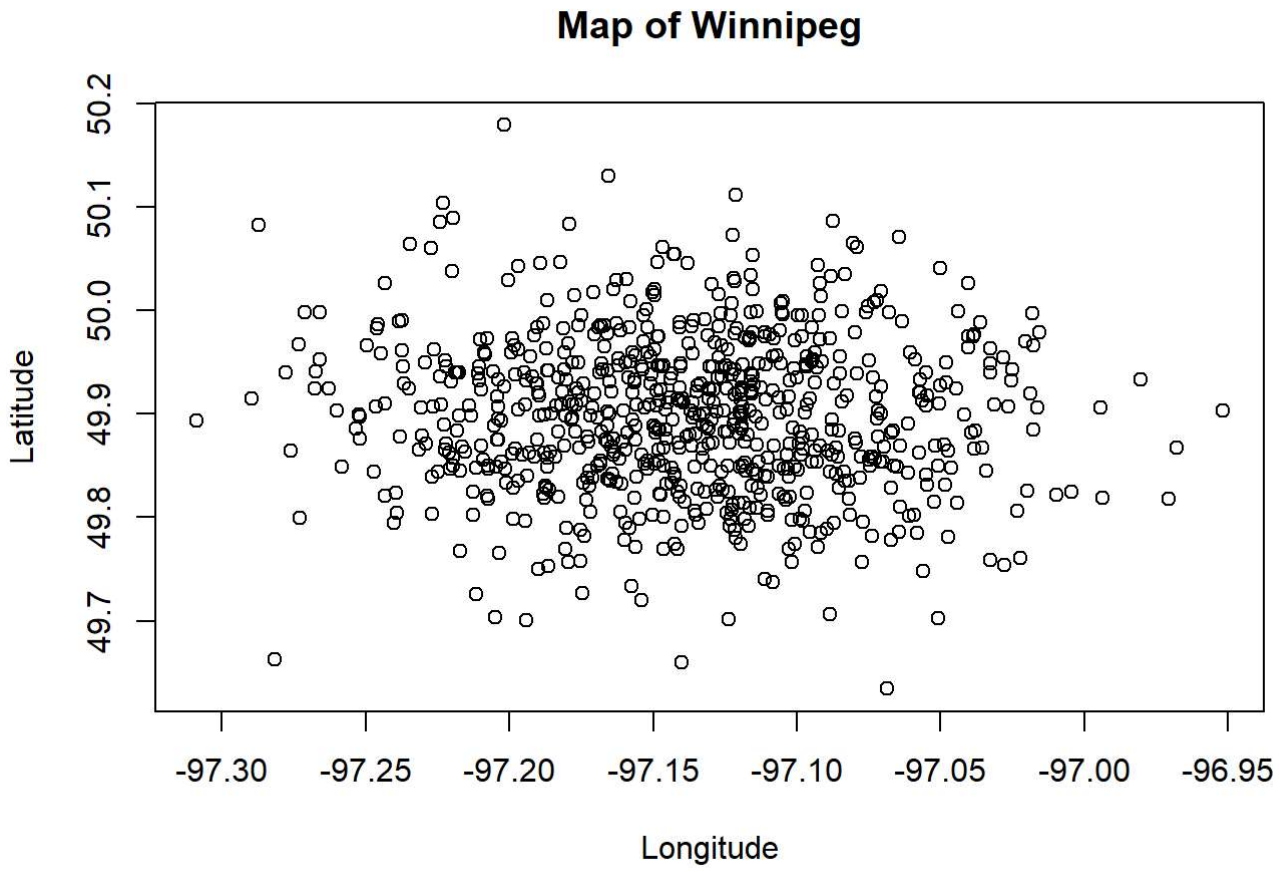
Ryne Swift Stat 3904 Project

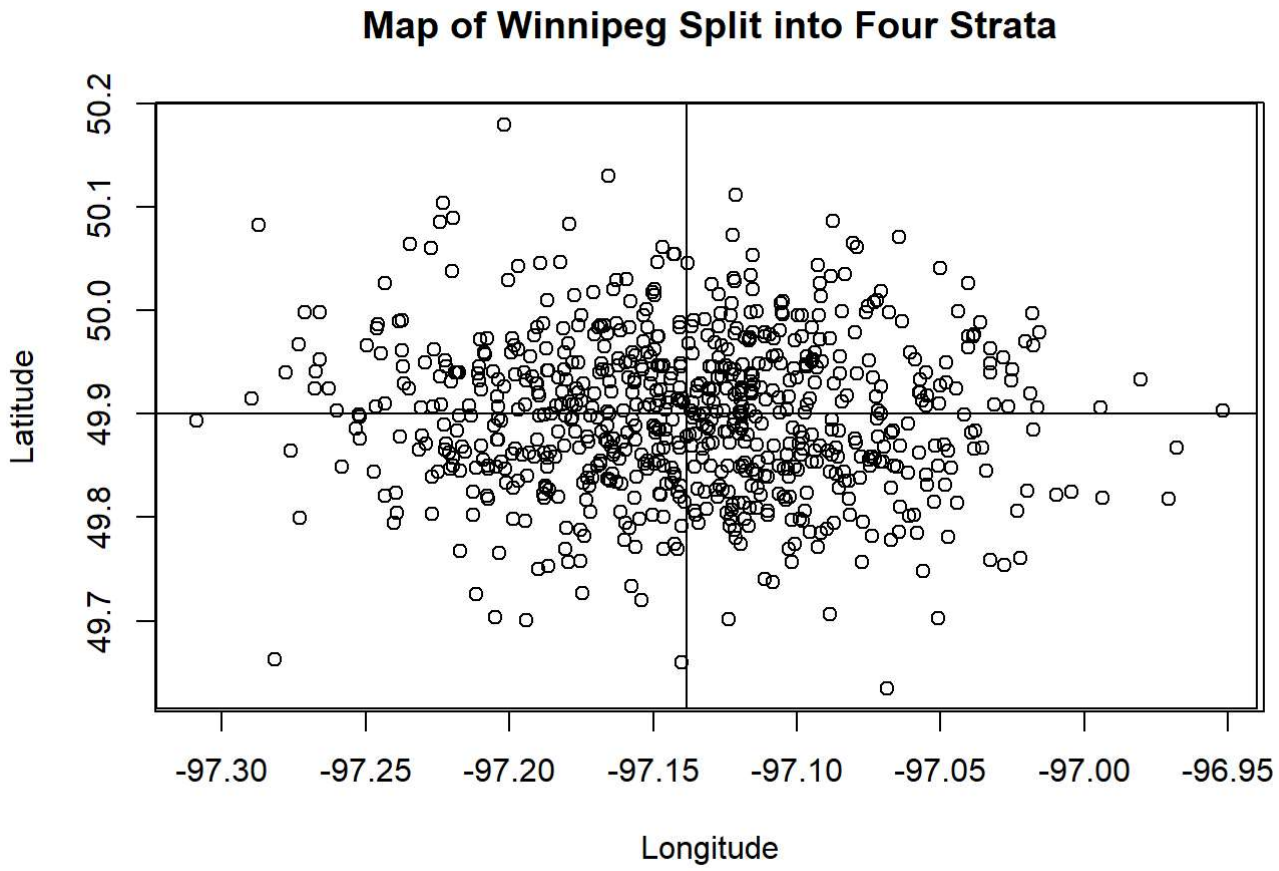
Bivariate Normal Map of Winnipeg

```
#install.packages("mvtnorm")
library(mvtnorm)


set.seed(3006)
#Setting this seed as it yields a map that resembles Winnipeg.


#Set to 750 because the population of Winnipeg was around 750,000 at the time
population_size<- 750

#The longitude and latitude of Portage and Main. This will be the center of our data and plot
mean_coordinates<-c(49.8955,-97.1385)
cov_matrix<- matrix(c(0.006157, 0, 0, 0.0035),nrow=2,ncol=2,byrow=TRUE)


#Creating a multivariate normal distribution of sample points, which represent people. We will u
se these points and their distances from each other to simulate disease spread.
samples <-rmvnorm(population_size, mean_coordinates, cov_matrix)

plot(samples[, 2], samples[, 1], xlab = "Longitude", ylab="Latitude", main="Map of Winnipeg")
```

