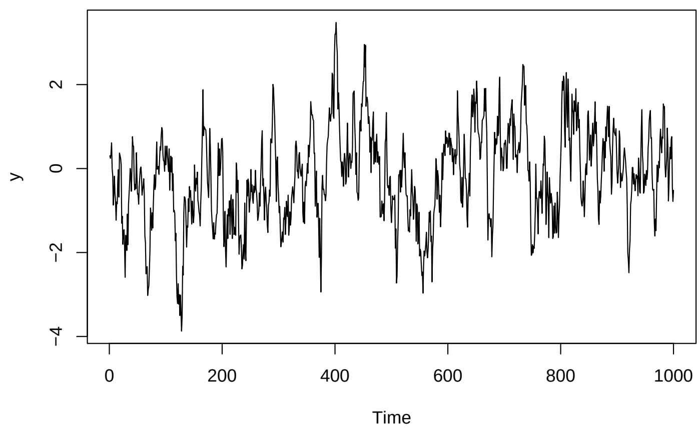
MATH3026/4022: Time Series Analysis/Time Series and Forecasting
Computer Practical 1 - SOLUTIONS

1. (b) Use the following command to produce a time plot of the simulated data

SOLUTION:

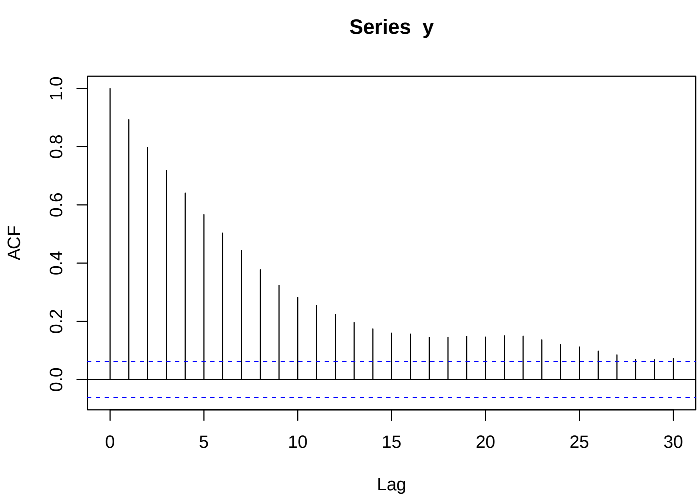
```
#Produce a time plot of the simulated AR(1) data
ts.plot(y)
```



(c) Use the following command to produce a plot of the sample autocorrelation function against the lag.

SOLUTION:

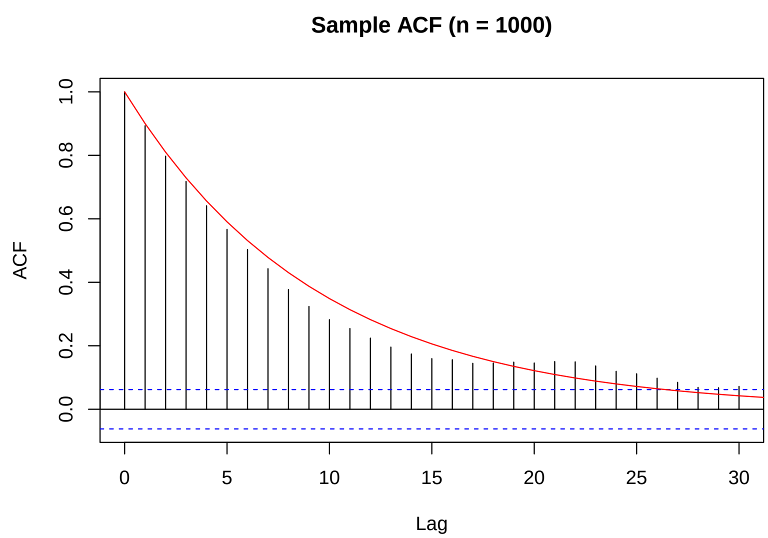
```
#Produce a plot of the sample ACF for the simulated AR(1) data
acf(y)
```

