 value -0.051258
## iter  10 value -0.051258
## iter  11 value -0.051259
## iter  12 value -0.051259
## iter  12 value -0.051259
## iter  12 value -0.051259
## final  value -0.051259
## converged
## initial  value -0.047678
## iter   2 value -0.047722
## iter   3 value -0.047749
## iter   4 value -0.047750
## iter   5 value -0.047750
## iter   6 value -0.047750
## iter   7 value -0.047750
## iter   7 value -0.047750
## iter   7 value -0.047750
## final  value -0.047750
## converged
```



```
## $fit
##
## Call:
## stats::arima(x = xdata, order = c(p, d, q), seasonal = list(order = c(P, D,
##     Q), period = S), xreg = xmean, include.mean = FALSE, transform.pars = trans,
##     fixed = fixed, optim.control = list(trace = trc, REPORT = 1, reltol = tol))
##
## Coefficients:
##          ar1      ar2    xmean
##       1.5436  -0.7723  -0.3148
## s.e.  0.0439   0.0438   0.2925
##
## sigma^2 estimated as 0.8943:  log likelihood = -274.24,  aic = 556.48
##
## $degrees_of_freedom
## [1] 197
##
## $ttable
##       Estimate     SE  t.value p.value
## ar1     1.5436 0.0439  35.1789  0.0000
## ar2    -0.7723 0.0438 -17.6306  0.0000
## xmean  -0.3148 0.2925  -1.0762  0.2831
##
## $AIC
## [1] 2.782378
##
## $AICc
## [1] 2.78299
##
## $BIC
## [1] 2.848344
```

Fitting an MA(1) model

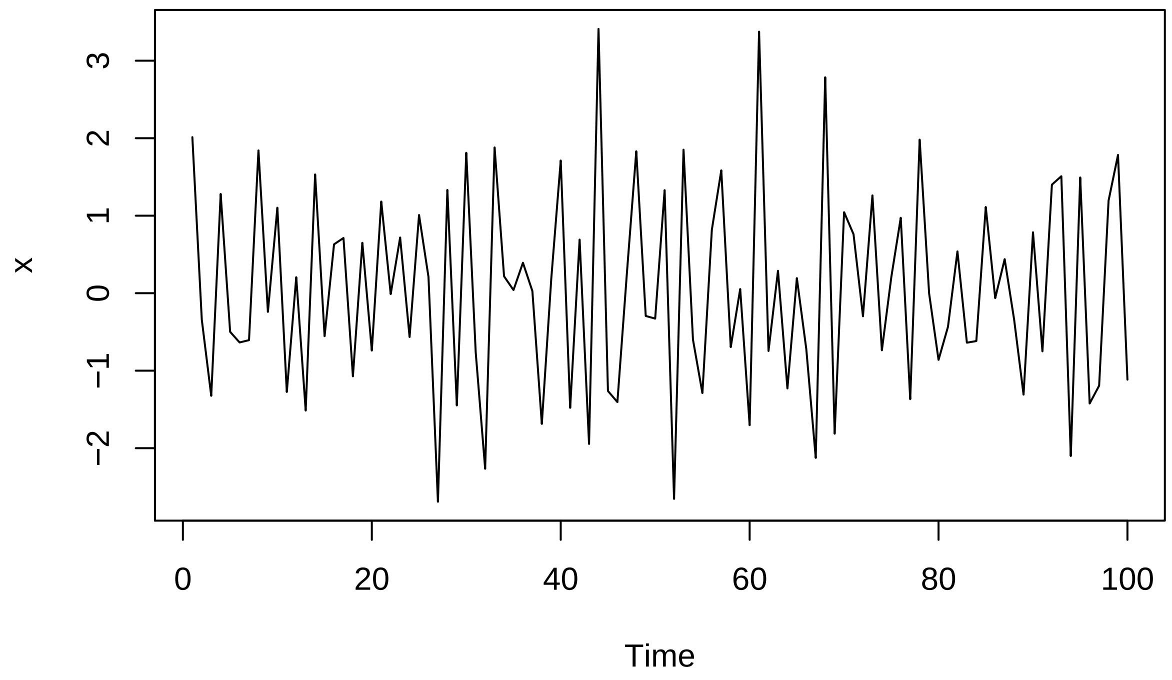
In this exercise, we generated data from an MA(1) model,

$X_t = W_t - .8w_{t-1}$

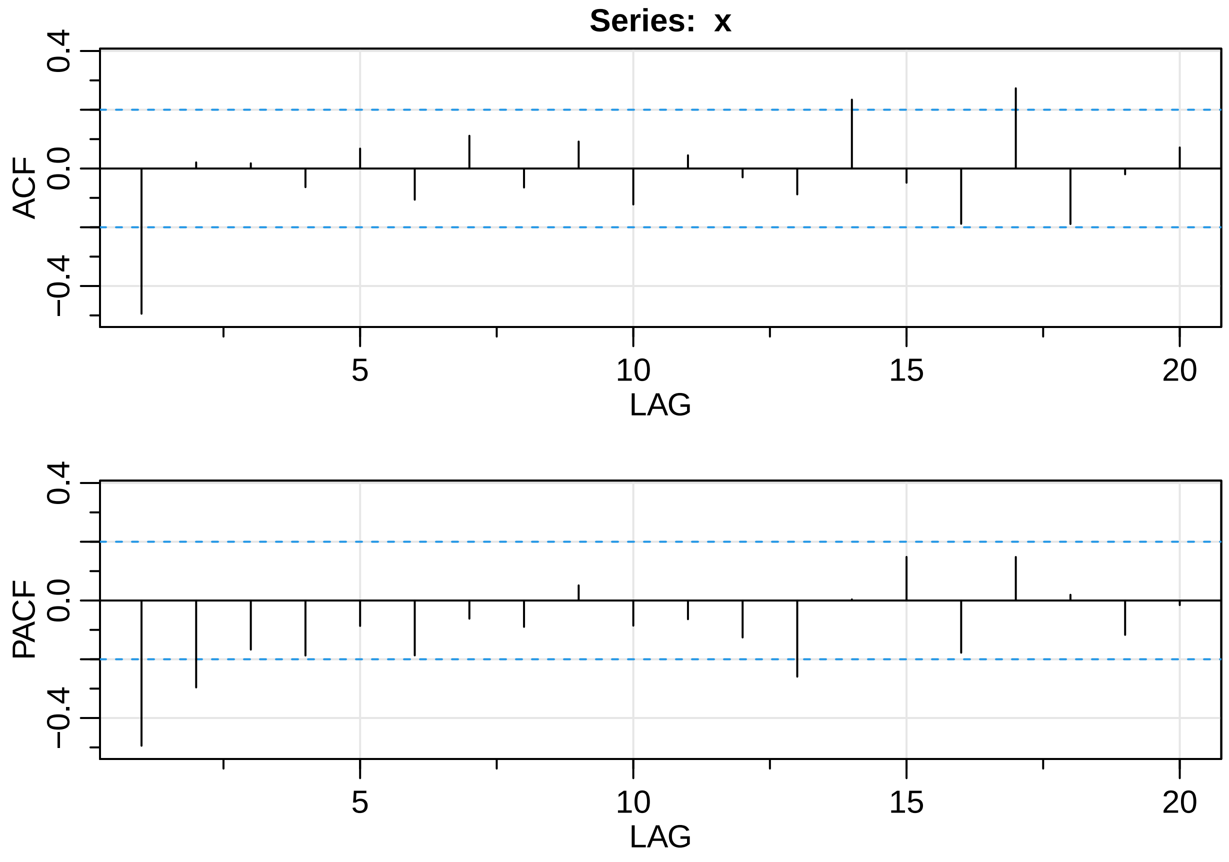
x <- arima.sim(model = list(order = c(0, 0, 1), ma = -.8), n = 100). Look at the simulated data and the sample ACF and PACF to determine the order based on the table given in the first exercise. Then fit the model.

Recall that for pure MA(q) models, the theoretical ACF will cut off at lag q while the PACF will tail off.