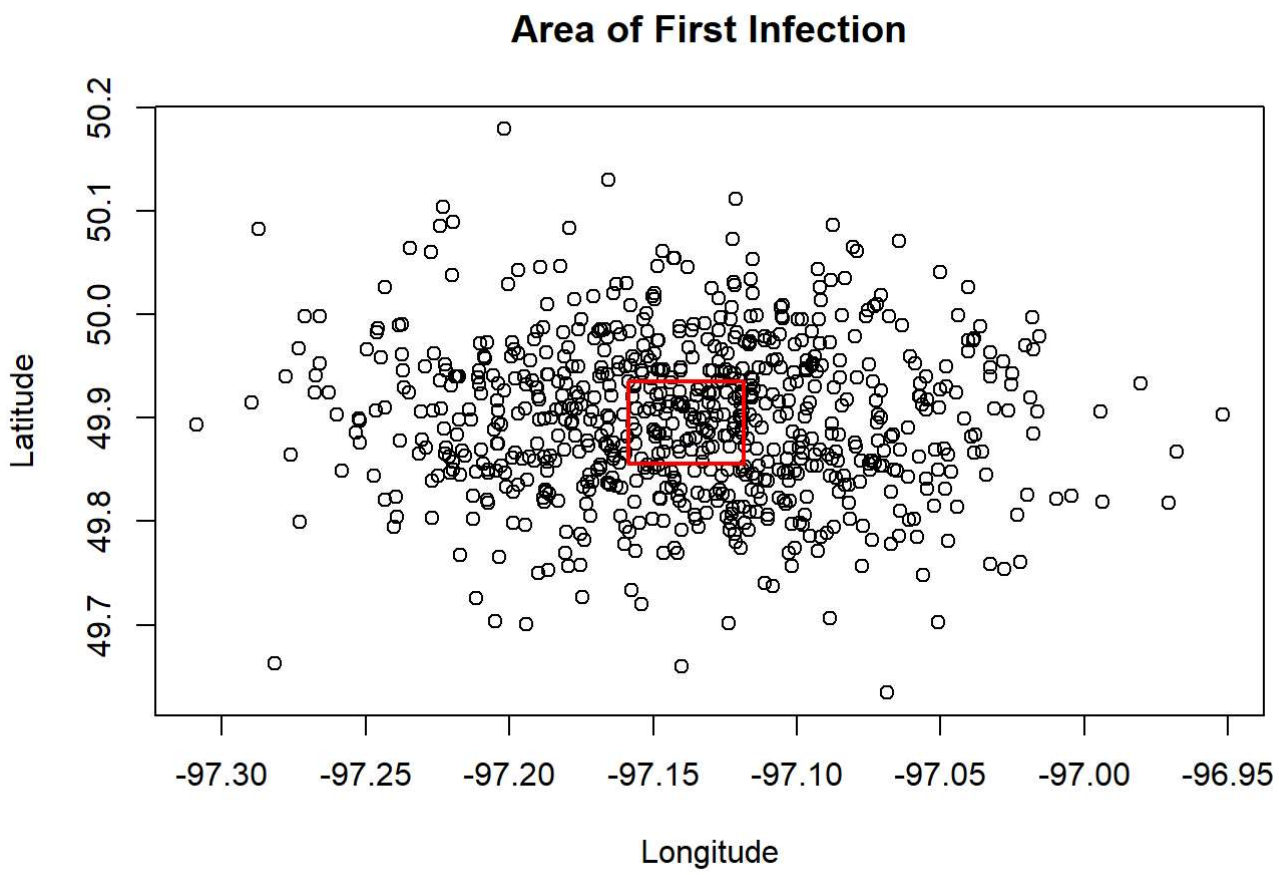


Bivariate Normal with Regions

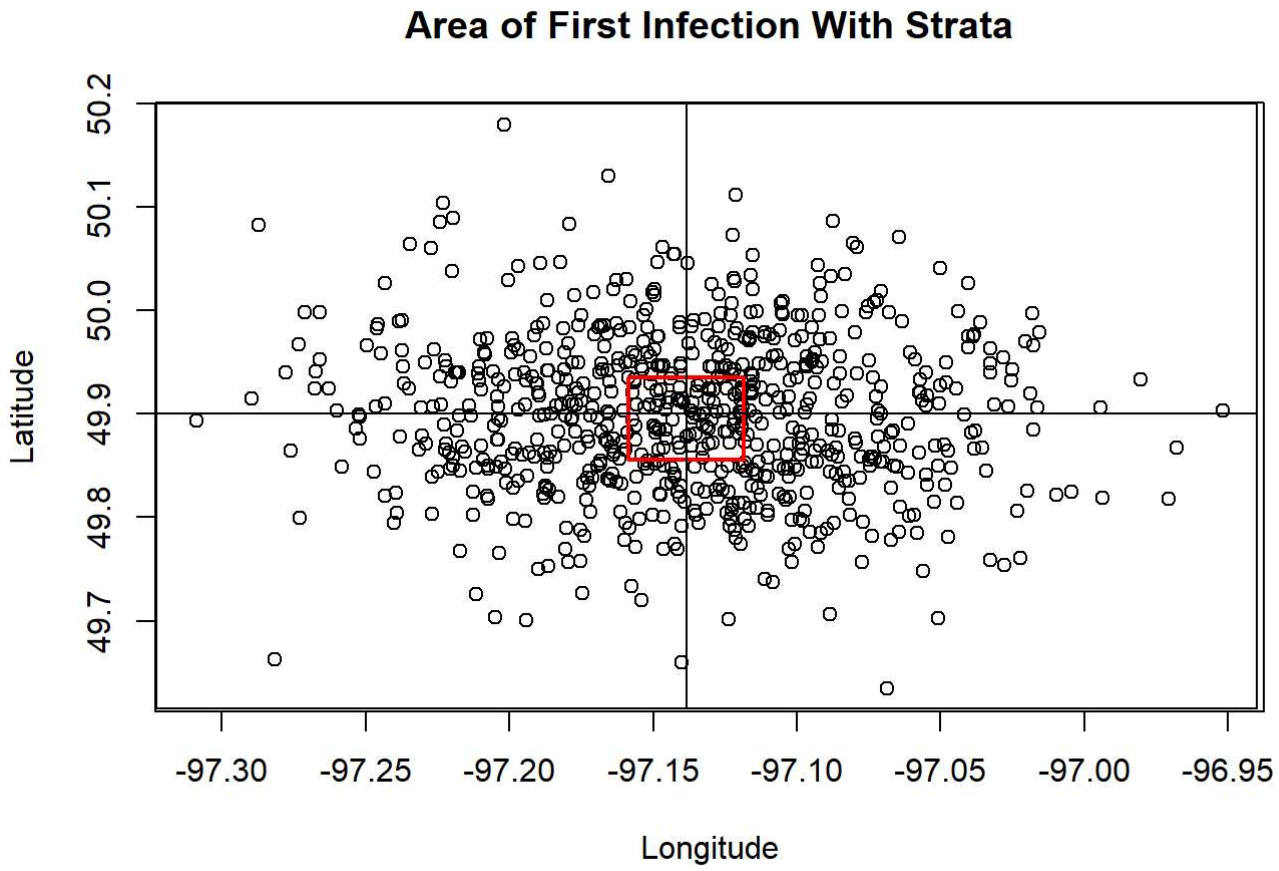
```
#Dividing Winnipeg into four sections/strata based on longitude and latitude
longitudeNW=c(49.9, 49.9,  50.2, 50.2, 49.9)
latitudeNW=c(-97.323, -97.138483, -97.138483, -97.323, -97.323)
coordinatesNW=data.frame(longitudeNW, latitudeNW)


longitudeNE=c(49.9, 49.9, 50.2, 50.2,  49.9)
latitudeNE=c(-97.138483, -96.94, -96.94, -97.138483, -97.138483)
coordinatesNE=data.frame(longitudeNE, latitudeNE)


longitudeSW=c(49.9, 49.9, 49.615, 49.615,  49.9)
latitudeSW=c(-97.138483, -97.323, -97.323, -97.138483,  -97.138483)
coordinatesSW=data.frame(longitudeSW, latitudeSW)


longitudeSE=c(49.9, 49.9, 49.615, 49.615, 49.9)
latitudeSE=c(-97.138483, -96.94, -96.94, -97.138483, -97.138483)
coordinatesSE=data.frame(longitudeSE, latitudeSE)


#Plot of Winnipeg divided into 4 Strata
plot(samples[, 2], samples[, 1], xlab = "Longitude", ylab="Latitude", main="Map of Winnipeg Spli
t into Four Strata")
lines(x=coordinatesNW$latitudeNW, y=coordinatesNW$longitudeNW, col="black")
lines(x=coordinatesNE$latitudeNE, y=coordinatesNE$longitudeNE, col="black")
lines(x=coordinatesSW$latitudeSW, y=coordinatesSW$longitudeSW, col="black")
lines(x=coordinatesSE$latitudeSE, y=coordinatesSE$longitudeSE, col="black")
```




```
#This is the area where we will be randomly infecting the first point. The area we will be rando
mly infecting is evenly distributed across the 4 strata.

longitudeCenter=c(49.935508, 49.935508, 49.855508, 49.855508, 49.935508)
latitudeCenter=c(-97.118483, -97.158483, -97.158483, -97.118483, -97.118483)
coordinatesCenter=data.frame(longitudeCenter, latitudeCenter)

plot(samples[, 2], samples[, 1], xlab = "Longitude", ylab="Latitude", main="Area of First Infect
ion")
lines(x=coordinatesCenter$latitudeCenter, y=coordinatesCenter$longitudeCenter, col="red", lwd=2)
```




```
plot(samples[, 2], samples[, 1], xlab = "Longitude", ylab="Latitude", main="Area of First Infect
ion With Strata")
lines(x=coordinatesCenter$latitudeCenter, y=coordinatesCenter$longitudeCenter, col="red", lwd=2)
lines(x=coordinatesNW$latitudeNW, y=coordinatesNW$longitudeNW, col="black")
lines(x=coordinatesNE$latitudeNE, y=coordinatesNE$longitudeNE, col="black")
lines(x=coordinatesSW$latitudeSW, y=coordinatesSW$longitudeSW, col="black")
lines(x=coordinatesSE$latitudeSE, y=coordinatesSE$longitudeSE, col="black")
```



Random First Infection Function

```
#Gets all the points in a square that is roughly in the center (the red square). This sets the b
oundaries of the area of first infection.
#Long_lat_df will be the dataframe containing the coordinates of all the points of Winnipeg.

centerPoints = function(long_lat_df){

  #long_lat_df[i,2] is longitude
  #long_lat_df[i,1] is latitude
  center_points_df = long_lat_df[(long_lat_df[, 2]<=-97.118483) & (long_lat_df[, 2]>-97.158483)
& (long_lat_df[, 1]>49.855508) & (long_lat_df[, 1]<49.935508),]

  colnames(center_points_df)=c("Longitude", "Latitude")

  return(center_points_df)
}



#We'll be randomly adding or subtracting a pair of randomly generated uniform values, divided by
25 and 50 respectively, from the very center coordinate of the population which will give us a r
andom coordinate. We will then measure which coordinate is closest to the newly generated random
coordinate. The point that is closest to the newly generated random coordinate will be the first
infected.

#This function randomly chooses a point in the center of the population to infect
randomInfect=function(df){

  #Gets all the points in a square that is roughly in the center (the red square).
  center_points=centerPoints(df)


  #Gets the row positions of all the center-points within the dataframe. This gets all the point
s within the area of first infection, which is the red square.
  centerPlacements= which((df[, 2]<=-97.118483) & (df[, 2]>-97.158483) & (df[, 1]>49.855508) &
(df[, 1]<49.935508))


  #Picking a random coordinate to infect first
  longitude_shift=runif(1, -1, 1)    #min=-1, and max=1 so we can randomly subtract or add
  latitude_shift=runif(1, -1, 1)

  longitude_shift=(4*longitude_shift)/100
  latitude_shift=(2*latitude_shift)/100

  randomShift=c(longitude_shift, latitude_shift)


  #Gets the very center of the square
  centerCoordinate=c(-97.138483, 49.895508)
```

```
  #Whichever point is closest to this coordinate here will be first infected
  random_first_infect_coordinate=centerCoordinate+randomShift


  longitude_distances=center_points[,1]-random_first_infect_coordinate[1]
  latitude_distances=center_points[,2]-random_first_infect_coordinate[2]

  abs_val_longitude_distances=abs(longitude_distances)
  abs_val_latitude_distances=abs(latitude_distances)


  #Getting the sums of the absolute values of the distances from the points to the coordinates o
f first infection
  sum_distances = abs_val_longitude_distances + abs_val_latitude_distances


  #Gets the lowest sum of the absolute values of the coordinates. The coordinate with the lowest
sum will be our first infected point. This is because it will have the least distance from the c
oordinate chosen to assign a first infection.
  place=centerPlacements[which.min(sum_distances)]

  #returns the placement of the closest point so we can infect them
  return(place)
}
```

These two functions were made by my professor. He allowed us to use them for our project. Geometric and Exponential Epidemic Model Functions

```
#An infected person will remain infectious for 3 time points
#We will be simulating 20 time points after the first infection

#Geometric Model for simulating infections
epigenGeometric <-function(pop,alpha,beta,interval){
  x <-pop[,1]
  y <- pop[,2]
  length <- nrow(pop)
  id <- 1:length
  inftime <- rep(0, length)

  distance <- inftime
  k <- randomInfect(pop)
  inftime[k] <- 1
  for(T in 1:interval){
    for(i in 1:length){
      if(inftime[i]==0){
        d<-0
        for(j in 1:length){
          if(inftime[j]!=0 && inftime[j] <= T &&inftime[j]>=(T-2)){
            distance[i] <-sqrt((x[i]-x[j])^2+(y[i]-y[j])^2)
            d <- d+ distance[i]^(-beta)
          }
        }
        pit <- 1-exp(-alpha*d)
        u <- runif(1,0,1)
        if(pit > u){
          inftime[i] <-T+1
        }
      }
    }
  }
  epidemic <- data.frame(id, x, y, inftime)
  return(epidemic)
}


#Exponential Model for simulating infections
epigenExponential <-function(pop,alpha,beta,interval){
  x <-pop[,1]
  y <- pop[,2]
  length <- nrow(pop)
  id <- 1:length
  inftime <- rep(0, length)

  distance <- inftime
  k <- randomInfect(pop)
  inftime[k] <- 1
  for(T in 1:interval){
    for(i in 1:length){
      if(inftime[i]==0){
        d<-0
```

```
        for(j in 1:length){
          if(inftime[j]!=0 && inftime[j] <= T &&inftime[j]>=(T-2)){
            distance[i] <-sqrt((x[i]-x[j])^2+(y[i]-y[j])^2)

            d <- d+ exp(distance[i]*(-beta))
          }
        }
        pit <- 1-exp(-alpha*d)
        u <- runif(1,0,1)
        if(pit > u){
          inftime[i] <-T+1
        }
      }
    }
  }
  epidemic <- data.frame(id, x, y, inftime)
  return(epidemic)
}
```

