lab3_fall2017.R

nhanhuynh

Tue Oct 17 12:52:36 2017

```
############ Part 1: Model Identification
###### (c)determine if each process is causal or invertible (look at pdf file for forms of the time seri

# (i) AR(2) with parameters: 0.7 and -0.1
# the AR polynomial for this process is: 1-0.7z+0.1z^2
# set the above polynomial function equal to 0, polyroot() to solve:
polyroot(c(1,-.7,.1))
```

```
## [1] 2+0i 5-0i
```

```
# (ii) MA(2) with parameters: -.7 and .1
# the MA polynomial function for this process is: 1-0.7z+0.1z^2
polyroot(c(1,-.7,.1))
```

```
## [1] 2+0i 5-0i
```

```
# (iii) ARMA(1,1):
# we check if the process is causal or invertible separately.

# check causality by setting up AR polynomial function and set it to 0:
# AR parameter is 0.5 --> AR polynomial function: 1-0.5z
polyroot(c(1,-.5))
```

```
## [1] 2+0i
```

```
# check invertibility by setting up MA polynomial function:
# MA parameter is 0.4 --> MA polynomial function: 1-0.4z
polyroot(c(1,.4))
```

```
## [1] -2.5+0i
```

```
# (iv) ARMA(2,1)
# AR polynomial function is: 1-0.75z+0.5625z^2
polyroot(c(1,-.75,.5625))