```r
# astsa is preloaded
x <- arima.sim(model = list(order = c(0, 0, 1), ma = -.8), n = 100)

# Plot x
plot(x)
```



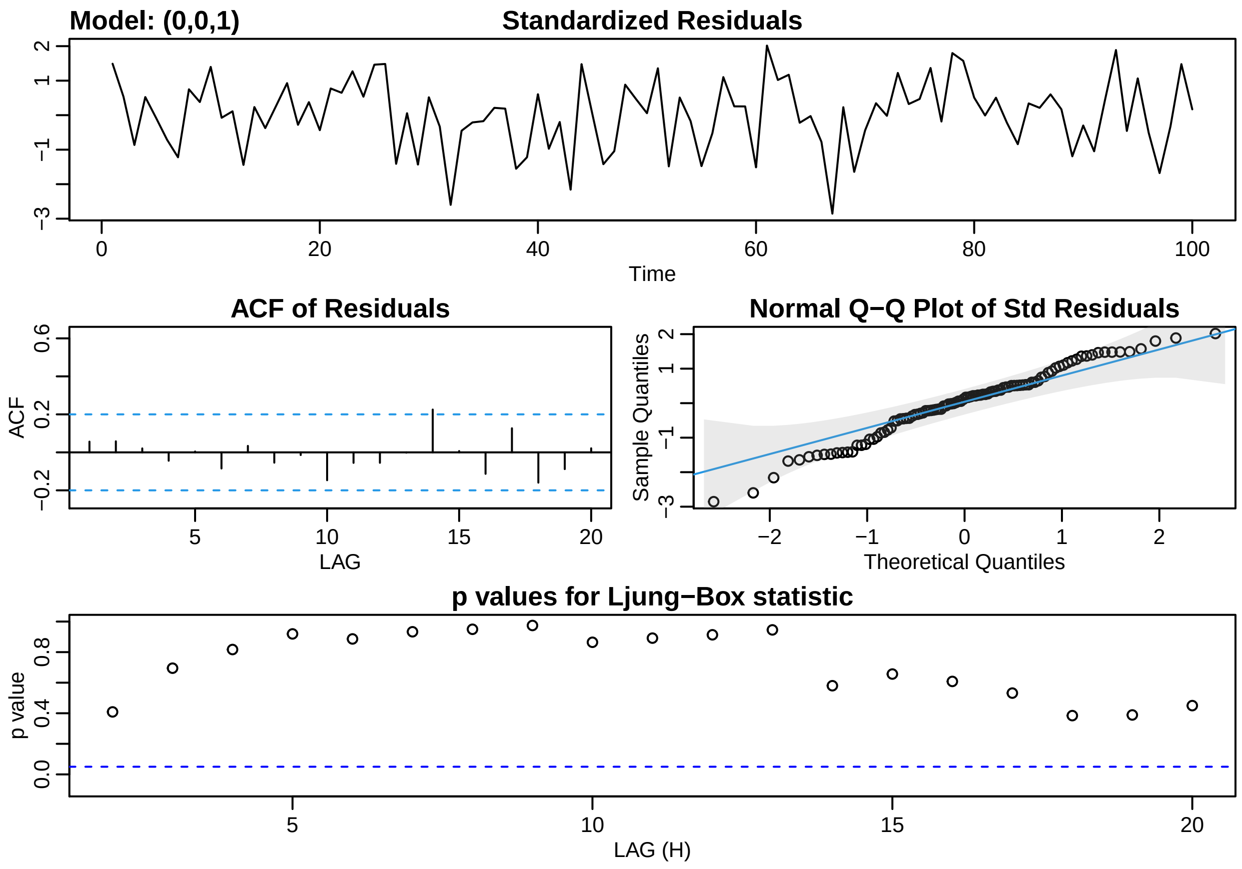
```
# Plot the sample P/ACF of x
acf2(x)
```



```
##       [,1]  [,2]  [,3]  [,4]  [,5]  [,6]  [,7]  [,8] [,9] [,10] [,11] [,12]
## ACF  -0.49  0.02  0.02 -0.06  0.07 -0.11  0.11 -0.06 0.09 -0.12  0.04 -0.03
## PACF -0.49 -0.30 -0.17 -0.19 -0.09 -0.19 -0.06 -0.09 0.05 -0.09 -0.06 -0.13
##      [,13] [,14] [,15] [,16] [,17] [,18] [,19] [,20]
## ACF  -0.09  0.23 -0.05 -0.19  0.27 -0.19 -0.02  0.07
## PACF -0.26  0.00  0.15 -0.18  0.15  0.02 -0.12 -0.02
```