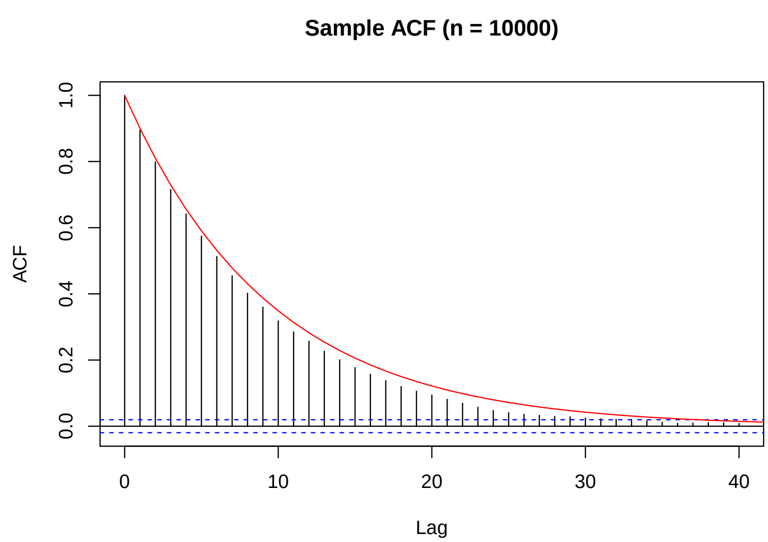


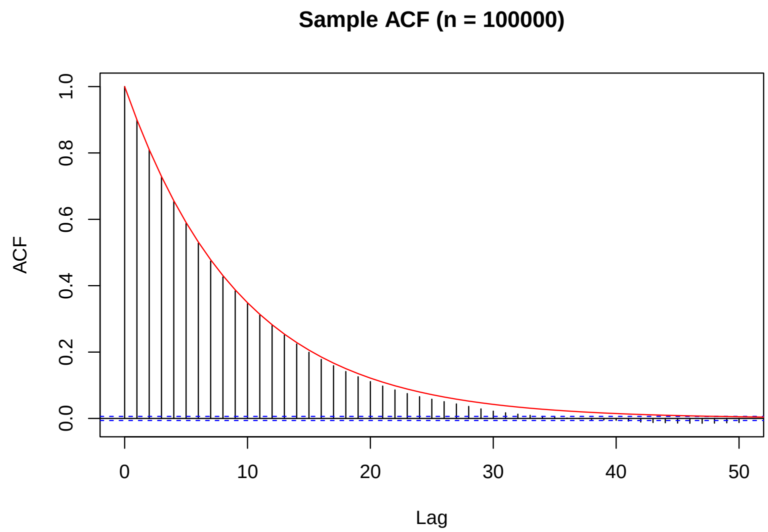
(e) Using the command below, produce another plot of the sample ACF, this time with the theoretical ACF shown on the plot as a red line. Is the sample ACF close to the theoretical ACF? Repeat this step for simulated AR(1) processes with $n = 10000$ and $n = 100000$ observations. What do you notice?

SOLUTION:

```
#Produce a plot of the sample ACF for the simulated AR(1) data
acf(y)

#Add a line (in red) that shows the theoretical ACF against lags 0 to 100
y_lag<-c(0:100)
lines(y_lag,Q1.AR1.acf,col="red")
```







We see that, as n increases, the sample ACF gets closer to the true ACF, which we should expect.

2. Now that you have explored and run the R code in Question 1, carry out the following tasks, using the commands presented in Question 1. as a guide where necessary. Please remember to save and comment your R code.

   (a) Simulate $n = 1500$ observations from an $MA(1)$ process with parameters $\theta_1 = 0.6$ and $\sigma_z^2 = 0.49$.

   SOLUTION:

