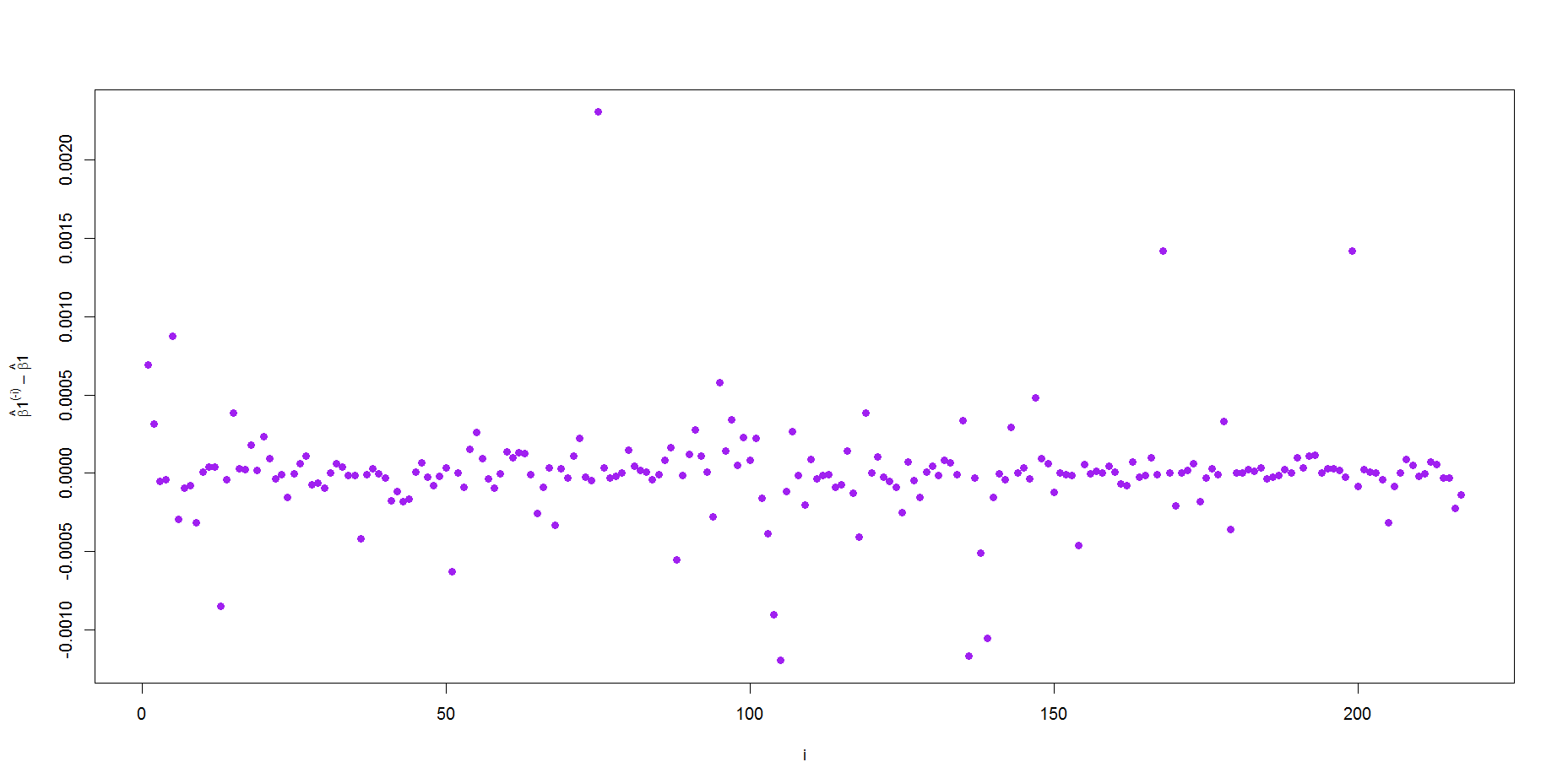
2e)

Figure 3. A plot of the influences, 𝛽1(−𝑖)-𝛽1versus i, from part 2e)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Codes

1a We have Downloaded the data from the url and saved it in a file named “SearchResults.txt” and then scanned that file in R and stored it in Row wise

1a.

xdata = scan("SearchResults.txt",skip=3,nlines=232,what="char")

z = matrix(xdata,ncol=12,byrow=T)

b.