Colouring the Plot for Different Regions

```
#In this function, we assume that we will have called a plot before calling these functions.
#This function displays the severity of an epidemic by region

#long_coords and lat_coords mean longitude coordinates and latitude coordinates
map_severity = function(num_infected, long_coords, lat_coords){
  if(num_infected == 0){
    polygon(x = long_coords, y = lat_coords, col = "white")
  }

  else if(num_infected >= 1 && num_infected <= 48){
    polygon(x = long_coords, y = lat_coords, col = "green")
  }

  else if(num_infected >= 49 && num_infected <= 97){
    polygon(x = long_coords, y = lat_coords, col = "yellow")
  }

  else if(num_infected >= 98 && num_infected <= 145){
    polygon(x = long_coords, y = lat_coords, col = "orange")
  }

  else {
    polygon(x = long_coords, y = lat_coords, col = "red")
  }
}


#Displays heat maps of the infections throughout the course of the epidemic
heatMapDisplayer = function(plot_interval, plot_interval_start, epi_time_start, epi_time_end, in
ftime, epidemic_df, titles){


  latitudeNW = c(49.9, 49.9, 50.2, 50.2, 49.9)
  longitudeNW = c(-97.323, -97.138483, -97.138483, -97.323, -97.323)

  latitudeNE = c(49.9, 49.9, 50.2, 50.2, 49.9)
  longitudeNE = c(-97.138483, -96.94, -96.94, -97.138483, -97.138483)

  latitudeSW = c(49.615, 49.615, 49.9, 49.9, 49.615)
  longitudeSW = c(-97.323, -97.138483, -97.138483, -97.323, -97.323)

  latitudeSE = c(49.615, 49.615, 49.9, 49.9, 49.615)
  longitudeSE = c(-97.138483, -96.94, -96.94, -97.138483, -97.138483)


  #Iterating through all the time points of the epidemic
  for(i in 1:epi_time_end){
    if(((i %% plot_interval) == plot_interval_start && (i > 1)) || (i == epi_time_end)){

      # Compute total infections up to the current time point
      totalPerStrata = strata_infected(inftime, epidemic_df, epi_time_start, i)
```

```
      # Plotting code without dev.new()
      plot(epidemic_df[, 2], epidemic_df[, 1], xlab = "Longitude", ylab = "Latitude")
      title(main = titles[i-2])

      # North-West region
      lines(x = longitudeNW, y = latitudeNW, col = "purple")
      map_severity(totalPerStrata[3], longitudeNW, latitudeNW)

      # North-East region
      lines(x = longitudeNE, y = latitudeNE, col = "green")
      map_severity(totalPerStrata[4], longitudeNE, latitudeNE)

      # South-West region
      lines(x = longitudeSW, y = latitudeSW, col = "orange")
      map_severity(totalPerStrata[2], longitudeSW, latitudeSW)

      # South-East region
      lines(x = longitudeSE, y = latitudeSE, col = "blue")
      map_severity(totalPerStrata[1], longitudeSE, latitudeSE)

      legend(
        x = -97.045, y = 50.2,
        legend = c("0 Infections", "1-48 Infections", "49-97 Infections", "98-145 Infections",
"145 or more Infections"),
        fill = c("white", "green", "yellow", "orange", "red"),
        cex = 0.75
      )
    }
  }
}


#plot_interval dictates how often a plot is generated. Enter it as 2 if you want to skip every s
econd plot
#Displays infection maps of the infections of the epidemic
activeInfectionsDisplayer=function(plot_interval, plot_interval_start, lowerLimit, epi_length, i
nftime, epidemic_df, title){

  #Iterates through all the time points of the epidemic
  for(i in 1:epi_length){
    if(((i%%plot_interval)==plot_interval_start)&&(i>lowerLimit)||(i==epi_length)){
      two_time_lag= which(inftime==(i-2))
      time_lag= which(inftime==(i-1))
      current_time= which(inftime==(i))

      plot(epidemic_df[, 2], epidemic_df[, 1], xlab="Longitude", ylab="Latitude")
      title(main=title[i-2])
      points(epidemic_df[two_time_lag,2], epidemic_df[two_time_lag,1], pch=20, col="purple")
      points(epidemic_df[time_lag,2], epidemic_df[time_lag,1], pch=20, col="pink")
      points(epidemic_df[current_time,2], epidemic_df[current_time,1], pch=20, col="blue")
```

```
      #i>=5 as that is the minimum amount of time that someone will have recovered, and that is
within our plot cycle of 2
      if(i>=5){
        recovered = which((inftime>=1) & (inftime<(i-2)))
        points(epidemic_df[recovered,2], epidemic_df[recovered,1], pch=20, col="black")
      }
      leg=c("Infected 2 TPA", "Infected 1 TPA", "Newly Infected", "Recovered")
      legend(x=-97.03, y=50.2, legend=leg, pch=20, col=c("purple", "pink", "blue", "black"))
    }
  }
}
```

Finding the population of each strata, and finding the number of infected people in each strata

```
#Finds the total population of each strata. Can use to find the total number of people in Winnip
eg, and the total number of people infected in each strata
pop_of_strata=function(long_lat_df){

  total_SE = nrow(which((long_lat_df[,2] >= -97.138483) & (long_lat_df[,1]<=49.9)))
  total_NE = nrow(which((long_lat_df[,2] >= -97.138483) & (long_lat_df[,1]>49.9)))

  total_SW = nrow(which((long_lat_df[,2] <= -97.138483) & (long_lat_df[,1]<=49.9)))
  total_NW = nrow(which((long_lat_df[,2] <= -97.138483) & (long_lat_df[,1]>49.9)))

  population=c(total_SE, total_SW, total_NW, total_NE)
  return(population)
}
pop_of_strata(samples)
```

```
## NULL
```

```
#193 people live in the southeast; 181 people live in the southwest; 187 people live in the nort
hwest; 189 people live in the northeast.



#Set lowerLimit to null to get how many have contracted the disease at and before a time point
#If lowerLimit isn't null, gets how many are infected in a specified interval of time

#The function gets how many people are infected in each strata
strata_infected = function(infections, samps, lowerLimit, upperLimit) {


  if (is.null(lowerLimit)){
    #Filter infections below the time limit
    valid_infections = (infections <= upperLimit)
  }

  else{
    #Filter infections within the time range
    valid_infections = (infections >= lowerLimit) & (infections <= upperLimit)
  }


  #Subset samps where infections are within the time range
  valid_samps = samps[valid_infections, drop = FALSE]


  #Sometimes R collapses valid_samps into a vector. The if statement below is for correcting whe
n this does happen
  if (is.null(dim(valid_samps))) {
    valid_samps <- matrix(valid_samps, ncol = 2)
   }

  longitude = -97.138483
  latitude = 49.9



  #Infection counts per region
  SE = sum((valid_samps[, 2] >= longitude) & (valid_samps[, 1] <= latitude), na.rm = TRUE)
  NE = sum((valid_samps[, 2] >= longitude) & (valid_samps[, 1] > latitude), na.rm = TRUE)
  SW = sum((valid_samps[, 2] < longitude) & (valid_samps[, 1] <= latitude), na.rm = TRUE)
  NW = sum((valid_samps[, 2] < longitude) & (valid_samps[, 1] > latitude), na.rm = TRUE)


  infected=c(SE, SW, NW, NE)
  return(infected)
}
```

Epidemic Summary Functions

```
#Finds the length of an epidemic by seeing if there are any active infections in the strata. Thi
s function is to be used in a loop.
#The first time point without any active infections in any strata will be declared the length of
the epidemic. If an epidemic still has active infections at time point 20, time point 20 will be
the length of the epidemic.

epiLength=function(infected_per_region, upperLimit){

  #Renaming the variable for clarity
  ipr = infected_per_region

  #Runs true if no one is infected in any of the strata/
  if((ipr[1]==0)&&(ipr[2]==0)&&(ipr[3]==0)&&(ipr[4]==0)){
    return(upperLimit)
  }

  #The disease has not stopped spreading throughout the simulation. So, we say the epidemic leng
th of that disease is 20.
  else if(upperLimit==20){
    return(upperLimit)
  }

  #Returns 0, meaning to continue the epidemic simulation
  else{
    return(0)
  }
}



#Finds the peak of an epidemic by finding the time point with the most new infections.
#inftime is the time at which a point got infected
peak=function(inftime, epi_length){

  #Will be a count of the infections at each time point
  count=NULL


  for(i in 1:epi_length){
    tempCount = sum(inftime == i)
    count=c(count, tempCount)
  }

  #Will store the number of infections. Number of infections can't be negative, so start it at -
1
  highest=-1


  #Saves which time point has the most infections
  placement=NULL
  for(i in 1:length(count)){
    num_infections=count[i]
```

```
    if(num_infections>highest){
      highest=num_infections
      placement=i
    }
  }
  return(placement)
}



#Prints 5 values which summarize the course of an epidemic
epiSummary=function(samps, inftime, epi_length){


  #3 because that is the minimum amount of time for an epidemic to die out
  for(upper_limit in 3:epi_length){
    lower_limit = upper_limit - 2
    activeInfected=strata_infected(inftime, samps, lower_limit, upper_limit)

    length=epiLength(activeInfected, upper_limit)


    #If length is 0, breaks out of the for loop as the epidemic ended
    if(length!=0){
      #Stops the loop as the epidemic has ended
      break
    }
  }
  totalPerStrata=strata_infected(inftime, samps, NULL, epi_length)
  total_infected=sum(totalPerStrata)

  epi_peak=peak(inftime, epi_length)


  #Returns a summary of the epidemic
  summary=c(epi_peak, length, total_infected)
  return(summary)
}
```

Epidemic Simulating Functions

```
#Samps, is our dataframe of x and y coordinates; replications is how many epidemics we want to g
enerate; alpha and beta are the infection kernel parameters; time is how long we want an epidemi
c to run for.
#set exponential_simulation to true if you want to simulate an exponential distribution of a pan
demic, and false if you want a geometric distribution of one.
epi_simulator=function(samps, replications, alpha, beta, epi_length, exponential_simulation){

  total_peak=0; total_epi_length=0; total_total_infection=0

  for(h in 1:replications){

    if (exponential_simulation==TRUE){
      epidemic=epigenExponential(samps, alpha, beta, epi_length)
    }

    else{
      epidemic=epigenGeometric(samps, alpha, beta, epi_length)
    }

    epidemicSummary=epiSummary(samps, epidemic$inftime, epi_length)


    total_peak=total_peak+epidemicSummary[1]
    total_epi_length=total_epi_length+epidemicSummary[2]
    total_total_infection=total_total_infection+epidemicSummary[3]
  }

  #Gets the average peak, length and number of infections across all epidemics
  avgPeak=total_peak/replications
  avgEpiLength=total_epi_length/replications
  avgTotalInfection=total_total_infection/replications


  summary=c(alpha, beta, avgPeak, avgEpiLength, avgTotalInfection)
  return(summary)
}
```

Plot Displaying Functions for Both Geometric and Exponential Distributions

```
#The three functions below are very similar. The only difference is that the colour of the lines
of the plots are different.
#plot_colour is a string, which will be the colour that you want the plot to be


#num_epi dictates how many epidemics you want to simulate
#Set exponential_simulation to true if you want to simulate an exponential distribution of a pan
demic, and false if you want a geometric distribution of one.
epi_plot=function(samps, num_epi, alpha, beta, epi_length, plot_colour, exponential_simulation){


  for(h in 1:num_epi){

    if (exponential_simulation==TRUE){
      epidemic=epigenExponential(samps, alpha, beta, epi_length)
    }

    else{
      epidemic=epigenGeometric(samps, alpha, beta, epi_length)
    }

    infection_time=epidemic$inftime
    inftimes=NULL


    for(i in 1:epi_length){
      count=sum(infection_time==i)
      inftimes=c(inftimes, count)
    }

    #We do this if statement so our plot is properly labelled
    if(h!=1){
      par(new=TRUE)
      plot(inftimes, type="l", col=plot_colour, xaxt='n', yaxt='n', ann=FALSE, ylim=c(0,200))
    }
    #Labels the x and y axis
    else{
      par(new=TRUE)
      plot(inftimes, type="l", col=plot_colour, ann = FALSE, ylim=c(0,200))
    }
  }
}
```