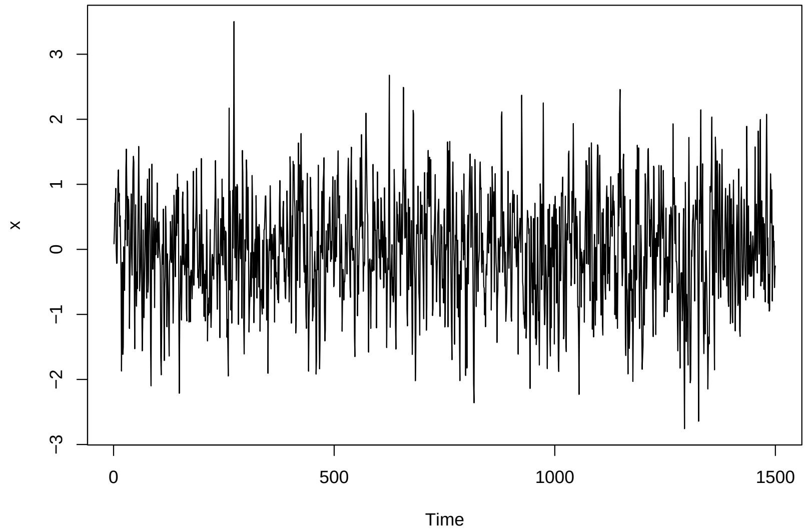
```
#Set the seed of your choice
set.seed(14102)

#Assign theta1 to be 0.6
theta1<-0.6

#sigma_z=sqrt(0.49) = 0.7
sigma_z<-0.7

#Simulate 1500 observations from the MA(1) model with
#theta1=0.6 and sigma_z=0.7
x<-arima.sim(n=1500,model=list(ma=c(theta1)),sd=sigma_z)
```

   (b) For the data simulated in (a) produce:

      (i) A time plot.

      SOLUTION:

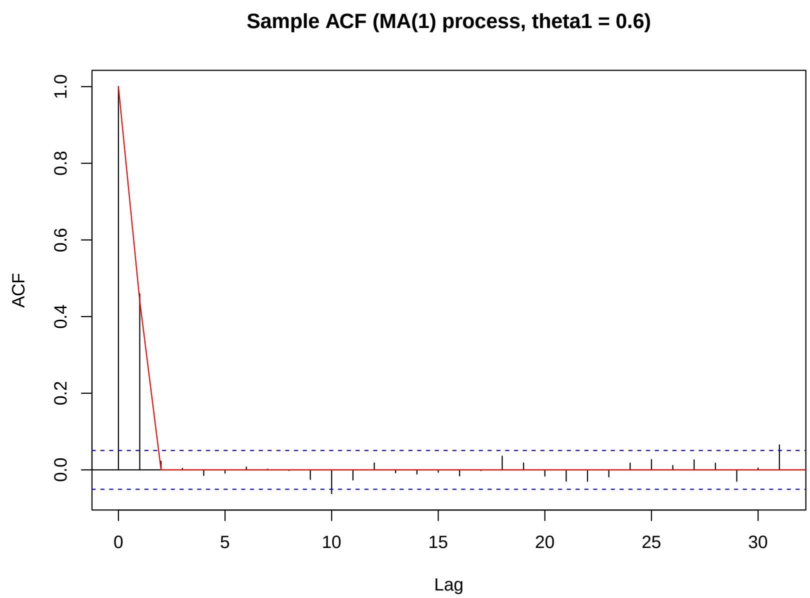
```
#Code to produce the time plot
ts.plot(x)
```



(ii) A plot of the sample ACF against the lag, with a line showing the theoretical ACF included on the plot for comparative purposes.

SOLUTION:

```
#Code to calculate the theoretical ACF for the MA(1) process
Q2.MA1.acf<-ARMAacf(ar=c(0),ma=c(theta1),lag.max=100)

#Produce a plot of the sample ACF for the simulated AR(1) data
acf(x,main="Sample ACF (MA(1) process, theta1 = 0.6)")

#Add a line (in red) that shows the theoretical ACF against lags
#0 to 100.
x_lag<-c(0:100)
lines(x_lag,Q2.MA1.acf,col="red")
```

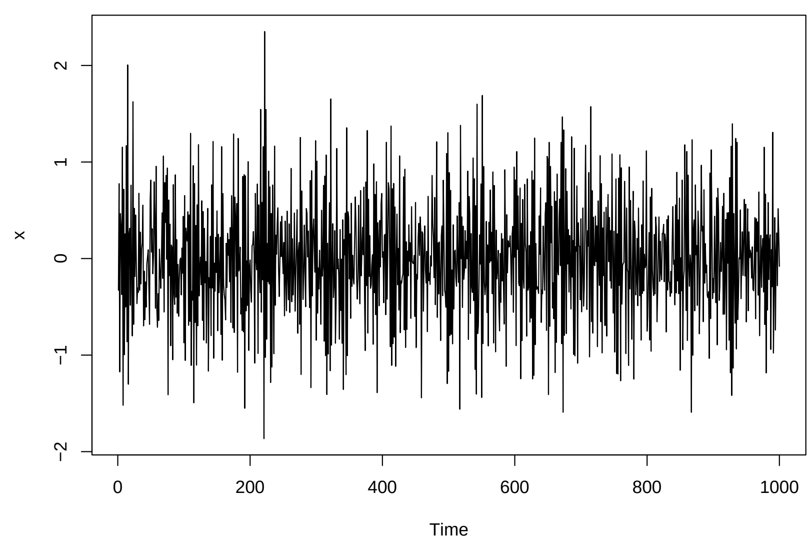


3. Instead of using `arima.sim` to simulate values from a time series, we can simulate values using code that we have written to do this directly.