## x11 = xdata

## xgrid = vector of values where you'll compute the kernel estimate.

## bwid = bandwidth

density = function(x11,xgrid,bwid){

n = length(xgrid)

y = rep(0, n)

for(i in 1:n){

y[i] = sum(dnorm(x11-xgrid[i], sd=bwid)) / length(x1)

}

y

}

##Getting the magnitude column

a = z[,4]

##Converting it into numeric and making it into a global variable.Data to be fed to the density function

p <<- as.numeric(a)

##Using Scott thumb rule to calculate bandwidth

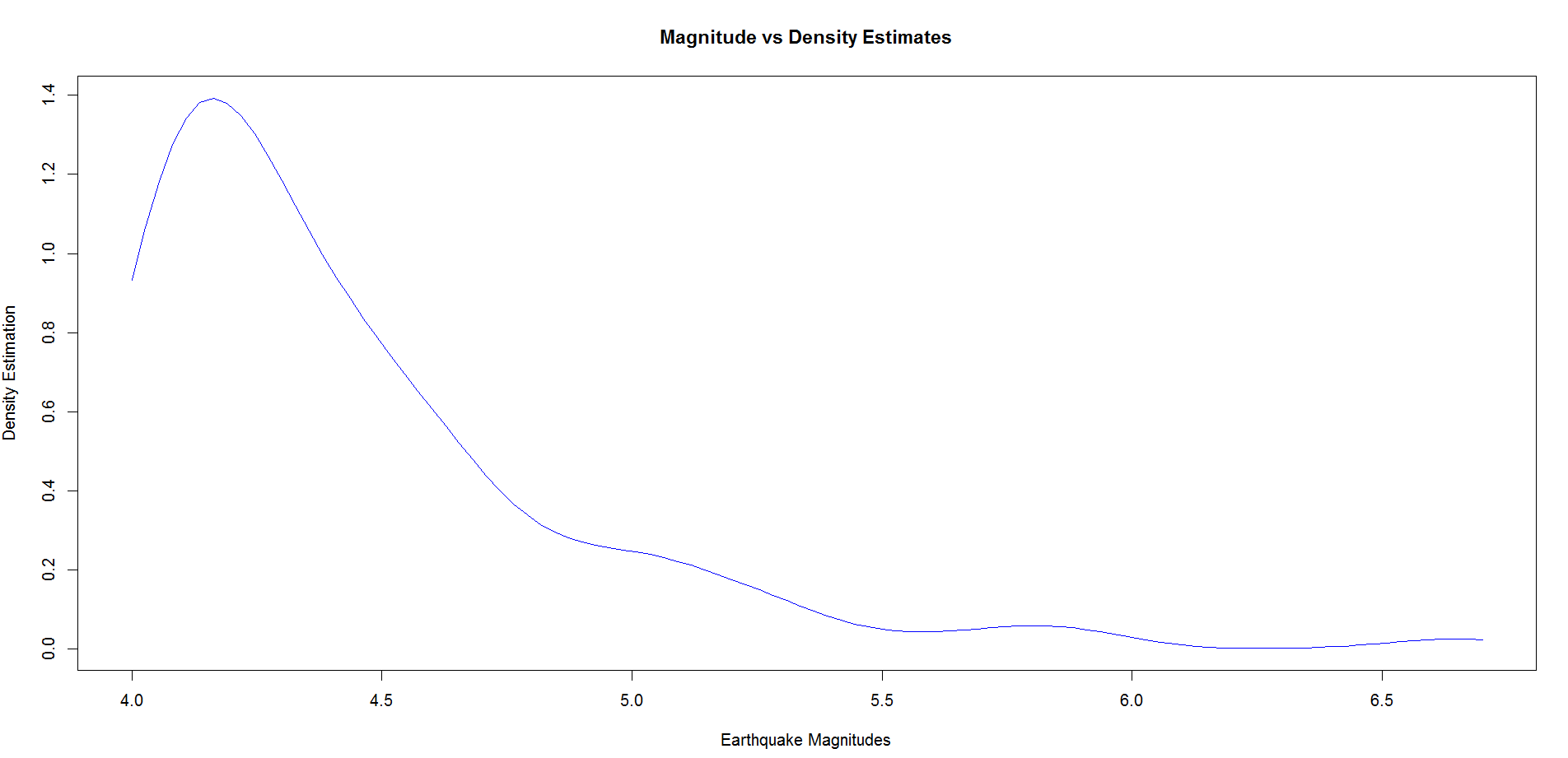
bwid=bw.nrd(p)