```

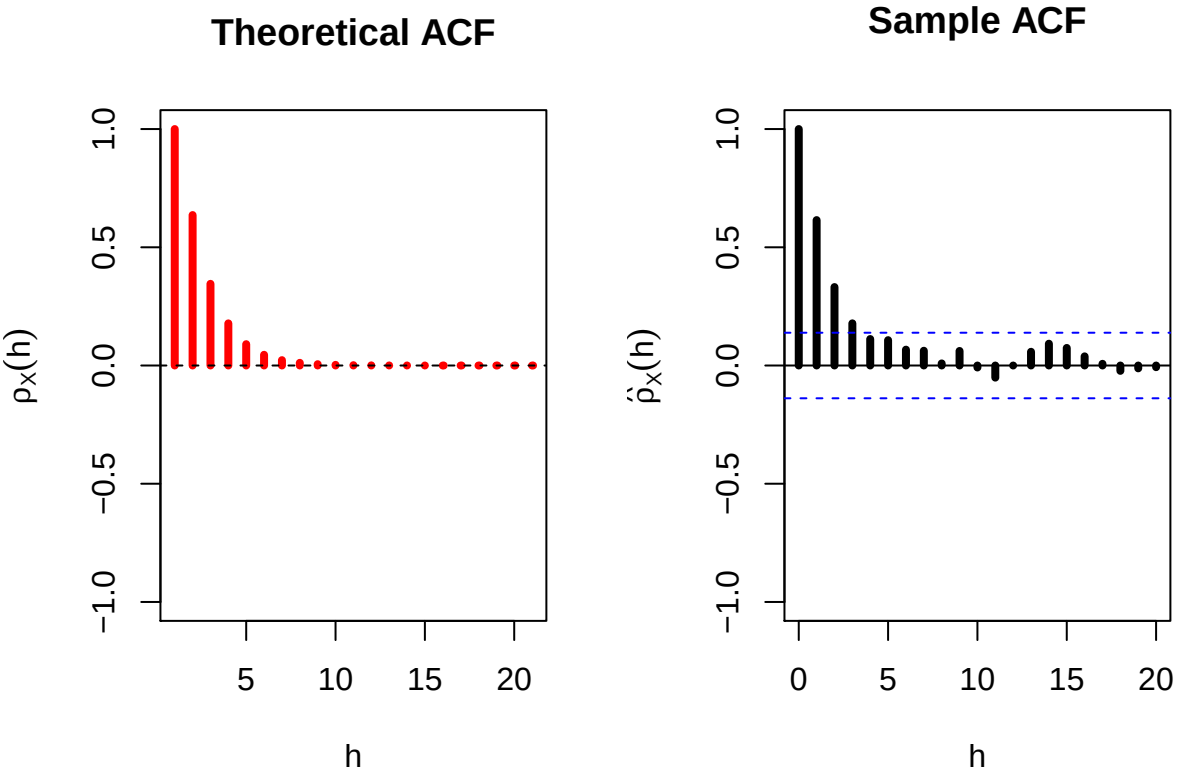
```
## [1] 0.666667+1.154701i 0.666667-1.154701i
```

```
# MA polynomial function is: 1+1.25z
polyroot(c(1,1.25))
```

```
## [1] -0.8+0i
```

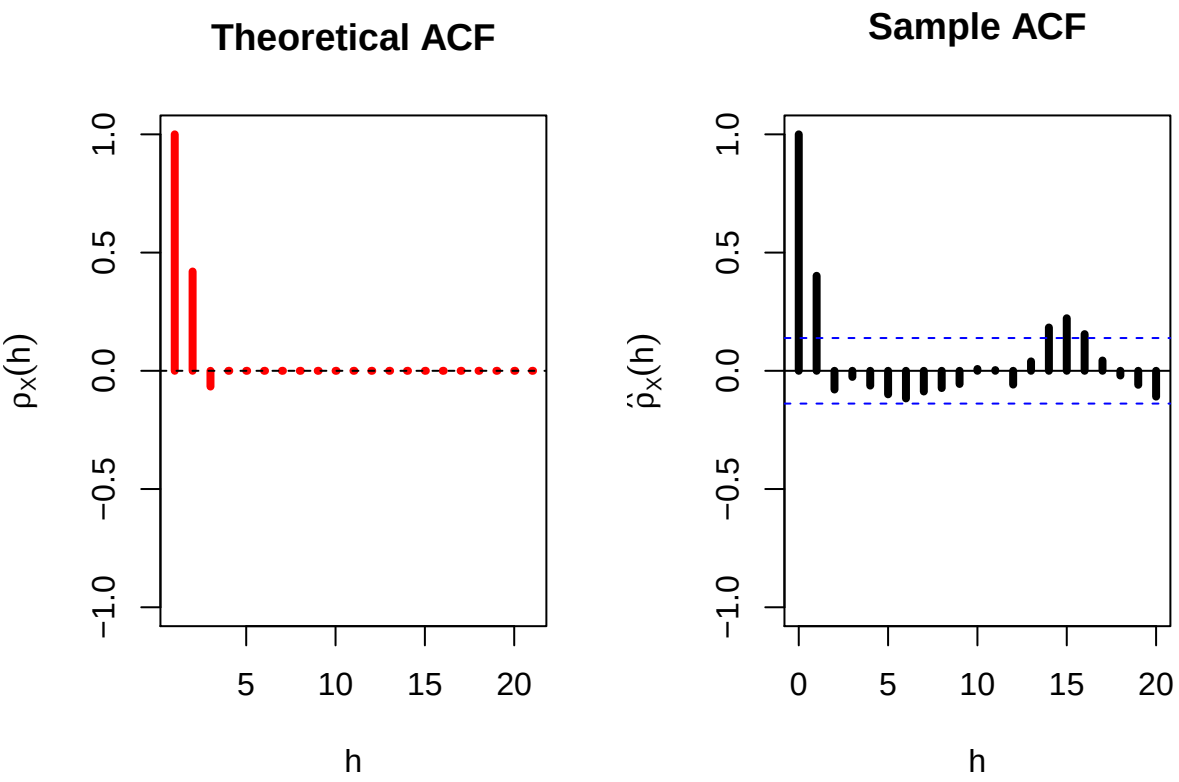
```
##### (d) Simulate 200 observations from each model and compare the sample ACF and PACF with the model

# (i)
# simulate AR(2) process using arima.sim()
# sd argument to specify standard deviation of normal white noise to generate AR
# phi_1=0.7 and phi_2=-0.1
# number of observations is 200
set.seed(1)
ar2 <- arima.sim(model = list(ar = c(0.7,-0.1),sd = 1),n = 200)
```

```r
# theoretical ACF using ARMAacf()
theo_acf<- ARMAacf(ar=c(0.7,-0.1),lag.max = 20,pacf=FALSE)
# display theoretical ACF and sample ACF on the same window:
par(mfrow=c(1,2))
# theoretical acf:
plot(theo_acf,type = "h",ylim = c(-1,1),
     main = "Theoretical ACF",
     col = "red",
     lwd=4,
     ylab = expression(rho[X](h)),
     xlab = "h")
abline(h = 0,lty=2) # Add horizontal line
# sample acf:
acf(ar2,lag.max = 20,
    main = "Sample ACF",
    ylim = c(-1,1),
    lwd=4,