   (b) Produce a time plot for the data simulated in (a).

   SOLUTION:

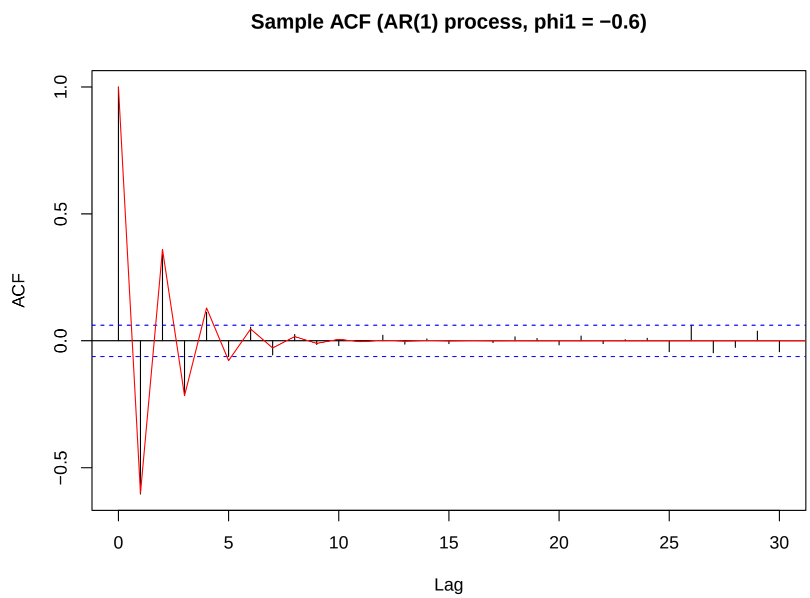
```
#Code to produce the time plot
ts.plot(x)
```



(c) Using the sample and theoretical ACFs, verify that the data you simulated in (a) are from an $AR(1)$ process with $\phi_1 = -0.6$.

SOLUTION:
Code to produce the plot:

```
#Produce a plot of the sample ACF for the simulated AR(1) data
acf(x,main="Sample ACF (AR(1) process, phi1 = -0.6)")

#Code to calculate the theoretical ACF for the A1(1) process
Q3.AR1.acf<-ARMAacf(ar=c(-0.6),ma=c(0),lag.max=100)

#Add a line (in red) that shows the theoretical ACF against lags
#0 to 100
x_lag<-c(0:100)
lines(x_lag,Q3.AR1.acf,col="red")
```



4. (a) Writing you own code (and without using `arima.sim`), simulate 2000 values from an $MA(1)$ process with $\theta_1 = 0.8$ and $\sigma_z^2 = 0.64$. You should assume that the white noise terms are normally distributed. Please remember to comment your R code and to save your work.

SOLUTION:

Below is an example of some code that can be used for this simulation.

