Lisa Over
Homework 4
February 10, 2015

GIBBS Function

```
#Gibbs function receives four parameters: "roe" for correlation,
 "N" for number of realizations, "lag" for determining how many
 realizations to skip between saves, and "burnin" for detemrining
 how many realizations to skip before starting to save. Gibbs
 generates N independent random normal values.

gibbs <- function(roe,N,lag,burnin) {

 #Set N to be Nations*lag+burnin
 N <- N*lag + burnin
 #Initialize x and y
 x <- 80
 y <- 80
 #Initialize vectors
 x.s <- NULL
 y.s <- NULL

   for(i in 1:N) {

 #generate a "y"
 y <- rnorm(1, roe*x, sqrt(1-roe^2))
 #generate an "x"
 x <- rnorm(1, roe*y, sqrt(1-roe^2))

   #if i is greater than burnin and if i is a multiple of the lag,
 store x and y
   if(i > burnin) {
     if(i %% lag == 0) {
     x.s <- c(x.s,x)
     y.s <- c(y.s,y)
     }
    }
   }
   vectors <- list("x" = x.s, "y" = y.s)
   return(vectors)
 }
```

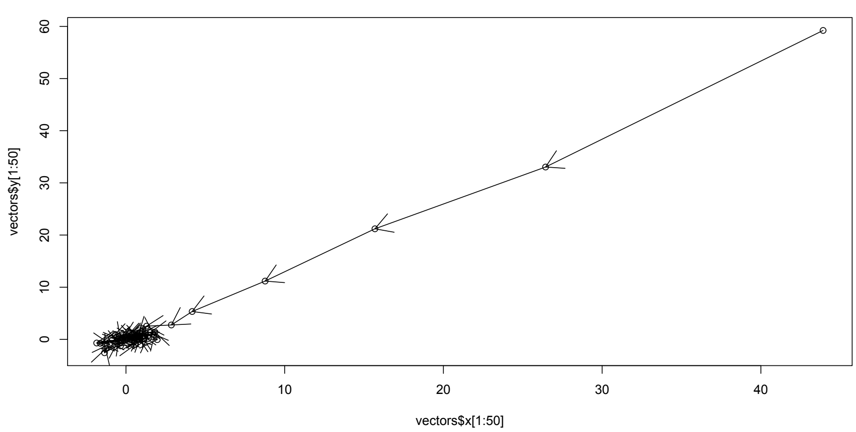
Lisa Over
Homework 4
February 10, 2015

##1##

```
#Run Gibbs with N=500, lag=1, and burnin=0
roe <- 0.75
N <- 500
lag <- 1
burnin <- 0

vectors <- gibbs(roe,N,lag,burnin)
```

#a

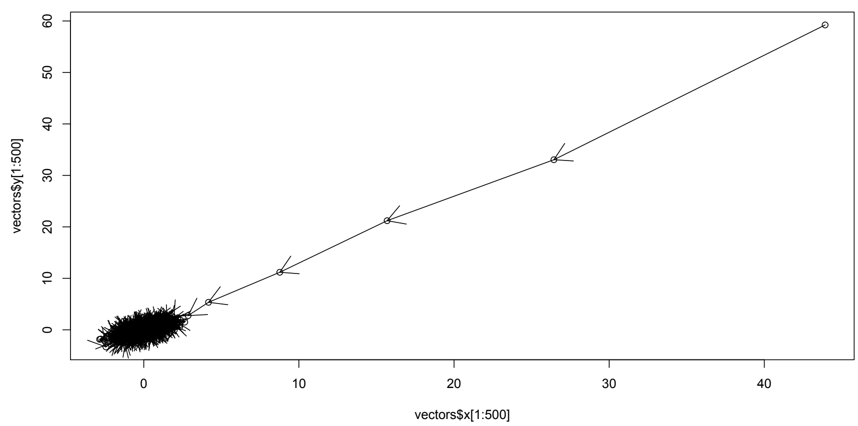
```
#Plot first 50 values and connect the points in order generated
 using arrows command

plot(vectors$x[1:50],vectors$y[1:50])
arrows(vectors$x[1:49],vectors$y[1:49],vectors$x[2:50],vectors$y[
 2:50],size=0.1,open=T)
```