    xlab = "h",
    ylab = expression(hat(rho)[X](h)))
```



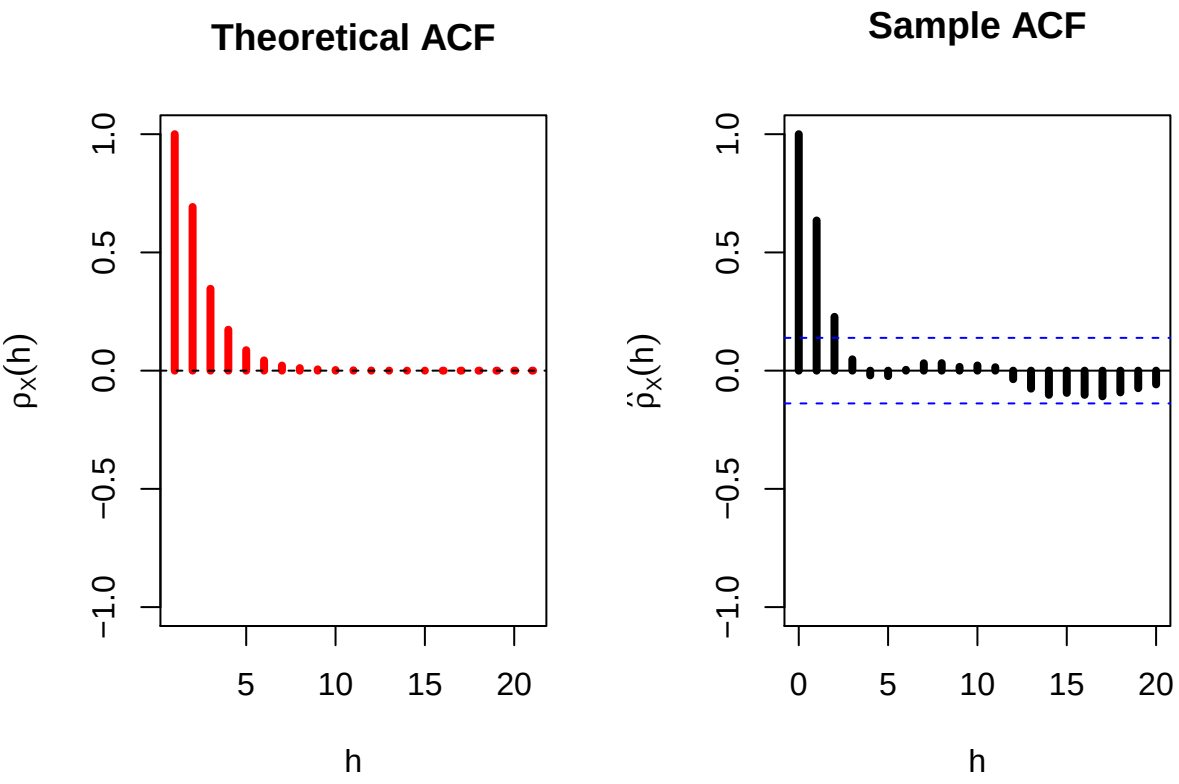

```r
# (ii)
# MA(2) process with theta_1=0.7 and theta_2=-0.1
# n=200 and sd=1
ma1<- arima.sim(model=list(ma=c(0.7,-0.1),sd=1),n=200)
# Theoretical ACF
theo_acf <- ARMAacf(ma = c(0.7,-0.1),lag.max = 20, pacf = FALSE)
# Plot
plot(theo_acf,type = "h",ylim = c(-1,1),
     main = "Theoretical ACF",
```

```
     col = "red",
     lwd=4,
     ylab = expression(rho[X](h)),
     xlab = "h")
abline(h = 0,lty=2) # Add horizontal line
# Sample ACF
acf(ma1,lag.max = 20,
    main = "Sample ACF",
    ylim = c(-1,1),
    lwd=4,
    xlab = "h",
    ylab = expression(hat(rho)[X](h)))
```

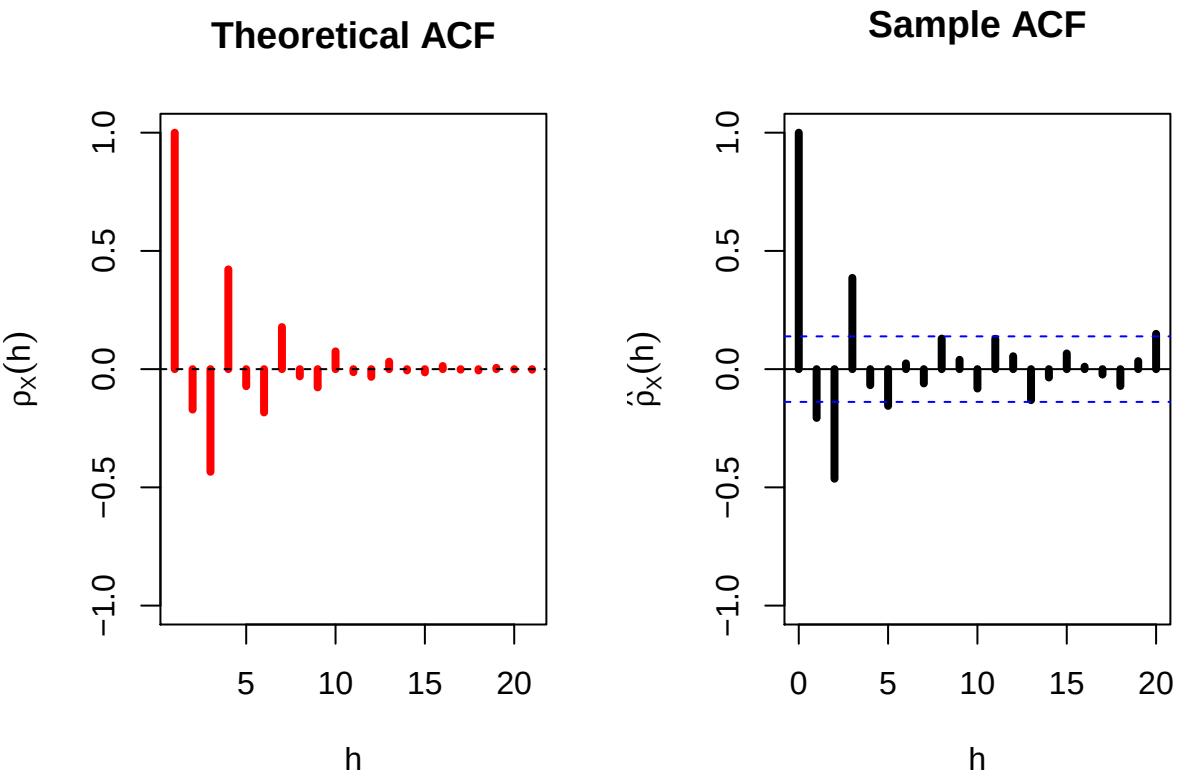



```
# (iii) ARMA(1,1)
# AR parameter is 0.5 and MA parameter is 0.4
# n=200 and sd (of gaussian white noise) is 1
arma11 <- arima.sim(model = list(ar = c(0.5),ma = c(0.4),sd = 1),n = 200)
# Theoretical ACF
theo_acf <- ARMAacf(ar = 0.5, ma = 0.4,lag.max = 20, pacf = FALSE)
# Plot
plot(theo_acf,type = "h",ylim = c(-1,1),
     main = "Theoretical ACF",
     col = "red",
     lwd=4,