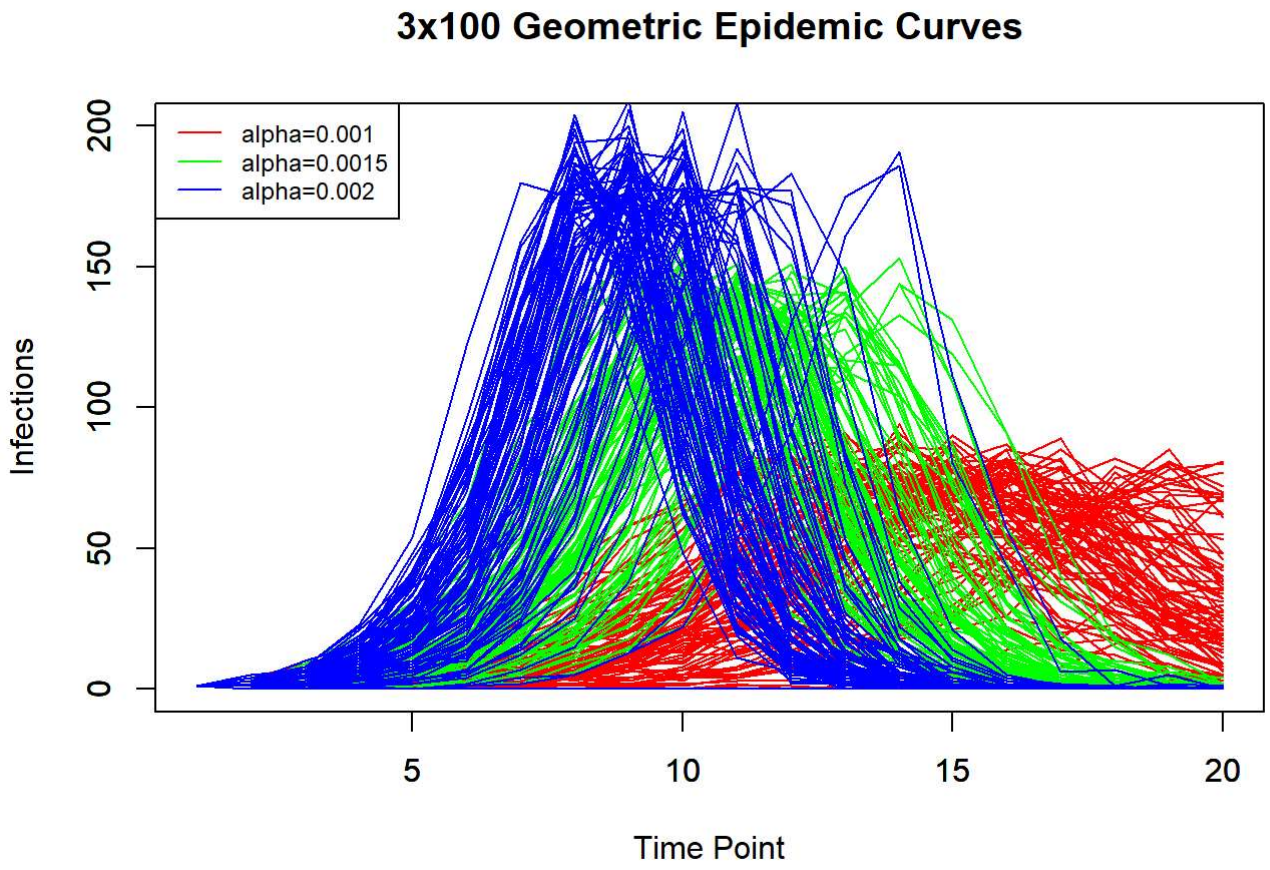
```
#samps will be the points that can be infected
#alpha will be the alpha value(s). set alpha to be just one value if you want it to be fixed
#beta will be the beta value(s). set beta to be just one value if you want it to be fixed
#epi_length will be a number, and dictates the length of each epidemic
#is_exp_epi is a boolean, and will instruct whether the epidemic was geometrically or exponentia
lly created

#epi_sims simulates both geometric and exponential distributions
epi_sims = function(samps, num_epidemics, alphas, betas, epi_length, is_exp_epi){

  epi_dataframe = data.frame(row.names = c("alpha", "beta", "Average Peak", "Average Epi Lengt
h", "Average Infections"))


  colnames_epi_df = c()


  #Records how many times the loops have executed. Necessary to correctly input data into epi_da
taframe
  iteration = 0

  for(i in 1:length(alphas)){
    for(j in 1:length(betas)){

      iteration = iteration + 1
       colnames_epi_df = c(colnames_epi_df, paste0("Epi ", iteration, " alpha=", alphas[i], " be
ta=", betas[j]))



      epi = epi_simulator(samps, num_epidemics, alphas[i], betas[j], epi_length, is_exp_epi)
      epi_dataframe[iteration] = epi
    }
  }
  colnames(epi_dataframe) = colnames_epi_df
  return(epi_dataframe)
}
```

Geometric Epidemic Curves Beta(Beta = 0.0005) Fixed

```
#Betas Fixed
set.seed(3000)
epi_plot(samples, 100, 0.001, 0.0005, 20 ,"red", F)
```

```
## Warning in par(new = TRUE): calling par(new=TRUE) with no plot
```

```
epi_plot(samples, 100, 0.0015, 0.0005, 20 ,"green", F)
epi_plot(samples, 100, 0.002, 0.0005, 20 ,"blue", F)
title(main = "3x100 Geometric Epidemic Curves", xlab = "Time Point", ylab="Infections"); legend
("topleft", legend=c("alpha=0.001", "alpha=0.0015", "alpha=0.002"), lwd=1, col=c("red", "green",
"blue"), cex=0.75)
```

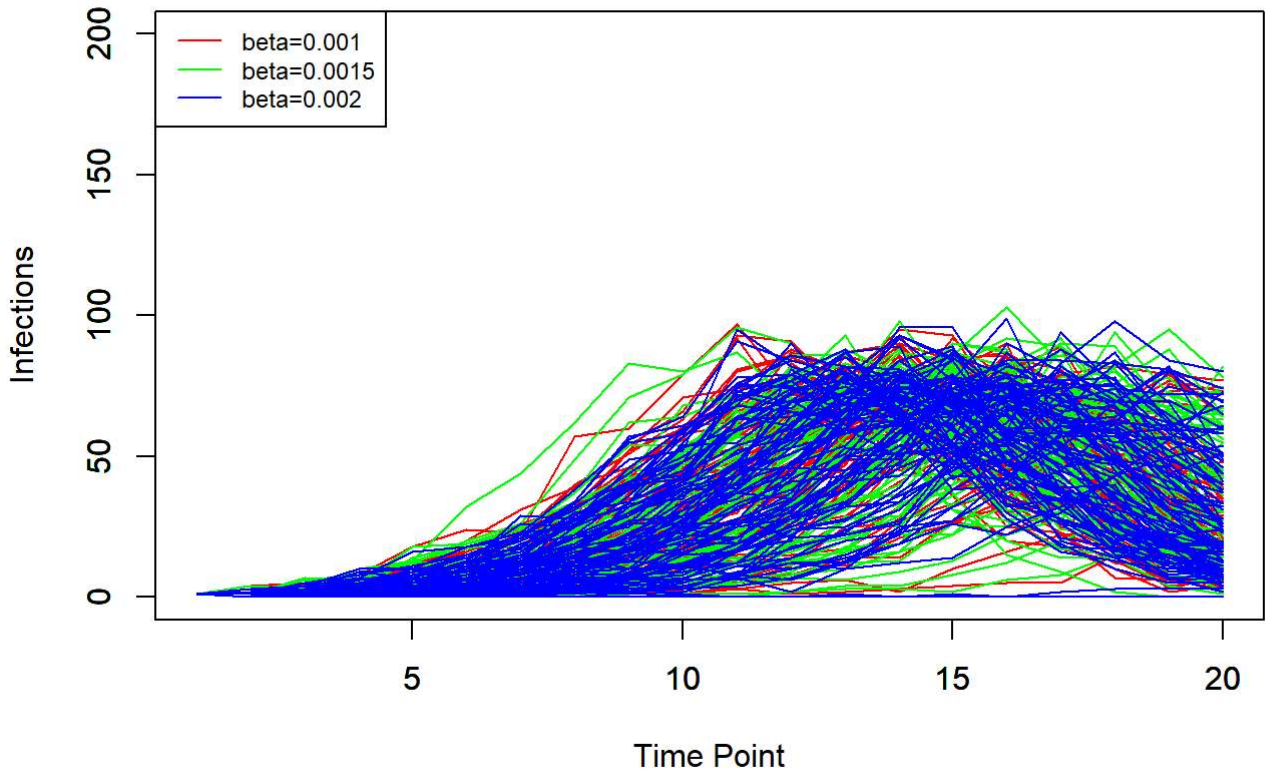


Geometric Epidemic Curves Alpha (Alpha = 0.001) Fixed

```
set.seed(3000)
epi_plot(samples, 100, 0.001, 0.001, 20, "red", F)
```

```
## Warning in par(new = TRUE): calling par(new=TRUE) with no plot
```

```
epi_plot(samples, 100, 0.001, 0.0015, 20, "green", F)
epi_plot(samples, 100, 0.001, 0.002, 20, "blue", F)
title(main = "3x100 Geometric Epidemic Curves", xlab = "Time Point", ylab="Infections"); legend
("topleft", legend=c("beta=0.001", "beta=0.0015", "beta=0.002"), lwd=1, col=c("red", "green", "b
lue"), cex=0.75)
```



Result Tables for Geometric Curves

```
#Same betas( beta=0.0005), different alphas
set.seed(3000)


geo_alpha_values = c(0.001, 0.0015, 0.002)
geo_beta_fixed = epi_sims(samples, 100, geo_alpha_values, 0.0005, 20, F)
geo_beta_fixed
```

```
##                    Epi 1 alpha=0.001 beta=5e-04 Epi 2 alpha=0.0015 beta=5e-04
## alpha                                    0.0010                         0.0015
## beta                                     0.0005                         0.0005
## Average Peak                            13.6200                        10.6200
## Average Epi Length                      17.8200                        19.2100
## Average Infections                     730.0900                       749.7800
##                    Epi 3 alpha=0.002 beta=5e-04
## alpha                                    0.0020
## beta                                     0.0005
## Average Peak                             9.1800
## Average Epi Length                      17.8900
## Average Infections                     750.0000
```

```
geo_beta_values = c(0.001, 0.0015, 0.002)
geo_alpha_fixed = epi_sims(samples, 100, 0.001, geo_beta_values, 20, F)
geo_alpha_fixed
```