```
# Fit an MA(1) to the data and examine the t-table
sarima(x, p = 0, d = 0, q = 1)
```

```
## initial  value 0.261771
## iter   2 value 0.075047
## iter   3 value 0.045818
## iter   4 value 0.039403
## iter   5 value 0.018146
## iter   6 value 0.015311
## iter   7 value 0.015157
## iter   8 value 0.015017
## iter   9 value 0.014948
## iter  10 value 0.014948
## iter  11 value 0.014948
## iter  11 value 0.014948
## final  value 0.014948
## converged
## initial  value 0.005703
## iter   2 value 0.005347
## iter   3 value 0.004040
## iter   4 value 0.003335
## iter   5 value 0.003148
## iter   6 value 0.003132
## iter   7 value 0.003131
## iter   8 value 0.003130
## iter   9 value 0.003130
## iter   9 value 0.003130
## iter   9 value 0.003130
## final  value 0.003130
## converged
```



```
## $fit
##
## Call:
## stats::arima(x = xdata, order = c(p, d, q), seasonal = list(order = c(P, D,
##     Q), period = S), xreg = xmean, include.mean = FALSE, transform.pars = trans,
##     fixed = fixed, optim.control = list(trace = trc, REPORT = 1, reltol = tol))
##
## Coefficients:
##           ma1   xmean
##       -0.8902  0.0271
## s.e.   0.0670  0.0120
##
```