     ylab = expression(rho[X](h)),
     xlab = "h")
abline(h = 0,lty=2) # Add horizontal line
# Sample ACF
acf(arma11,lag.max = 20,
```

```
    main = "Sample ACF",
    ylim = c(-1,1),
    xlab = "h",
    lwd=4,
    ylab = expression(hat(rho)[X](h)))
```




```
# (iv) ARMA(2,1)
# AR coefficients are -0.75 and -0.5625 and MA coefficient is 1.25
arma21 <- arima.sim(model = list(ar = c(-0.75,-0.5625),ma = 1.25,sd = 1),n = 200)
# Theoretical ACF
theo_acf <- ARMAacf(ar = c(-0.75,-0.5625), ma = 1.25,lag.max = 20, pacf = FALSE)
# Plot
plot(theo_acf,type = "h",ylim = c(-1,1),
     main = "Theoretical ACF",
     col = "red",
     lwd=4,
     ylab = expression(rho[X](h)),
     xlab = "h")
abline(h = 0,lty=2) # Add horizontal line
# Sample ACF
acf(arma21,lag.max = 20,
    main = "Sample ACF",
    ylim = c(-1,1),
    lwd=4,
    xlab = "h",
    ylab = expression(hat(rho)[X](h)))
```




```
######### Part 2: Model estimation using Yule-Walker equations:
# simulate AR(2) process
ar2 <- arima.sim(model = list(ar = c(1.5,-0.75),sd = 1),n = 144)