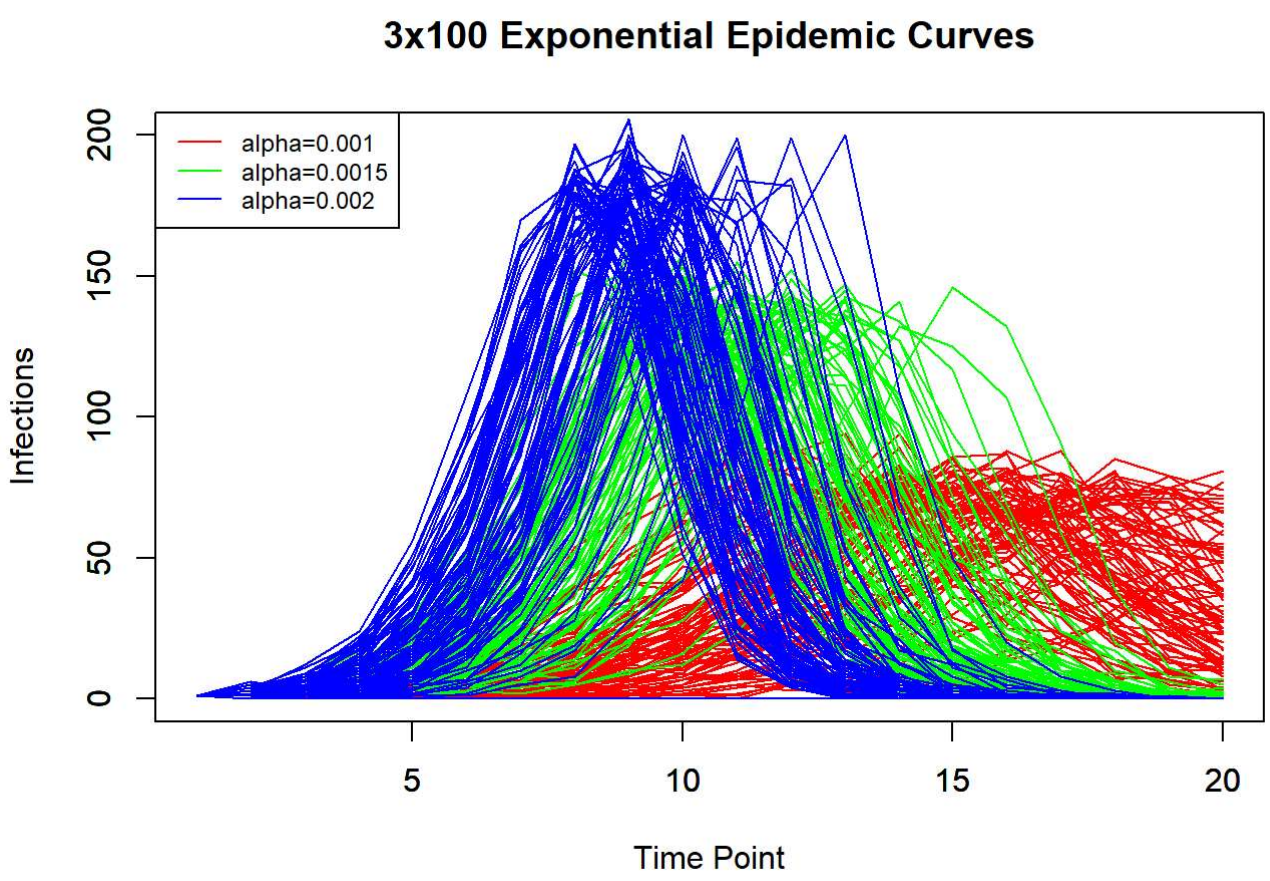
```
##                    Epi 1 alpha=0.001 beta=0.001 Epi 2 alpha=0.001 beta=0.0015
## alpha                                      0.001                         0.0010
## beta                                       0.001                         0.0015
## Average Peak                              14.160                        13.1000
## Average Epi Length                        18.610                        17.4400
## Average Infections                       730.820                       732.2300
##                    Epi 3 alpha=0.001 beta=0.002
## alpha                                      0.001
## beta                                       0.002
## Average Peak                              13.480
## Average Epi Length                        17.820
## Average Infections                       729.700
```

Exponential Epidemic Curves Beta (Beta = 0.0005) Fixed

```
#Betas Fixed
set.seed(3000)
epi_plot(samples, 100, 0.001, 0.0005, 20, "red", T)
```

```
## Warning in par(new = TRUE): calling par(new=TRUE) with no plot
```

```
epi_plot(samples, 100, 0.0015, 0.0005, 20, "green", T)
epi_plot(samples, 100, 0.002, 0.0005, 20, "blue", T)
title(main = "3x100 Exponential Epidemic Curves", xlab = "Time Point", ylab="Infections"); legen
d("topleft", legend=c("alpha=0.001", "alpha=0.0015", "alpha=0.002"), lwd=1, col=c("red", "gree
n", "blue"), cex=0.75)
```

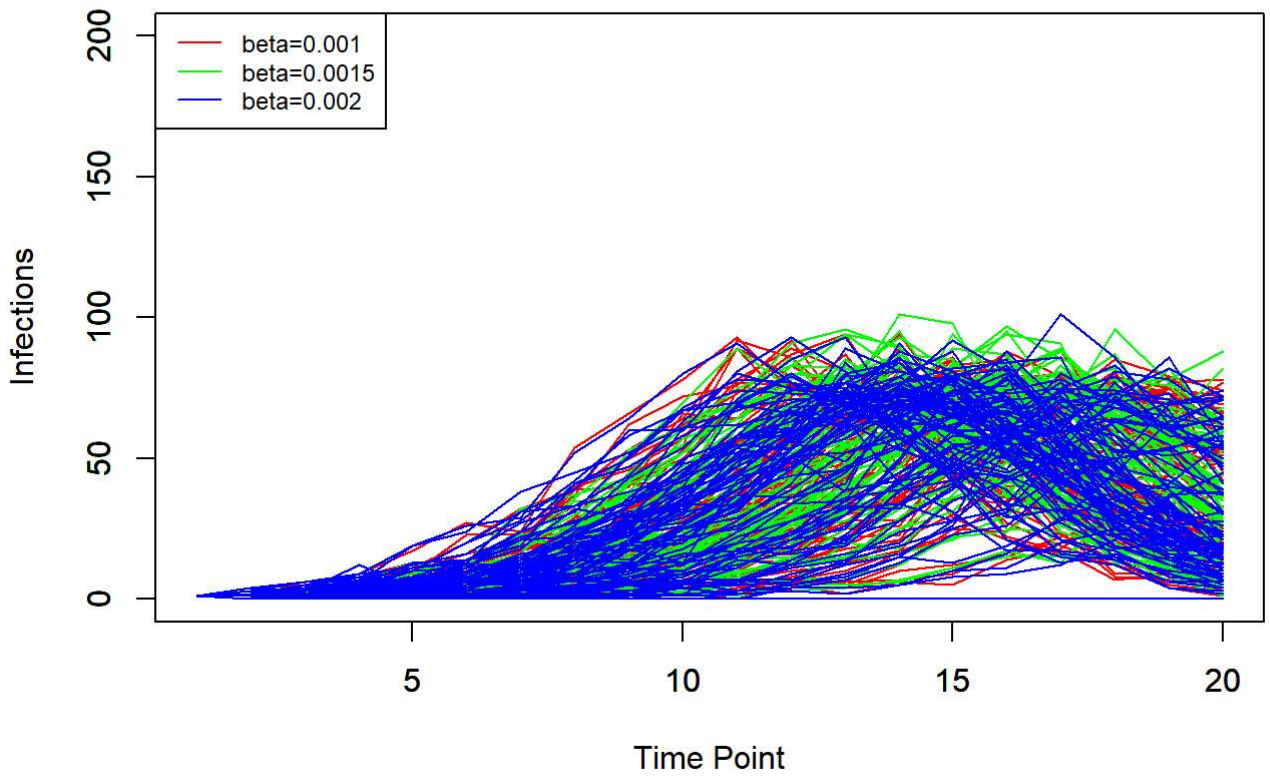


Exponential Epidemic Curves Alpha (Alpha = 0.001) Fixed

```
set.seed(3000)
epi_plot(samples, 100, 0.001, 0.001, 20, "red", T)
```

```
## Warning in par(new = TRUE): calling par(new=TRUE) with no plot
```

```
epi_plot(samples, 100, 0.001, 0.0015, 20, "green", T)
epi_plot(samples, 100, 0.001, 0.002, 20, "blue", T)
title(main = "3x100 Exponential Epidemic Curves", xlab = "Time Point", ylab="Infections"); legen
d("topleft", legend=c("beta=0.001", "beta=0.0015", "beta=0.002"), lwd=1, col=c("red", "green",
"blue"), cex=0.75)
```



Result Tables for Exponential Curves

```
#Same betas, different alphas
set.seed(3000)

exp_alpha_values = c(0.001, 0.0015, 0.002)
exp_beta_fixed = epi_sims(samples, 100, exp_alpha_values, 0.0005, 20, T)
exp_beta_fixed
```

```
##                    Epi 1 alpha=0.001 beta=5e-04 Epi 2 alpha=0.0015 beta=5e-04
## alpha                                    0.0010                         0.0015
## beta                                     0.0005                         0.0005
## Average Peak                            13.4300                        10.4900
## Average Epi Length                      17.6900                        19.2500
## Average Infections                     729.9400                       749.8900
##                    Epi 3 alpha=0.002 beta=5e-04
## alpha                                    0.0020
## beta                                     0.0005
## Average Peak                             9.0900
## Average Epi Length                      17.5200
## Average Infections                     750.0000
```

```
exp_beta_values = c(0.001, 0.0015, 0.002)
exp_alpha_fixed = epi_sims(samples, 100, 0.001, exp_beta_values, 20, T)
exp_alpha_fixed
```

```
##                    Epi 1 alpha=0.001 beta=0.001 Epi 2 alpha=0.001 beta=0.0015
## alpha                                      0.001                        0.0010
## beta                                       0.001                        0.0015
## Average Peak                              13.360                       13.5700
## Average Epi Length                        18.030                       17.9200
## Average Infections                       732.870                      728.5100
##                    Epi 3 alpha=0.001 beta=0.002
## alpha                                      0.001
## beta                                       0.002
## Average Peak                              13.160
## Average Epi Length                        17.510
## Average Infections                       729.840
```

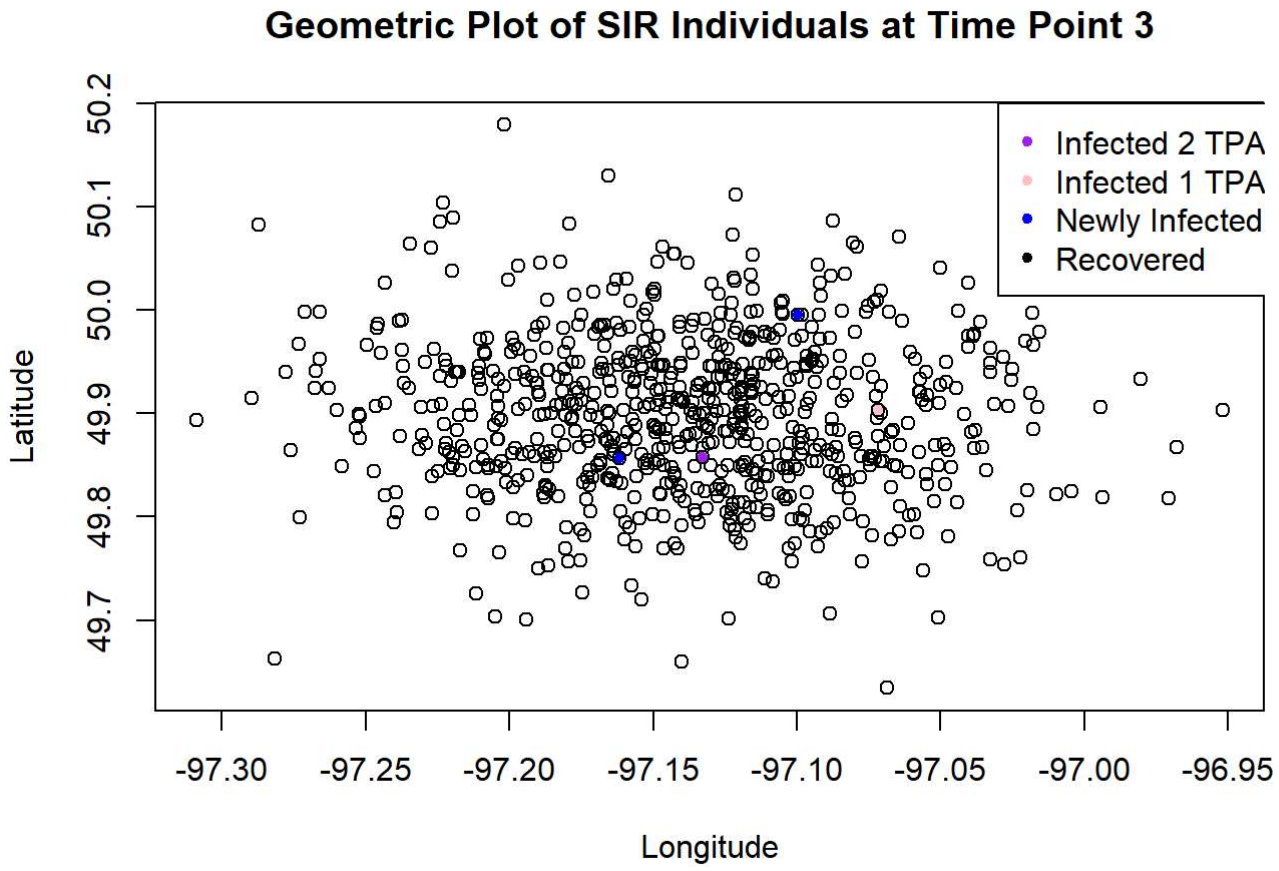
Infected Title and Heat Map Generator Function