```
## sigma^2 estimated as 0.9906:  log likelihood = -142.21,  aic = 290.41
##
## $degrees_of_freedom
## [1] 98
##
## $ttable
##       Estimate    SE  t.value p.value
## ma1    -0.8902 0.067 -13.2946  0.0000
## xmean   0.0271 0.012   2.2561  0.0263
##
## $AIC
## [1] 2.904137
##
## $AICc
## [1] 2.905374
##
## $BIC
## [1] 2.982292
```

Fitting an ARMA model

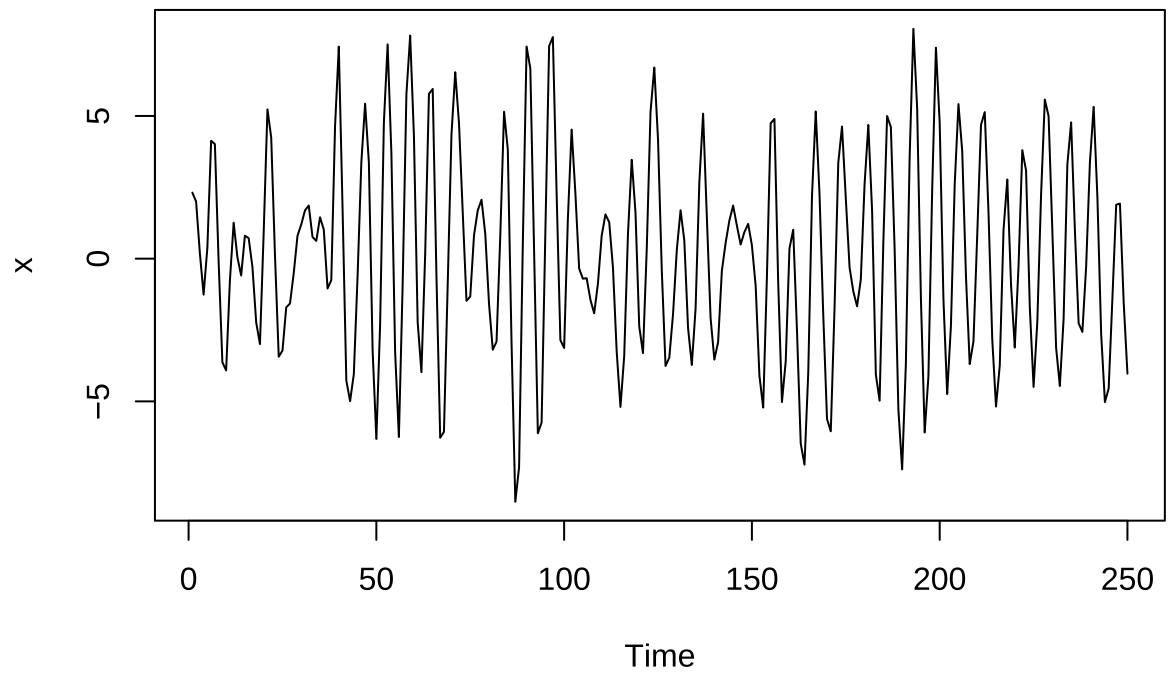
You are now ready to merge the AR model and the MA model into the ARMA model. We generated data from the ARMA(2,1) model,

$X_t = X_{t-1} - .9X_{t-2} + W_t + 0.8W_{t-1}$

