

# Stat 463, Lab 5

October 13, 2006

## 1 In Class

In this lab, we will learn to write our own functions in R. (If you would like more information about writing functions, see section 10 of *An Introduction to R*.) One important reason for learning this is to allow us to run simulations. In this section, we will write a function to simulate AR processes. For homework, you will write your own function to simulate MA processes.

1. We will start by writing a very simple function, so we can learn how things work in R. Functions take arguments, do something to the arguments, and return a result. Here is a short function to enter at the top of your script that will simply add two numbers and return the result.

```
add=function(a,b){  
  result=a+b  
  result}
```

Now, run your script. Then, type `add(2,3)`, and notice the result. This function has two arguments, `a` and `b`. The result is returned by simply typing the object that you would like to have returned, in this case, the numerical variable `result`.

2. Now, we will write a function to simulate an  $AR(p)$  process of length  $T$ . What information is necessary to do this? We will need the AR parameters  $\phi_1, \dots, \phi_p$ , the variance of the white noise  $\sigma^2$ , and the length of the time series we would like to generate  $T$ . These will be the arguments for our function.

It would be nice if we could write one function that will generate an AR time series regardless of  $p$ . We will accomplish this by entering that AR parameters as a vector. Our function will be as follows:

```
arsim=function(phis, sigsq, T){
  p=length(phis) #find the number of lags in our AR
  noise=rnorm(T+p, sd=sqrt(sigsq)) #generate the white noise plus a few to get started
  x=c(noise[1:p],rep(0,T)) #put the initial noise terms in and set the rest to zero
  for (i in (p+1):(T+p)){ #this loop generates the AR series with the recursive formula
    x[i]=phis %*% x[i-(1:p)] +noise[i]
  }
  x=x[(p+1):(T+p)] #throw away those initial starting points
  x #return the time series
}
```

We should walk through this function definition to make sure we understand things. First, we see that there are three arguments. The first argument is `phis` which is a vector which contains the AR parameters in order,  $\phi_1, \dots, \phi_p$ . The second is `sigsq`, the variance of the white noise, and the last is how many observations we want, `T`.

The first line of the function finds the length of the `phis` which is  $p$ . We could pass an additional argument `p`, but this way is simpler. The second line generates  $T+p$  independent normal random variables with variance `sigsq`. In terms of how we have been writing our models, this command generates the  $w_t$ . We generate  $p$  extra white noise terms to use as our initial values for the first  $p$  values of our time series. We have to start our series somehow, and this is not a ridiculous way to do it. Later, we will come up with more clever ways to deal with this problem.

We are going to use the variable `x` to store our time series. So, the next command sets up the vector `x` with the first  $p$  white noise terms and the rest zeros. The next set of statements is a `for` loop. This loop generates the time series using our recursive definition of an AR series. Note the operator `%*%`. This take the inner product of two vectors. In other words, the first element of vector one times the first element of vector two plus the second element of vector one times the second element of vector two plus and so on. The command after the `for` loop simply gets rid of those first  $p$  white noise terms leaving us with the AR process only. The final `x` returns our time series.

3. Now that we have a function to generate an AR series, let us put it to work. Generate an  $AR(1)$  time series of length 200 and with  $\phi_1 = 0.5$

and  $\sigma^2 = 1$  using the command

```
x1=arsim(c(0.5), 1,200)
```

Take a look at a plot of the data and the ACF of this time series. Is this what you would expect?

Also, generate a similar time series but with  $\phi_1 = -0.5$ . How do the two time series look different? How about the ACF's?

4. We saw that with a  $|\phi_1| < 1$  the process appears reasonable and stationary, and the last simulations you did confirmed that. Try a  $\phi_1$  of 1.1, 1.01, 1.001, and 0.99. What do you notice?

## 2 Homework

1. Write a function called `masim` to generate a  $MA(q)$  series of length  $T$ . A few hints: This will be very similar but not exactly the same as the function to generate AR series. You may want to change the notation to make it more natural, `this` become `thetas` and `p` becomes `q`, etc. You will need  $T + q$  white noise terms to create an MA of order  $q$  and length  $T$ .
2. Make sure that your function works. Simulate a model with  $\sigma^2 = 1$ ,  $\theta_1 = 0.5$ , and  $\theta_2 = 2$  with  $T = 10,000$ . Make an ACF plot. Is this consistent with the model that you generated? Try generating a model with the same parameters but only 200 observations. Make an ACF. Repeat this a few times, what do you notice about the autocorrelations and the dotted blue lines?
3. Verify example 3.4 from the book. Simulate the two MA models of length 10,000 using your newly created function and the two sets of parameters in this example. Now, fit those two MA models using the command `arima(x, c(0,0,1))` where `x` is the name of your time series. (We'll talk more about this command later; just trust the output for now.) Also, make ACF plots of your two simulated series. Does this confirm what was shown in theory?