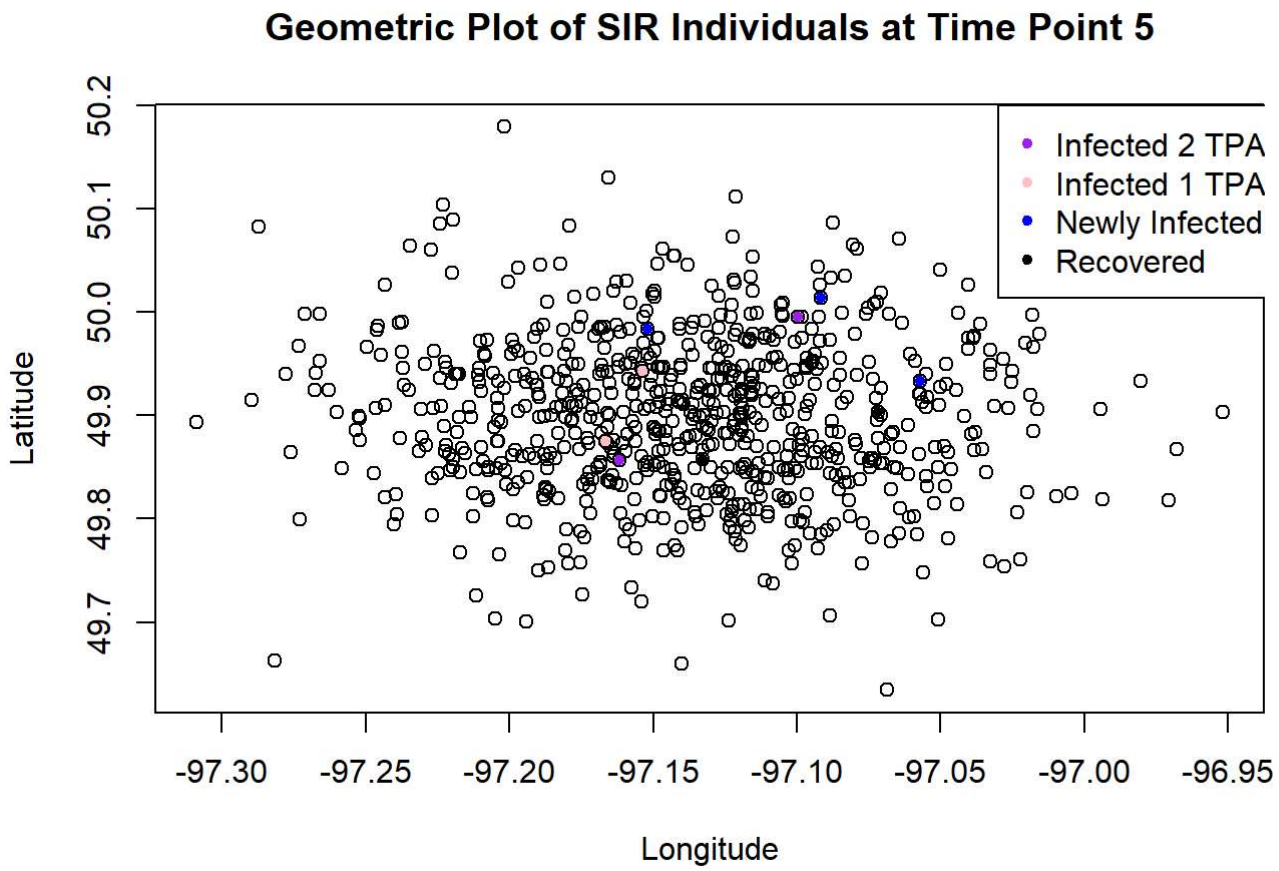
```
title_generator = function(is_exp, is_heat_map, start_time, end_time){

  title_start = ifelse(is_exp, "Exponential", "Geometric")
  middle_phrase = ifelse(is_heat_map, "Heat Map", "Plot of SIR Individuals")


  titles = c(rep("", end_time - start_time + 1))
  for(time_point in start_time:end_time){

    #Runs true if time_point is odd, or if time_point is the last time
    if ((time_point%%2 == 1) | time_point == end_time){
      titles[time_point - start_time + 1] = paste0(title_start, " ", middle_phrase,  " at Time P
oint ", time_point)
    }

  }
  return(titles)
}
```

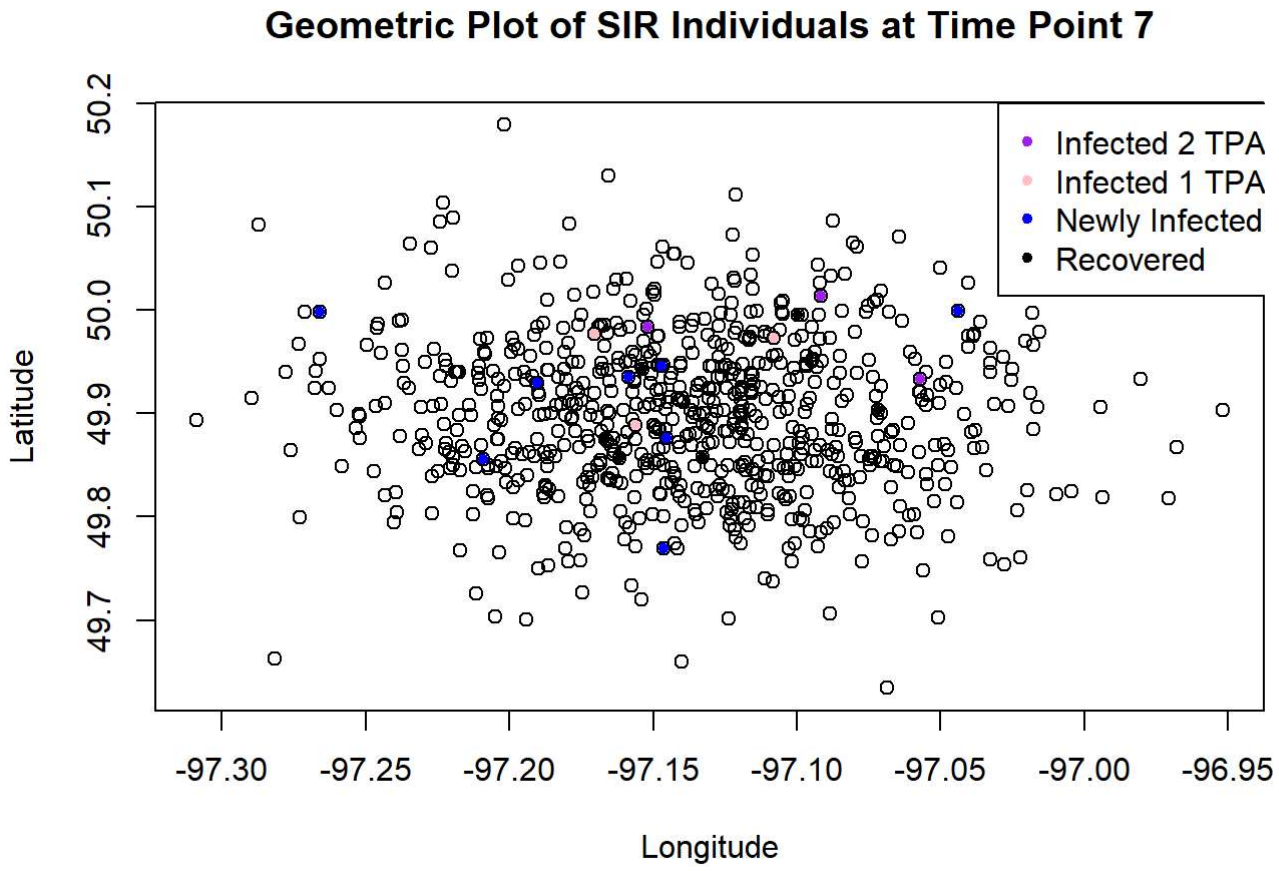
Note that the lines below run in RMarkdown but won't knit to HTML due to an unknown issue.

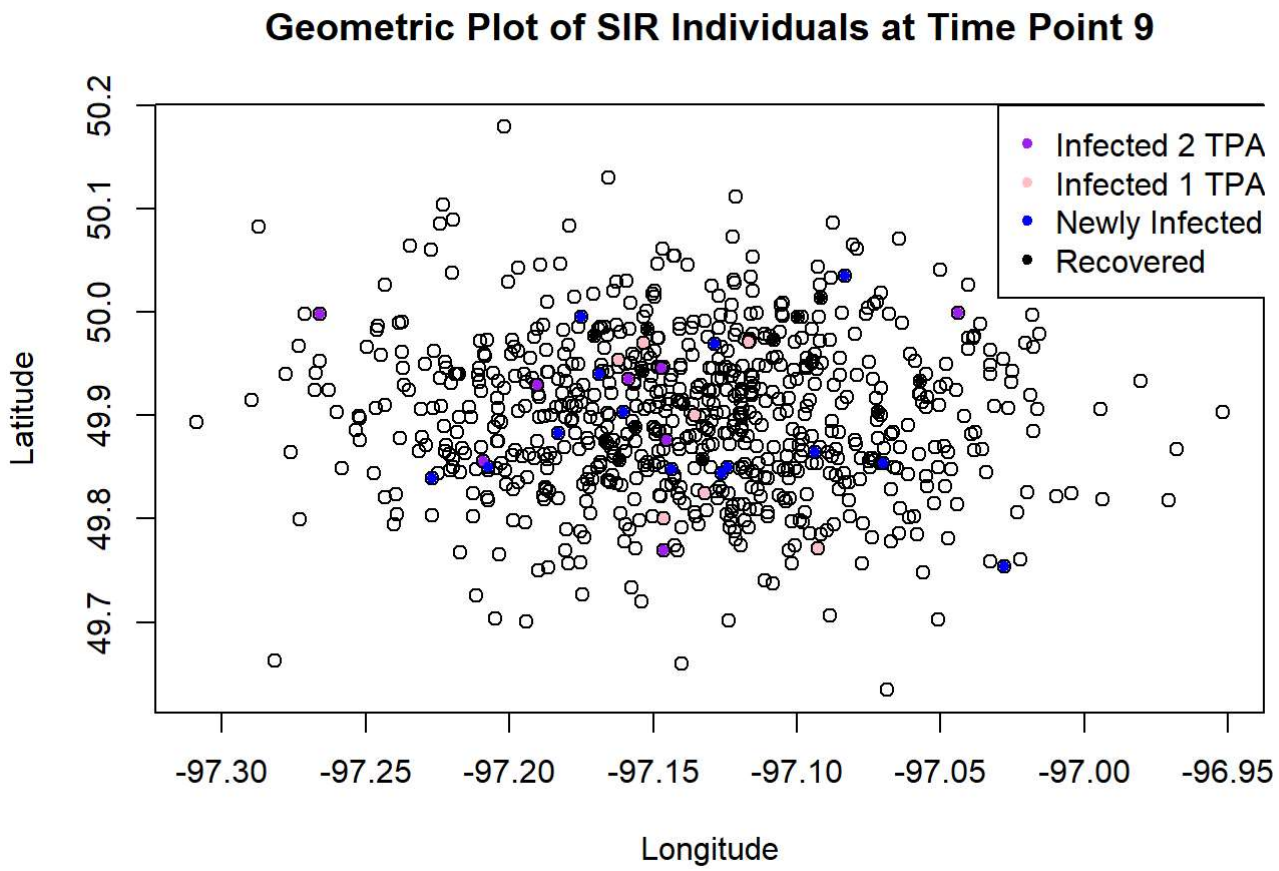
Geometric Heat Maps and Maps of Infected Individuals