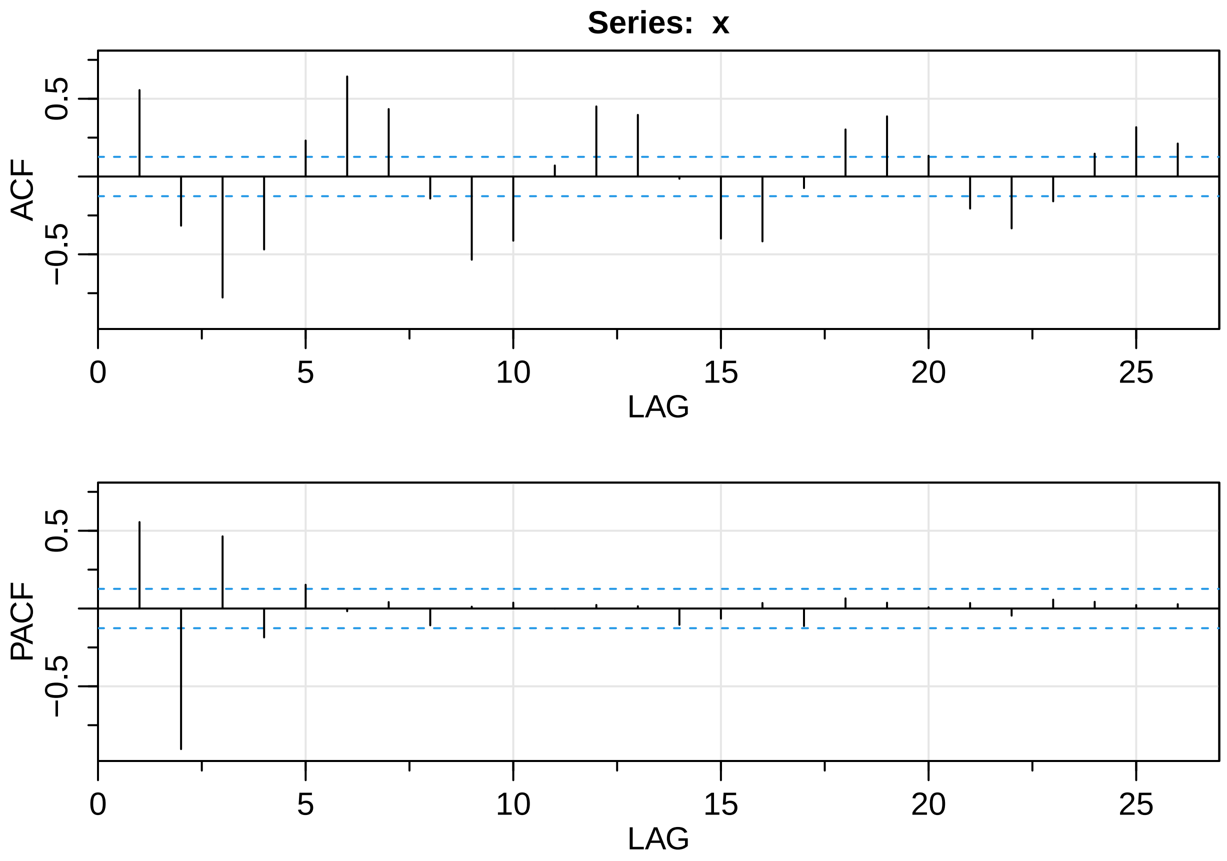
x <- arima.sim(model = list(order = c(2, 0, 1), ar = c(1, -.9), ma = .8), n = 250). Look at the simulated data and the sample ACF and PACF pair to determine a possible model.

Recall that for ARMA(p,q) models, both the theoretical ACF and PACF tail off. In this case, the orders are difficult to discern from data and it may not be clear if either the sample ACF or sample PACF is cutting off or tailing off. In this case, you know the actual model orders, so fit an ARMA(2,1) to the generated data. General modeling strategies will be discussed further in the course.