#b

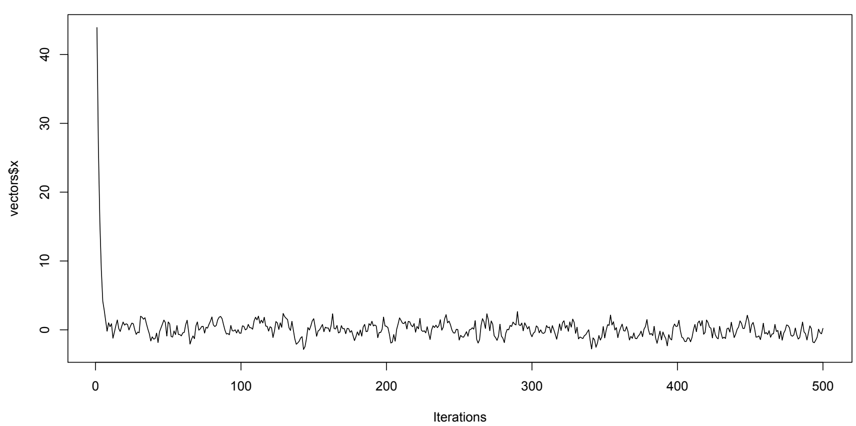
```
#Plot all 500 values with arrows to illustrate possible values
 for lag and burnin

plot(vectors$x[1:500],vectors$y[1:500])
arrows(vectors$x[1:499],vectors$y[1:499],vectors$x[2:500],vectors
 $y[2:500],size=0.1,open=T)
```

Lisa Over
Homework 4
February 10, 2015



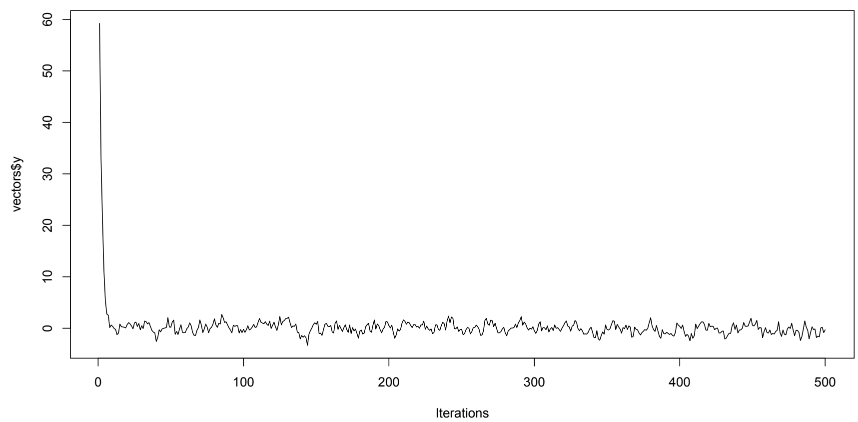
#c

```
#Use x.s and y.s from initial run to determine values for burnin
 based on drop in ts.plot - take larger drop to set burnin.

ts.plot(vectors$x,xlab="Iterations")
```