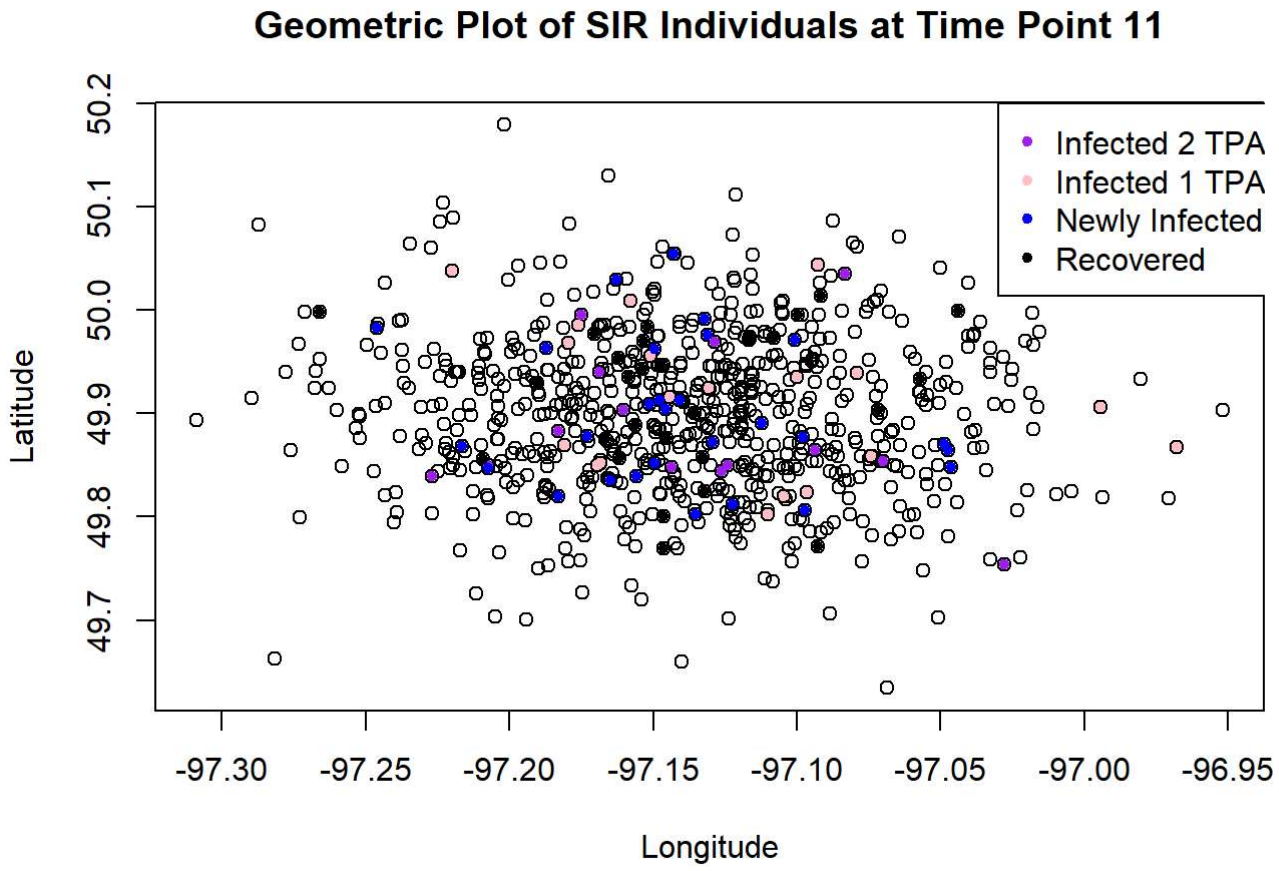
```
#Setting the titles that we will use for the plots
infectedTitleGeo = title_generator(F, F, 3, 20)
heatMapTitleGeo = title_generator(F, T, 3, 20)


set.seed(3001)
epigenGeo=epigenGeometric(samples, 0.001, 0.0005, 20)

#TPA stands for time point ago
activeInfectionsDisplayer(2, 1, 1, 20, epigenGeo$inftime, samples, infectedTitleGeo)
```

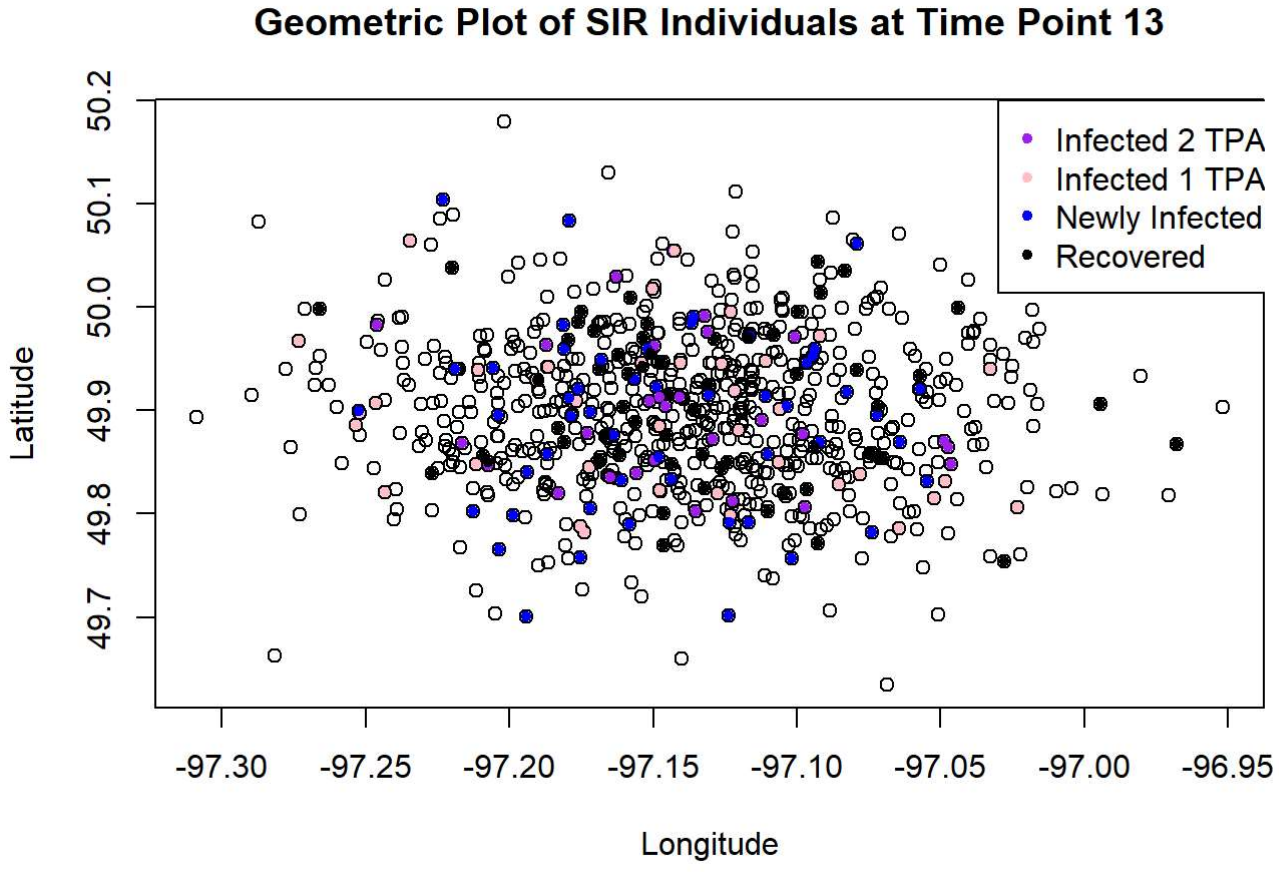

Geometric Plot of SIR Individuals at Time Point 3
Infected 2 TPA
Infected 1 TPA
Newly Infected
Recovered
Latitude
Longitude
50.2
50.1
50.0
49.9
49.8
49.7
-97.30
-97.25
-97.20
-97.15
-97.10
-97.05
-97.00
-96.95

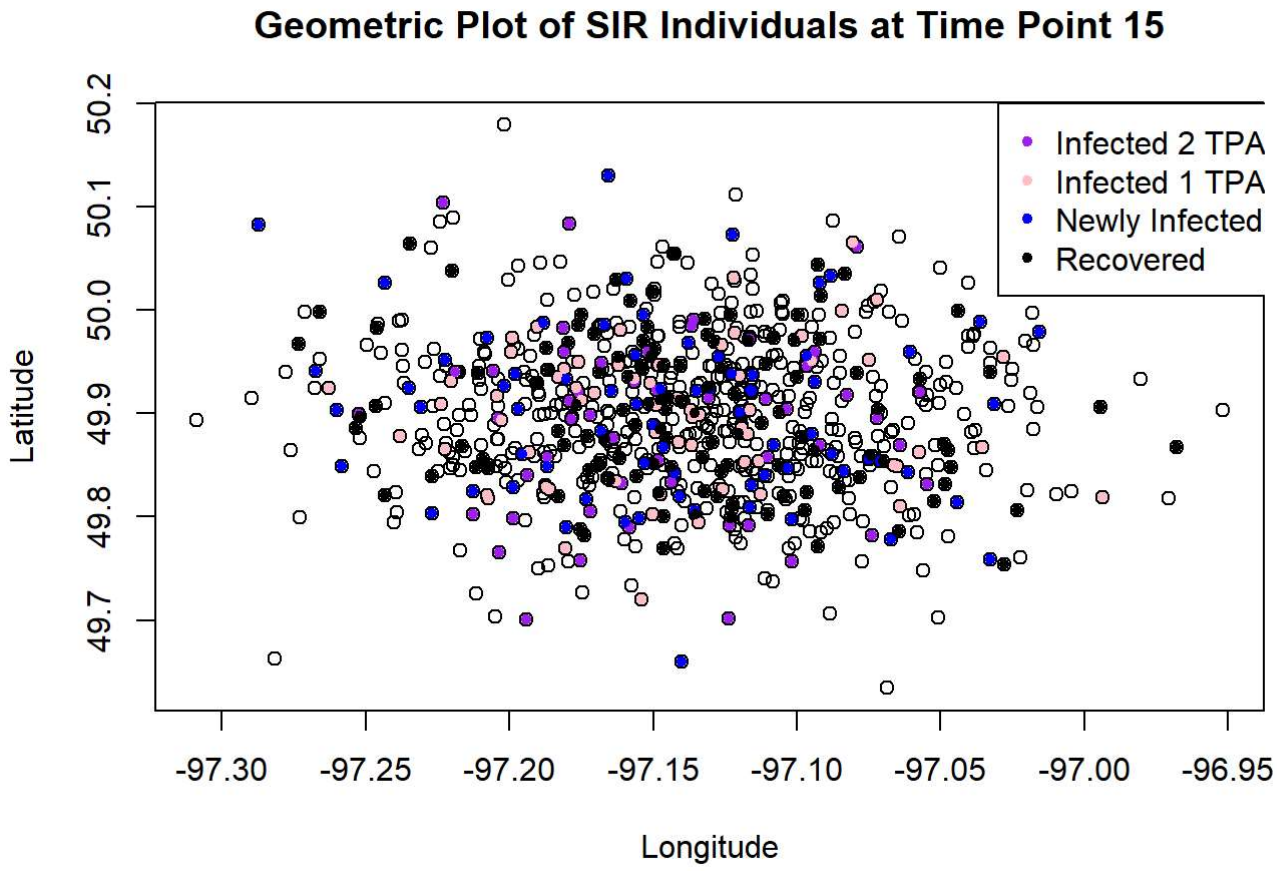

Geometric Plot of SIR Individuals at Time Point 5
Infected 2 TPA
Infected 1 TPA
Newly Infected
Recovered
Latitude
Longitude
50.2
50.1
50.0
49.9
49.8
49.7
-97.30
-97.25
-97.20
-97.15
-97.10
-97.05
-97.00
-96.95