##max and min to decide the grid

q = seq(min(m),max(m),length=100)

yden<<- density(p,q,bwid)

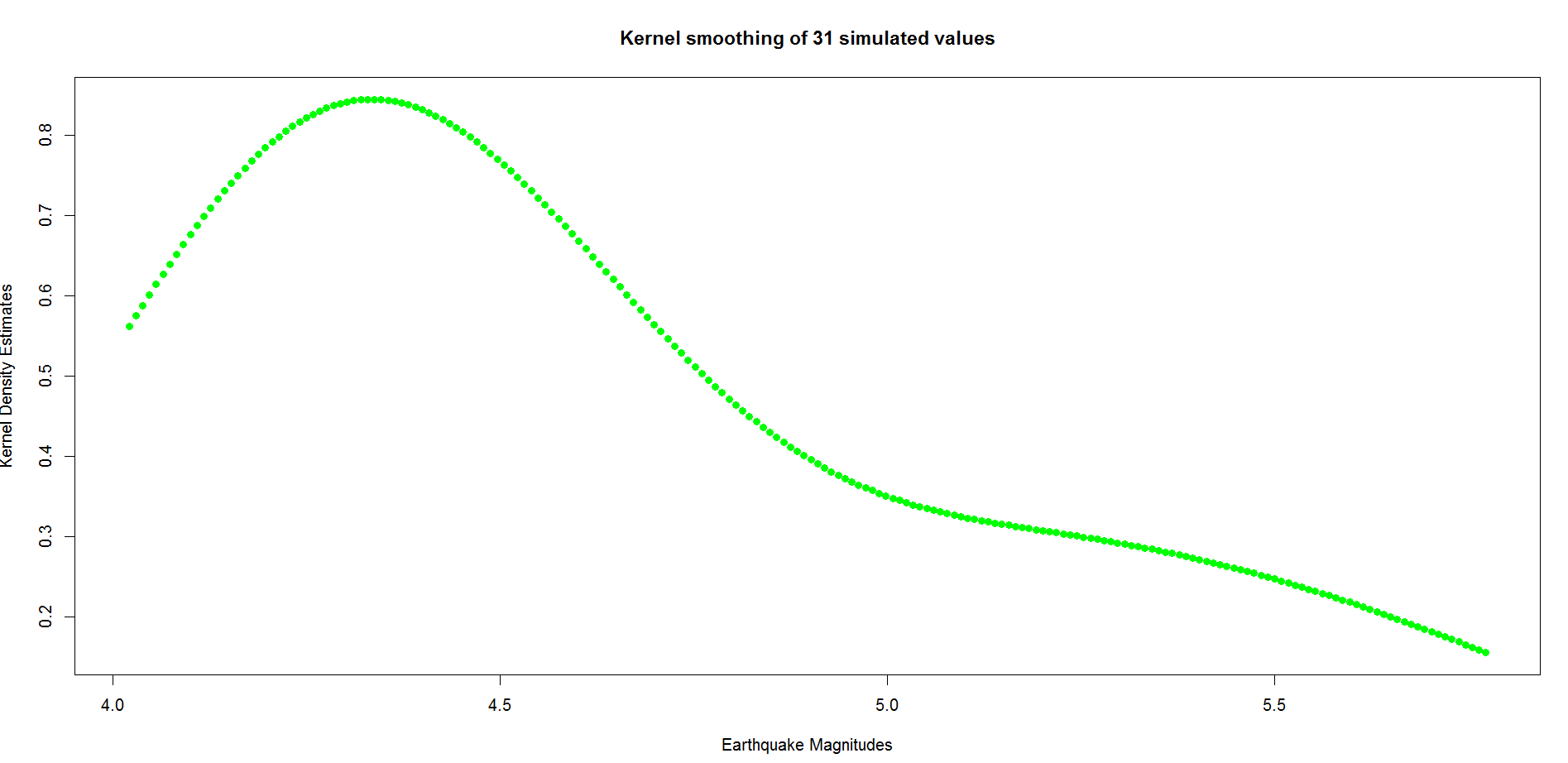
plot(q,yden,type="l",xlab="Earthquake Magnitudes",ylab="Kernel Density Estimates",col="red", main="Magnitude vs Density Estimates")