#### By hand:
# 1. Estimate the auto-correlation function
acv_ar <- acf(ar2,main = "Sample ACF",plot = F)
# 2. Construct the matrix Rho:
?toeplitz # construct symmetric matrix (Toeplitz matrix)
Rho <- toeplitz(acv_ar$acf[c(1,2)]) # rho_X(0) and rho_X(1)
# 3. Construct the vector rho:
rho <- acv_ar$acf[c(2,3)] # rho_X(1) and rho_X(2)
# 4. Apply the formula:
phi_hat <- solve(Rho) %*% rho
phi_hat
```

```
##           [,1]
## [1,]  1.486300
## [2,] -0.744114
```

```
# Compare the estimated parameters and the actual parameters. (close enough)
# 5. Estimate variance of gaussian white noise:
# (a) Obtain variance of X_t:
var_ar<- acf(ar2,type="covariance",plot=F)$acf[1]
sigma_z <- var_ar*(1-t(rho)%*%solve(Rho)%*%rho)
sigma_z
```

```
##          [,1]
## [1,] 1.134507
```

```
#### Using ar.yw() (Fit autoregressive models):
yw <- ar.yw(ar2,order = 2)
yw
```

```
##
## Call:
## ar.yw.default(x = ar2, order.max = 2)
##
## Coefficients:
##       1        2
##  1.4863  -0.7441
##
## Order selected 2  sigma^2 estimated as  1.159
```

```
### if we construct matrix from to solve for coefficients of Phi --> linear model and Yule Walker estima
```