Problem Set 2

Tony Lashley

April 13, 2015

1 Machinery for the Schelling Model

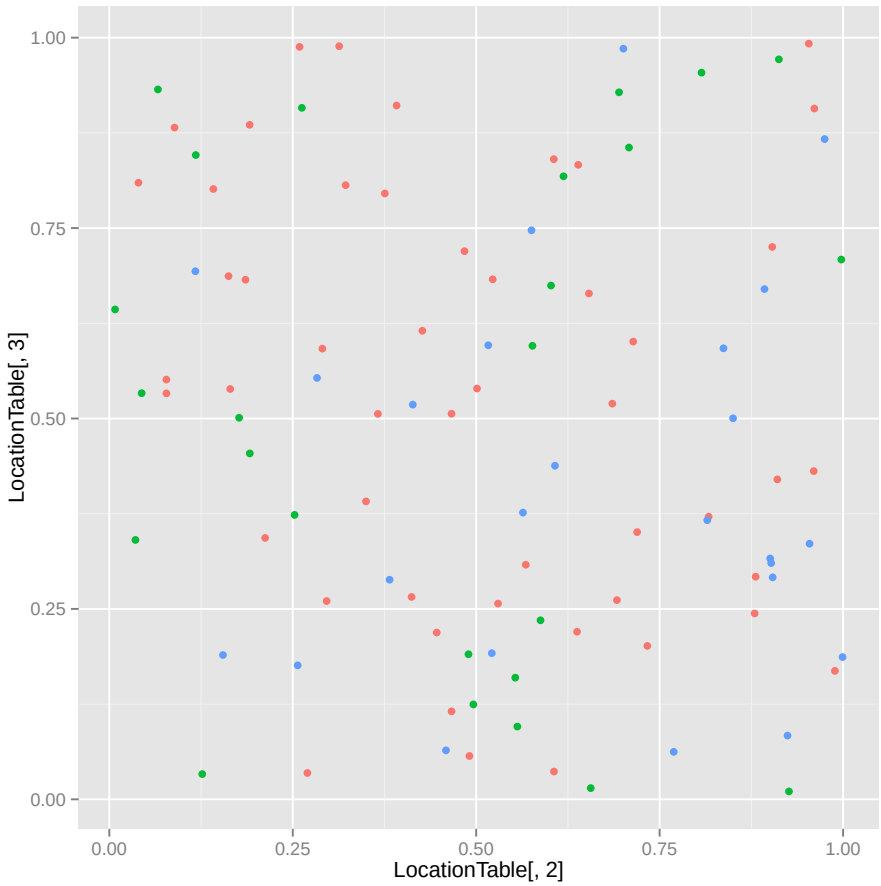
1.1 Write a function that calculates distances between coordinate points

```
individual <- c(x = 0,y = 0)
print(individual)

## x y
## 0 0

neighbors = matrix(1:8, ncol = 2, byrow = T)
print(neighbors)

##      [,1] [,2]
## [1,]    1    2
## [2,]    3    4
## [3,]    5    6
## [4,]    7    8

colnames(neighbors) <- c("X","Y")
dstances = matrix(ncol = 3, byrow = T)
colnames(dstances) <- c("X","Y", "Pythgorean")

f1 <- function(individual, neighbors){

  for (i in 1:nrow(neighbors)){

    neighbor_longitude = neighbors[i,1]
    ## Find your neighbor's longitude
    neighbor_latitude =  neighbors[i,2]
    ## Find your neighbor's latitude
    individual_longitude = individual[1]
    ## Find your own longitude
```

```r
    individual_latitude =  individual[2]
    ## Find your own latitude

    lftrghtdstance = abs(neighbor_longitude - individual_longitude)
    ## Find east/west distance between indiv. and neighbor
    updowndstance = abs(neighbor_latitude - individual_latitude)
    ## Find north/south distance between indiv. and neighbor
    pyth = sqrt(((lftrghtdstance)^2) + ((updowndstance)^2))
    ## Find Euclidian distance
    currentdistance = c(lftrghtdstance,updowndstance,pyth)
    ## Make vector with Manhattan and Euclidian distances


    dstances <- rbind(dstances, currentdistance)
    ## Add vector as row in matrix of distances
  }

  return(dstances)
}

f1(individual, neighbors)

##                   X  Y Pythgorean
##                  NA NA         NA
## currentdistance  1  2   2.236068
## currentdistance  3  4   5.000000
## currentdistance  5  6   7.810250
## currentdistance  7  8  10.630146
```

1.2 Write a function that simulates Schelling's Segregation model

```r
library(RANN)
library(ggplot2)
library(reshape2)

testRacialPreferenceTable <- matrix(1:15, ncol = 5, nrow = 3)
testRacialPreferenceTable[1,] <- c("R",1, 50, 5, 2)
testRacialPreferenceTable[2,] <- c("G", 0, 25, 5, 2)
testRacialPreferenceTable[3,] <- c("B", -1, 25, 5, 2)
colnames(testRacialPreferenceTable) <- c("Color", "Value", "Pop.", "Test Pool Size", "Racial

nR <- as.numeric(testRacialPreferenceTable[1,"Pop."])
nG <- as.numeric(testRacialPreferenceTable[2,"Pop."])
```

```r
nB <- as.numeric(testRacialPreferenceTable[3,"Pop."])

n <- sum(nR + nG + nB)
## Find total population from summing each racial population

Schelling <- function(racialPreferenceTable = testRacialPreferenceTable){
  set.seed(20016)
  LocationTable <- matrix(ncol = 3)
  ## Initalizing table for initial neighborhood coordinates

  for (i in 1:nR){
    x <- runif(1, min=0, max=1)
    ## Generate random X coordinate between 0 and 1 for point
    y <- runif(1, min=0, max=1)
    ## Generate random Y coordinate between 0 and 1 for point
    currentpointR = c(1,x,y)
    ## Create vector with point coordinates, labeling point as red
    LocationTable <- rbind(LocationTable, currentpointR)
    ## Add red point to table of all neighborhood coordinates
  }

  for (i in 1:nG){
    x <- runif(1, min=0, max=1)
    y <- runif(1, min=0, max=1)

    currentpointG = c(0,x,y)

    LocationTable <- rbind(LocationTable, currentpointG)
  }

  for (i in 1:nB){
    x <- runif(1, min=0, max=1)
    y <- runif(1, min=0, max=1)
    currentpointB = c(-1,x,y)

    LocationTable <- rbind(LocationTable, currentpointB)
  }

  LocationTable <- LocationTable[-1,]

  p <- qplot(x = LocationTable[,2], y = LocationTable [,3], col = ifelse(LocationTable[,1] <


  print(p)
}
```

```r
Schelling(testRacialPreferenceTable)
```