```r
x <- arima.sim(model = list(order = c(2, 0, 1), ar = c(1, -.9), ma = .8), n = 250)
# astsa is preloaded

# Plot x
plot(x)
```



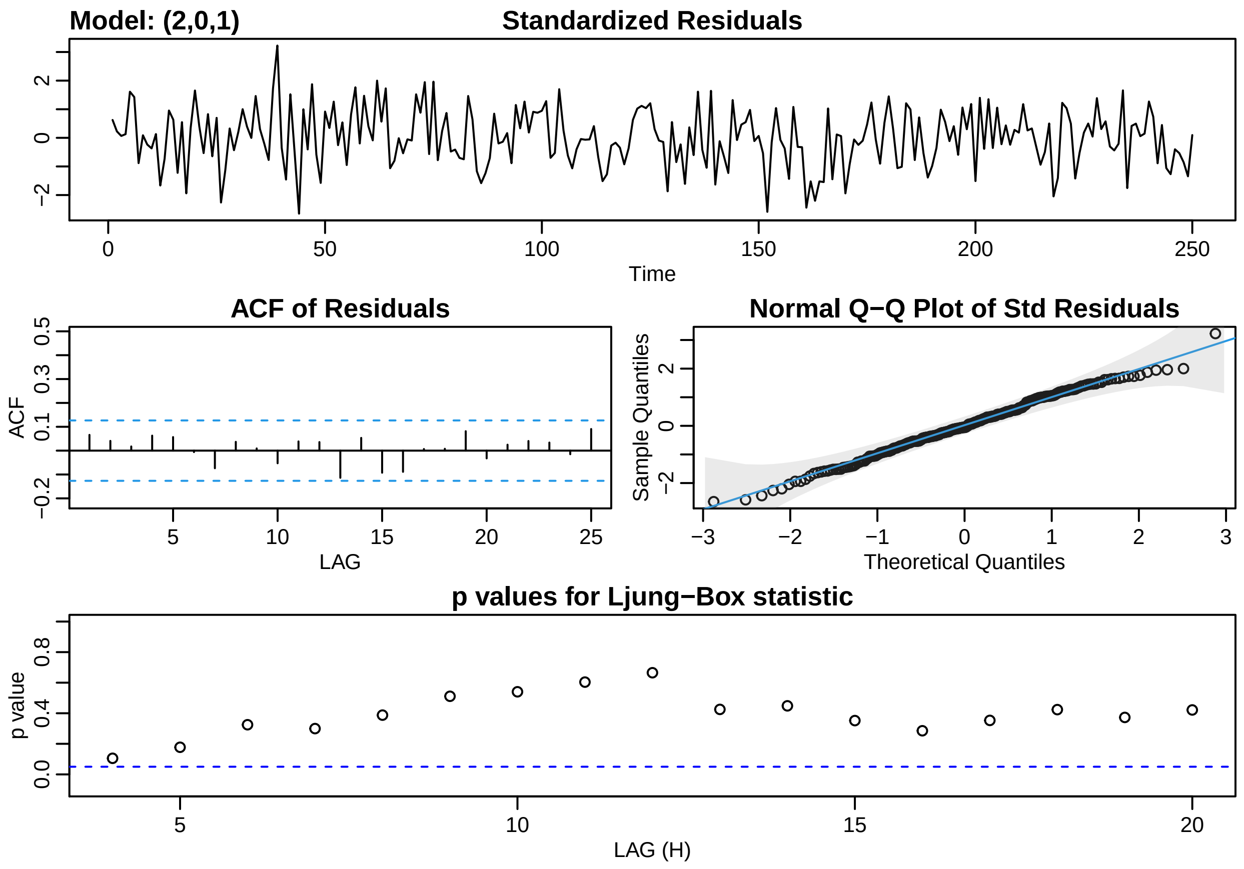
```r
# Plot the sample P/ACF of x
acf2(x)
```



```
##      [,1]  [,2]  [,3]  [,4] [,5]  [,6] [,7]  [,8]  [,9] [,10] [,11] [,12] [,13]
## ACF  0.56 -0.32 -0.78 -0.47 0.23  0.64 0.43 -0.14 -0.53 -0.41  0.07  0.45  0.40
## PACF 0.56 -0.90  0.46 -0.19 0.15 -0.02 0.04 -0.11  0.01  0.04  0.00  0.02  0.01
##      [,14] [,15] [,16] [,17] [,18] [,19] [,20] [,21] [,22] [,23] [,24] [,25]
## ACF  -0.01 -0.40 -0.42 -0.07  0.30  0.39  0.13 -0.21 -0.33 -0.16  0.15  0.32
## PACF -0.10 -0.07  0.04 -0.11  0.07  0.04  0.01  0.04 -0.05  0.06  0.04  0.02
##      [,26]
## ACF   0.21
## PACF  0.03
```

```
# Fit an ARMA(2,1) to the data and examine the t-table
sarima(x, p = 2, d = 0, q = 1)
```

```
## initial  value 1.276415
## iter   2 value 0.503367
## iter   3 value 0.293659
## iter   4 value -0.021257
## iter   5 value -0.041574
## iter   6 value -0.046259
## iter   7 value -0.049275
## iter   8 value -0.049286
## iter   9 value -0.049286
## iter  10 value -0.049286
## iter  10 value -0.049286
## iter  10 value -0.049286
## final  value -0.049286
## converged
## initial  value -0.040784
## iter   2 value -0.040803
## iter   3 value -0.040815
## iter   4 value -0.040845
## iter   5 value -0.040862
## iter   6 value -0.040862
## iter   6 value -0.040862
## iter   6 value -0.040862
## final  value -0.040862
## converged
```



```
## $fit
##
## Call:
## stats::arima(x = xdata, order = c(p, d, q), seasonal = list(order = c(P, D,
##     Q), period = S), xreg = xmean, include.mean = FALSE, transform.pars = trans,
##     fixed = fixed, optim.control = list(trace = trc, REPORT = 1, reltol = tol))
##
## Coefficients:
##          ar1      ar2     ma1   xmean
##       0.9845  -0.8666  0.7974  0.1143
## s.e.  0.0314   0.0308  0.0361  0.1225
##
```

```
## sigma^2 estimated as 0.9003:  log likelihood = -344.52,  aic = 699.04
##
## $degrees_of_freedom
## [1] 246
##
## $ttable
##       Estimate     SE  t.value p.value
## ar1     0.9845 0.0314  31.3984  0.0000
## ar2    -0.8666 0.0308 -28.1192  0.0000
## ma1     0.7974 0.0361  22.0681  0.0000
## xmean   0.1143 0.1225   0.9333  0.3516
##
## $AIC
## [1] 2.796153
##
## $AICc
## [1] 2.796806
##
## $BIC
## [1] 2.866583
```