Lisa Over
Homework 4
February 10, 2015

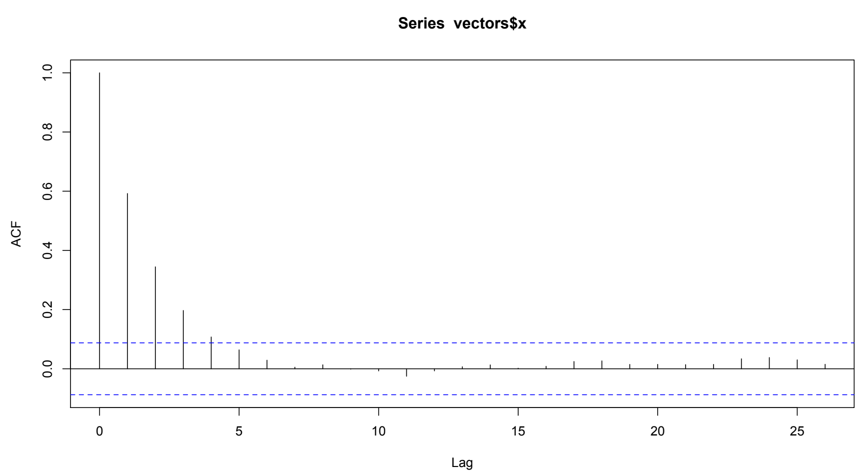
```
ts.plot(vectors$y,xlab="Iterations")
```




```
#Set burnin to 20 (10 may work but choose 20 to be conservative)
```

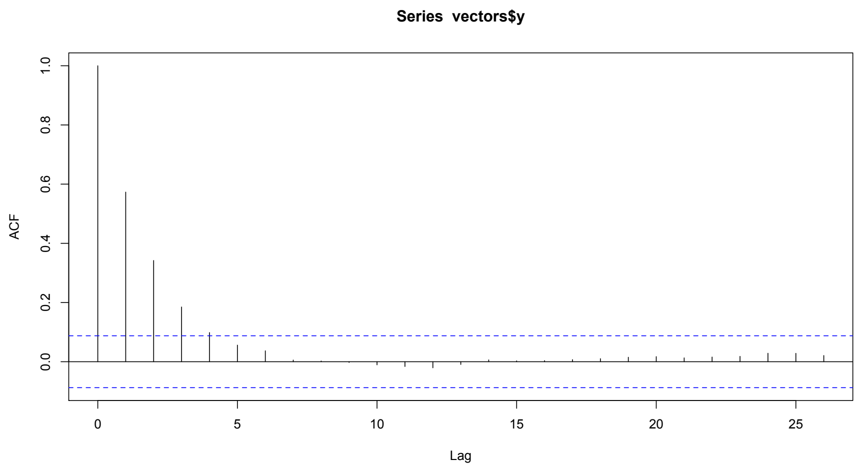
#d

```
#Use x.s and y.s from initial run to determine values for lag
 based on largest acf - autocorrelation - for both vectors, x.s
 and y.s (conservatively select bar that falls below the line).

acf(vectors$x)
```



Lisa Over
Homework 4
February 10, 2015

```
acf(vectors$y)
```




```
#Set lag to 5
```

#e

```
#Run Gibbs with N=10000, lag=5, and burnin=20 and verify
 normality
roe <- 0.75
N <- 10000
lag <- 5
burnin <- 20

vectors <- gibbs(roe,N,lag,burnin)

mean(vectors$x)
#[1] -0.003423303

var(vectors$x)
#[1] 1.014041

quantile(vectors$x)
#           0%          25%          50%          75%         100%
#-3.792598217 -0.683614847  0.002497831  0.655396187  3.577334894
```

Lisa Over
Homework 4
February 10, 2015

```
mean(vectors$y)
#[1] -0.000683986

var(vectors$y)
#[1] 1.006094

quantile(vectors$y)
#          0%         25%         50%         75%        100%
#-3.61015779 -0.68406521 -0.01410746  0.66701307  3.92869627
```

##2##

```
roe <- -0.9
N <- 15000
lag <- 5
burnin <- 20

r.s <- NULL
p.q1 <- NULL
p.q2 <- NULL
p.q1n4 <- NULL
p <- 0

while(roe <= .9) {

 #create roe vector
 r.s <- c(r.s,roe)

 #Run gibbs to generate two independent normal random variables
 vectors <- gibbs(roe,N,lag,burnin)

 #initialize xy.s - vector to store scenarios
 xy.s <- NULL
 #restrict x and y to quadrant 1
 xy.s <- (vectors$x > 0 & vectors$y > 0)
 count <- 0
    count <- length(xy.s[xy.s=="TRUE"])
 p <- count/N
 #accumulate probabilities for quadrant 1
 p.q1 <- c(p.q1,p)

 xy.s <- NULL
```