##Plotting with the desired label

c = max(yden);   
d = max(p) - min(p);  
g = 1/d;  
n = 31   
b = c\*d;   
x = c() ## these will contain the results.   
i = 0  
while(i < n)  
{   
x0 = runif(1)\*d + min(p) ## this simulates the uniform g(x) on (-2,6)   
fx0 = density(p,x0,bw1) ## this computes f(x0)   
if(runif(1) < (fx0 / (b\*g))){  
i = i+1   
x[i] = x0   
if(i/100 == floor(i/100)) cat(i, " ")   
}   
}   
x3 = seq(min(x),max(x),length=200);  
bw3 = bw.nrd(x);  
k3 = density2(x,x3,bw3)  
plot(x3,k3,xlab="Earthquake Magnitudes",ylab="Kernel Density Estimates",col="blue",pch=16,main="Kernel smoothing of 31 simulated values")

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*88

c = max(y);   
d = max(w) - min(w);  
b = c\*d;   
g = 1/d;  
n = 31   
x = c() ## these will contain the results.   
i = 0  
while(i < n)  
{   
x0 = runif(1)\*d + min(w) ## this simulates the uniform g(x) on (-2,6)   
fx0 = kerneldensity2(w,x0,bw1) ## this computes f(x0)   
if(runif(1) < (fx0 / (b\*g))){  
i = i+1   
x[i] = x0   
if(i/100 == floor(i/100)) cat(i, " ")   
}   
}   
x3 = seq(min(x),max(x),length=200);  
bw3 = bw.nrd(x);  
k3 = kerneldensity2(x,x3,bw3)  
plot(x3,k3,xlab="Earthquake Magnitudes",ylab="Kernel Density Estimates",col="blue",pch=16,main="Kernel smoothing of 31 simulated values")



2a)

housingPrices = read.table("LAhousingprices.txt",header=T)

y = as.numeric(as.vector(housingPrices[,3]))

x1 = as.numeric(as.vector(housingPrices[,4]))

x2 = as.numeric(as.vector(housingPrices[,7]))

x3 = as.numeric(as.vector(housingPrices[,9]))

z = !is.na(y+x1+x2+x3)

yy = y[z]

x11 = x1[z]

x22 = x2[z]

x33 = x3[z]

2b)

reg <<- lm(yy ~ x11+x22+x33)

coefficients(reg)[2]

Output- x11

0.007823431

2c)

srch <- function(i) {

yyT = yy[-i]

x11T = x11[-i]

x22T = x22[-i]

x33T = x33[-i]

yT=lm(yyT ~ x11T +x22T+x33T)

coefficients(yT)[2]

}

srch(1) - coefficients(reg)[2]

Output- x11T

0.0006900277

2d)

tuple <- function(){

influence = c()

for(i in c(1:217))

{

influence = append(influence,srch(i))

}

influence

}

influenceValues = tuple()

2e)

par(mar=c(5,5,5,5))

plot(c(1:217),influenceValues, type="p",xlab="i", ylab=expression(paste(hat(beta)\*1^"(-i)"," – ",hat(beta)\*1),"1"),cex.lab=0.9,pch=16,col="purple")