

Supplementary Practical Exercises for 161.342

Attempt the following questions before looking at the solutions in R (which are in small type underneath).

1. Simulation of White Noise Data

- Using `rnorm` simulate a white noise series containing 100 realisations, and place the result in a vector `e`.
- Create a time plot of the white noise series.
- Create a correlogram of the white noise series.
- By placing a suitable command all on one line, generate several white noise series and their associated time plots (use the ‘up’ arrow on the keyboard to repeat the process).
- Generate several correlograms of white noise series. Observe how some values appear ‘significant’ even when the white noise process is uncorrelated.

Solution:

```
> e <- rnorm(100)
> plot(e, type='l')
> acf(e)
> plot(rnorm(100), type='l') # repeat with up arrow
> acf(rnorm(100)) # again repeat with up arrow
```

2. Simulate 100 realisations for the random walk: $X_t = X_{t-1} + e_t$, where $\{e_t\}$ is white noise, and $e_t \sim N(0, 1)$.

- Place the simulated values in a vector `x`.
- Create a time plot of the simulated random walk series.
- Create a correlogram of the simulated series.
- Using the `diff` command, create a time plot of the differenced series. You should observe that the trends have disappeared.
- Create a correlogram of the differenced series. You should observe that the values now generally fall within the significance lines.
- By placing a suitable command on one line, generate several random walk series and their associated time plots (use the ‘up’ arrow on the keyboard to repeat the process).

Solution:

```
> e <- rnorm(100); x <- rep(0,100); x[1] <- e[1]
> for (t in 2:100) x[t] <- x[t-1] + e[t]
> plot(x, type='l')
> acf(x)
> plot(diff(x), type='l')
> acf(diff(x))
> e=rnorm(100);x=rep(0,100); for(t in 2:100)x[t]=x[t-1]+e[t];plot(x,ty='l')
```

3. Simulation for an ARMA model.

- (a) Simulate 100 realisations of the following ARMA(1,1) model, placing the results in a vector \mathbf{x} :

$$X_t = \frac{3}{4}X_{t-1} + e_t - \frac{1}{2}e_{t-1}$$

- (b) Create a time plot and correlogram of the simulated series.
- (c) Fit an AR(1) model to the simulated series. Get the fitted model parameters.
- (d) Fit an MA(1) model to the simulated series. Get the fitted model parameters.
- (e) Fit an ARMA(1,1) model to the simulated series. Get the fitted model parameters.
- (f) Use AIC to find the best fitted model from the 3 above. Obviously it should be the ARMA(1,1) model (as this was used to generate the data). Can you think of any reason why the ARMA(1,1) model would not give the best fit?

Solution:

```
> e <- rnorm(100); x <- rep(0,100); x[1] <- e[1]
> for (t in 2:100) x[t] <- (3/4)*x[t-1] + e[t] - (1/2)*e[t-1]
> plot(x, type='l'); acf(x)

> arima(x, order=c(1,0,0))

Call:
arima(x = x, order = c(1, 0, 0))

Coefficients:
      ar1  intercept
      0.3150  -0.2805
s.e.    0.0962   0.1254

sigma^2 estimated as 0.7436:  log likelihood = -127.13,  aic = 260.27
> arima(x, order=c(0,0,1))

Call:
arima(x = x, order = c(0, 0, 1))

Coefficients:
      ma1  intercept
      0.2854  -0.2768
s.e.    0.0916   0.1113

sigma^2 estimated as 0.7523:  log likelihood = -127.71,  aic = 261.42
> arima(x, order=c(1,0,1))

Call:
arima(x = x, order = c(1, 0, 1))

Coefficients:
      ar1      ma1  intercept
      0.3642  -0.0540  -0.2814
s.e.    0.2719   0.2857   0.1278

sigma^2 estimated as 0.7433:  log likelihood = -127.12,  aic = 262.23

> AIC(arima(x, order=c(1,0,0)))
[1] 260.2669
> AIC(arima(x, order=c(0,0,1)))
[1] 261.4157
> AIC(arima(x, order=c(1,0,1)))
[1] 262.2321
```