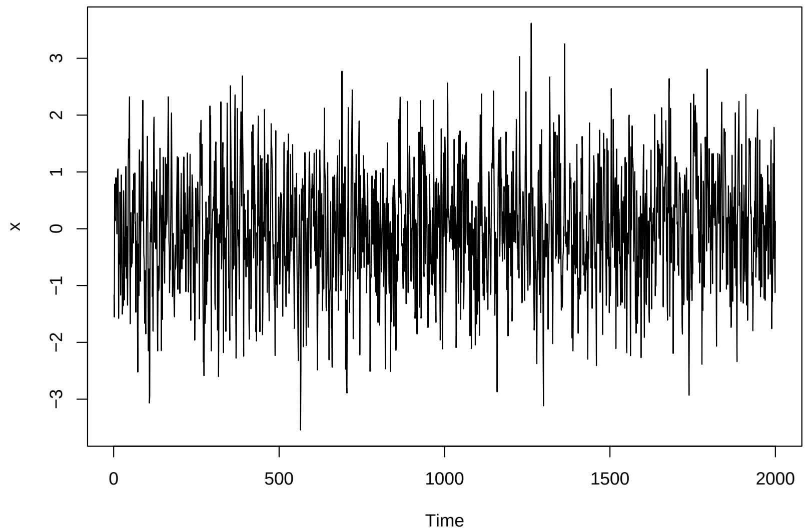
```
#Set the seed
set.seed(9345)
```

```
#State the required number of of simulated values (n=2000).
n<-2000

#Define an empty vector of length n which will eventually contain
#the simulated time series values.
x<-numeric(n)

#Define the values of phi1 and sigma_z
theta1<-0.8
sigma_z<-sqrt(0.64)

#Define an nx1 vector of independent and identically distributed white
#noise terms: z_1,...,z_n. Here, we assume that the white noise terms
#are normally distributed.
z<-rnorm(n,mean=0,sd=sigma_z)

#Set the first value of the series (first element of z) to be the random
#white noise term (z_1).
x[1]<-z[1]

#Values 2:n of the vector x will depend on previous values of the
#series, according to the MA(1) model. We use the `for loop' below to
#populate values of x.
for(t in 2:n){
  x[t]<-theta1*z[t-1]+z[t]
}

#We define the simulate data to be a time series object.
x<-ts(x,start=1,end=n)
```

(b) For the data simulated in (a) produce:

(i) A time plot.

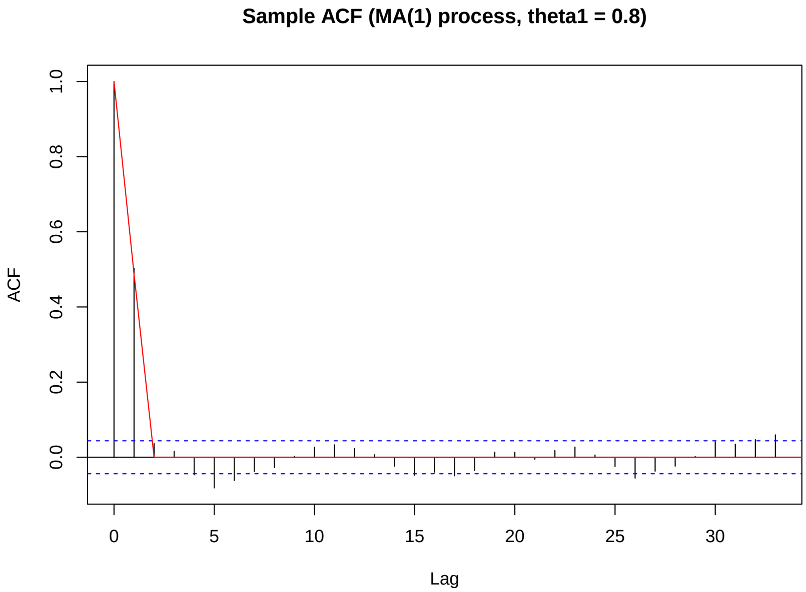
SOLUTION:



(ii) A plot of the sample ACF against the lag, with a line showing the theoretical ACF included on the plot for comparative purposes. Do the data appear to be from an $MA(1)$ process?

SOLUTION:

```
#Produce a plot of the sample ACF for the simulated MA(1) data
acf(x,main="Sample ACF (MA(1) process, theta1 = 0.8)")

#Code to calculate the theoretical ACF for the A1(1) process
Q4.MA1.acf<-ARMAacf(ar=c(0),ma=c(0.8),lag.max=100)

#Add a line (in red) that shows the theoretical ACF against lags
#0 to 100
x_lag<-c(0:100)
lines(x_lag,Q4.MA1.acf,col="red")
```



5. The dataset `eng_rain` can be found on the course moodle page in CSV format and contains annual rainfall measurements (in millimetres) for England for the years 1862–2021, sourced from the UK Met Office databank:

   https://www.metoffice.gov.uk/research/climate/maps-and-data/uk-and-regional-series.