Lisa Over
Homework 4
February 10, 2015

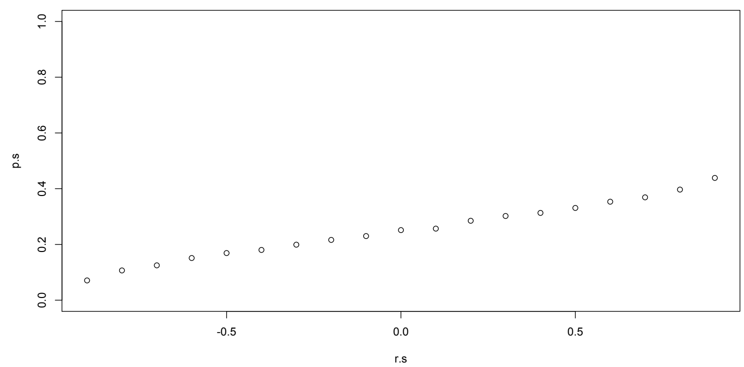
```
 #restrict x and y to quadrant 2
 xy.s <- (vectors$x < 0 & vectors$y > 0)
 count <- 0
    count <- length(xy.s[xy.s=="TRUE"])
 p <- count/N
 #accumulate probabilities for quadrant 1
 p.q2 <- c(p.q2,p)

 xy.s <- NULL
 #restrict x and y to quadrants 1 and 4
 xy.s <- (vectors$x > 0)
 count <- 0
    count <- length(xy.s[xy.s=="TRUE"])
 p <- count/N
 #accumulate probabilities for quadrants 1 and 4
 p.q1n4 <- c(p.q1n4,p)


 #increment roe by 0.1
 roe <- roe + 0.1
}

#Plot probability vs. roe with vertical axis limits (0,1) for
 each scenario
```

Lisa Over
Homework 4
February 10, 2015

#a

```
plot(r.s,p.q1,ylim=c(0,1))
```

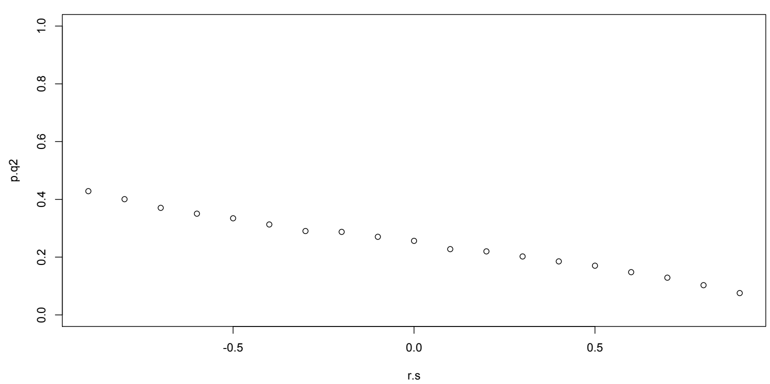
P(X > 0 and Y > 0) approaches 0 as roe approaches -1, is 0.25 when roe=0, and approaches 0.5 as roe approaches 1.



#b

```
plot(r.s,p.q2,ylim=c(0,1))
```

P(X < 0 and Y > 0) approaches 0.5 as roe approaches -1, is 0.25 when roe=0, and approaches 0 as roe approaches 1.



Lisa Over
Homework 4
February 10, 2015

#c

```
plot(r.s,p.q1n4,ylim=c(0,1))
```

```
P(X > 0) is 0.5 as roe moves from -0.9 to 0.9.
```