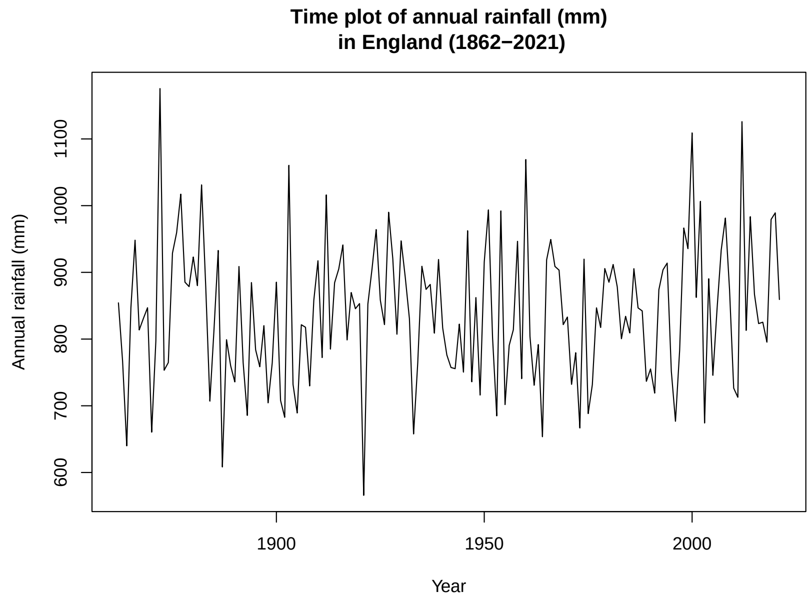
   We will work with these data in this question.

   (c) Produce a time plot of the annual rainfall data. Does the time series appear to be stationary? Justify your answer.

   SOLUTION:

   The following code was used to produce the time plot below. Note the use of the terms `xlab`, `ylab` and `main` to provide plot axes labels and a title.

```
ts.plot(annual_rain,ylab="Annual rainfall (mm)",xlab="Year",
main="Time plot of annual rainfall (mm)\n in England (1862-2021)")
```



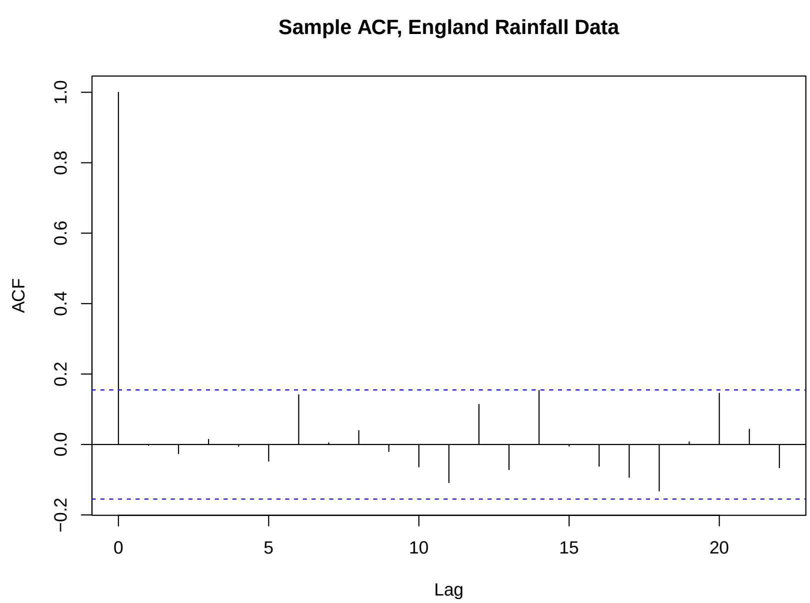
Looking at the time plot, it seems plausible that the time series is stationary.

(d) What type of model would you suggest for the annual rainfall data? Explain your reasoning.

SOLUTION:

As discussed during lectures, a good starting point when considering different models is to assess the behaviour of the sample ACF. Hence, we produce a plot of the sample ACF for the England rainfall data, shown below.

```
acf(annual_rain,main="Sample ACF, England Rainfall Data")
```



Examining the plot of the sample ACF we see that, after lag 0, the sample ACF has values very close to zero. This may suggest that rain totals in different years are uncorrelated. If X_t is the annual rainfall in year t, we might consider a model of the form

$$X_t = \mu + Z_t$$

where μ is the mean annual rainfall in a given year and $\{Z_t\}$ is a white noise process with variance σ_z^2. Note – we could fit this model easily using the `lm` command in R (assuming that the white noise is normally distributed).