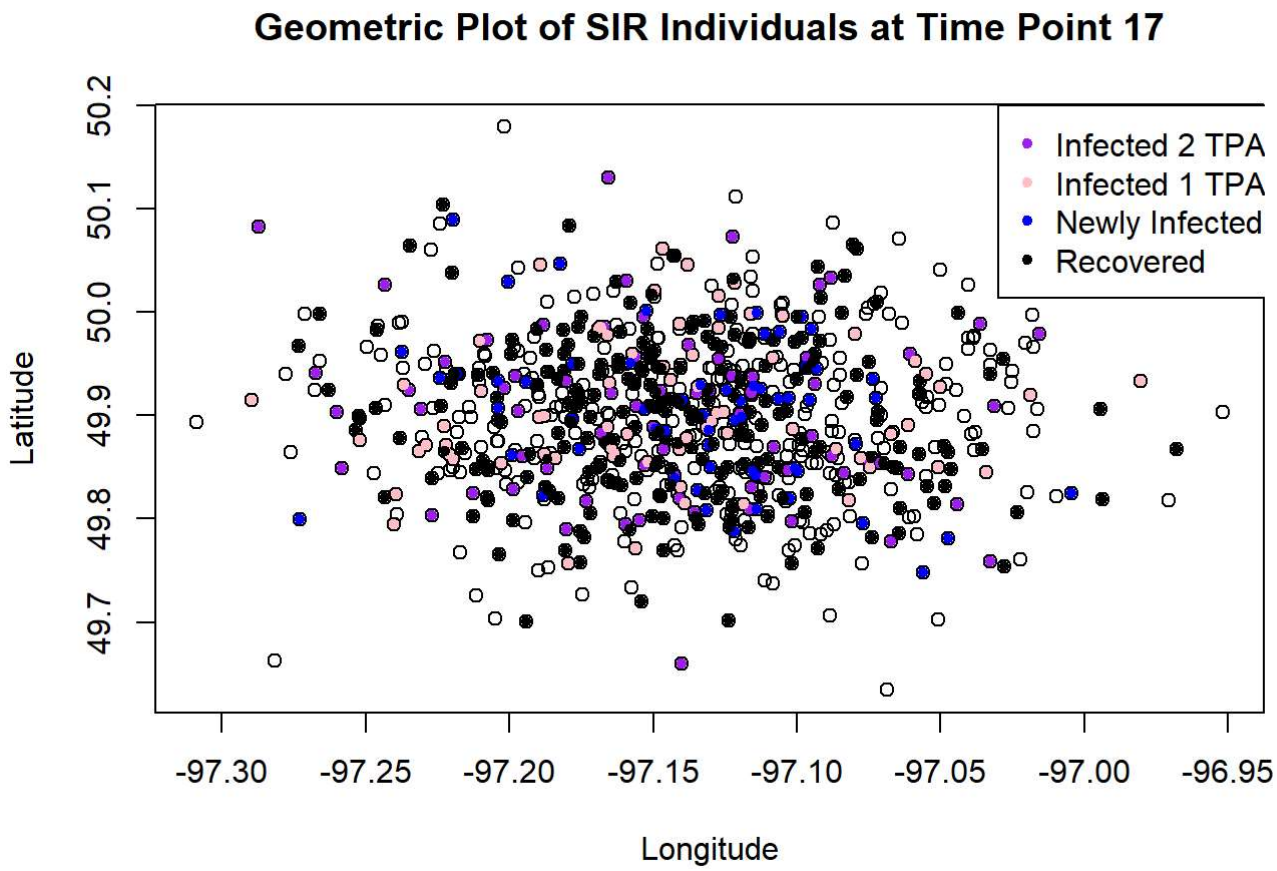

Geometric Plot of SIR Individuals at Time Point 7
Infected 2 TPA
Infected 1 TPA
Newly Infected
Recovered
Latitude
Longitude
49.7
49.8
49.9
50.0
50.1
50.2
-97.30
-97.25
-97.20
-97.15
-97.10
-97.05
-97.00
-96.95

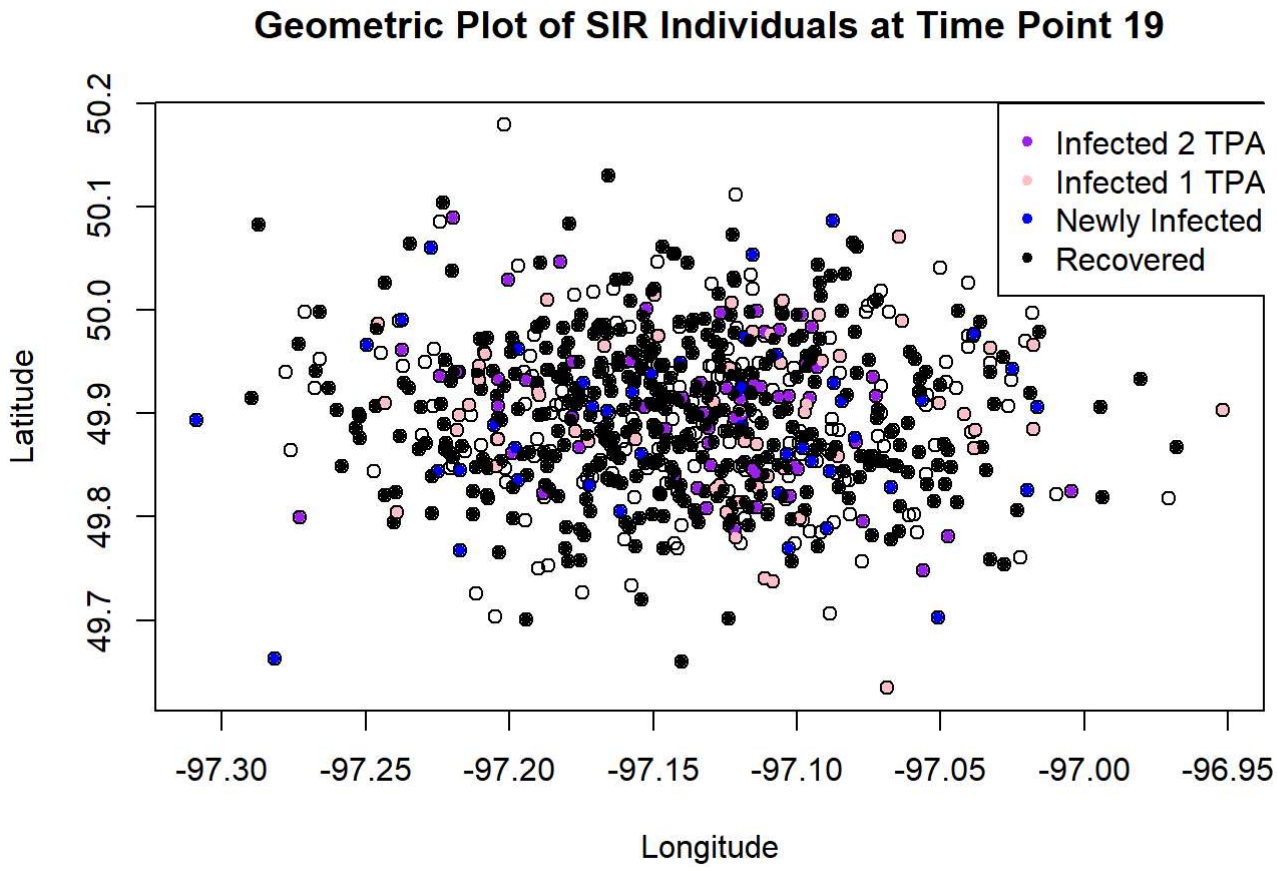

Geometric Plot of SIR Individuals at Time Point 9
Infected 2 TPA
Infected 1 TPA
Newly Infected
Recovered
Latitude
Longitude
49.7
49.8
49.9
50.0
50.1
50.2
-97.30
-97.25
-97.20
-97.15
-97.10
-97.05
-97.00
-96.95

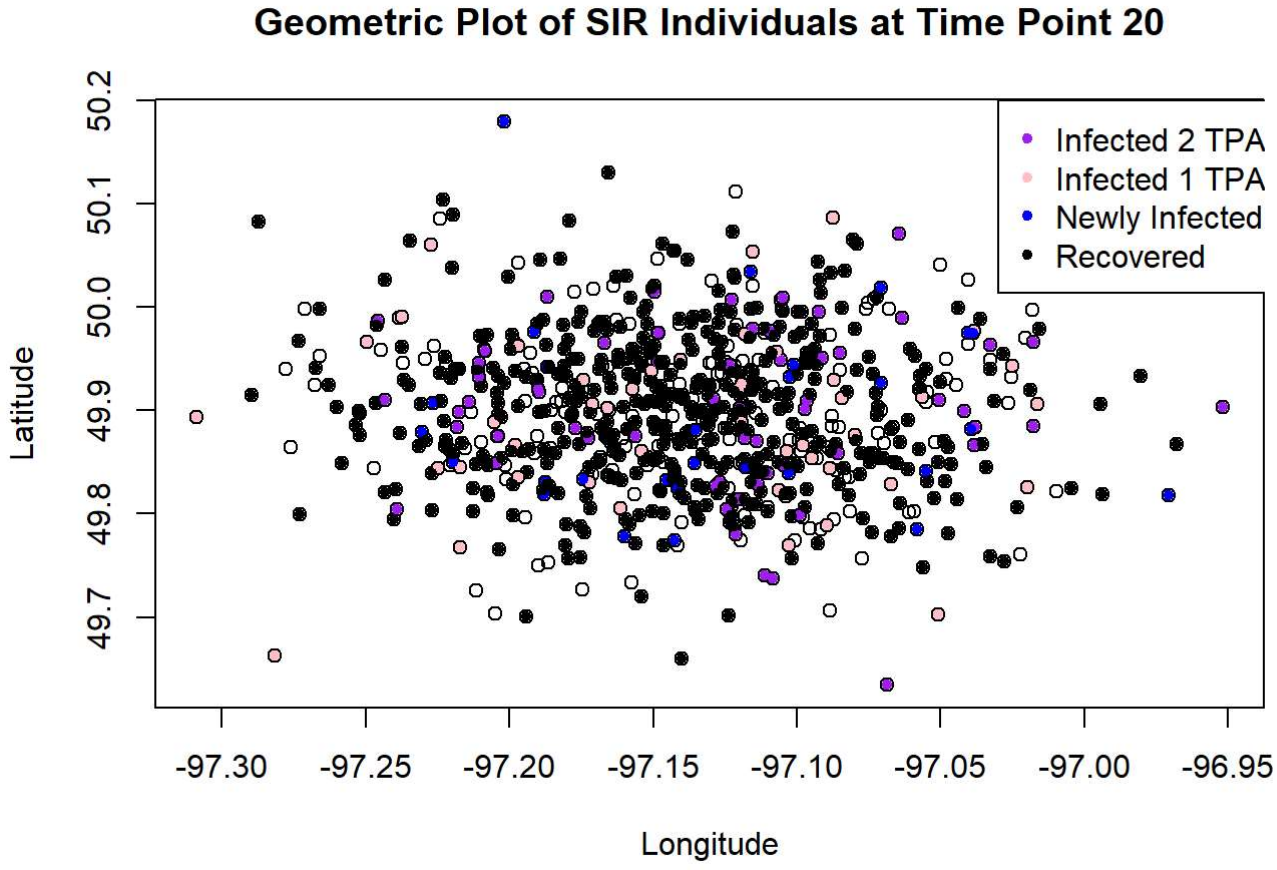

Geometric Plot of SIR Individuals at Time Point 11
Infected 2 TPA
Infected 1 TPA
Newly Infected
Recovered
Latitude
Longitude
50.2
50.1
50.0
49.9
49.8
49.7
-97.30
-97.25
-97.20
-97.15
-97.10
-97.05
-97.00
-96.95


Geometric Plot of SIR Individuals at Time Point 13
Infected 2 TPA
Infected 1 TPA
Newly Infected
Recovered
Latitude
Longitude
50.2
50.1
50.0
49.9
49.8
49.7
-97.30
-97.25
-97.20
-97.15
-97.10
-97.05
-97.00
-96.95

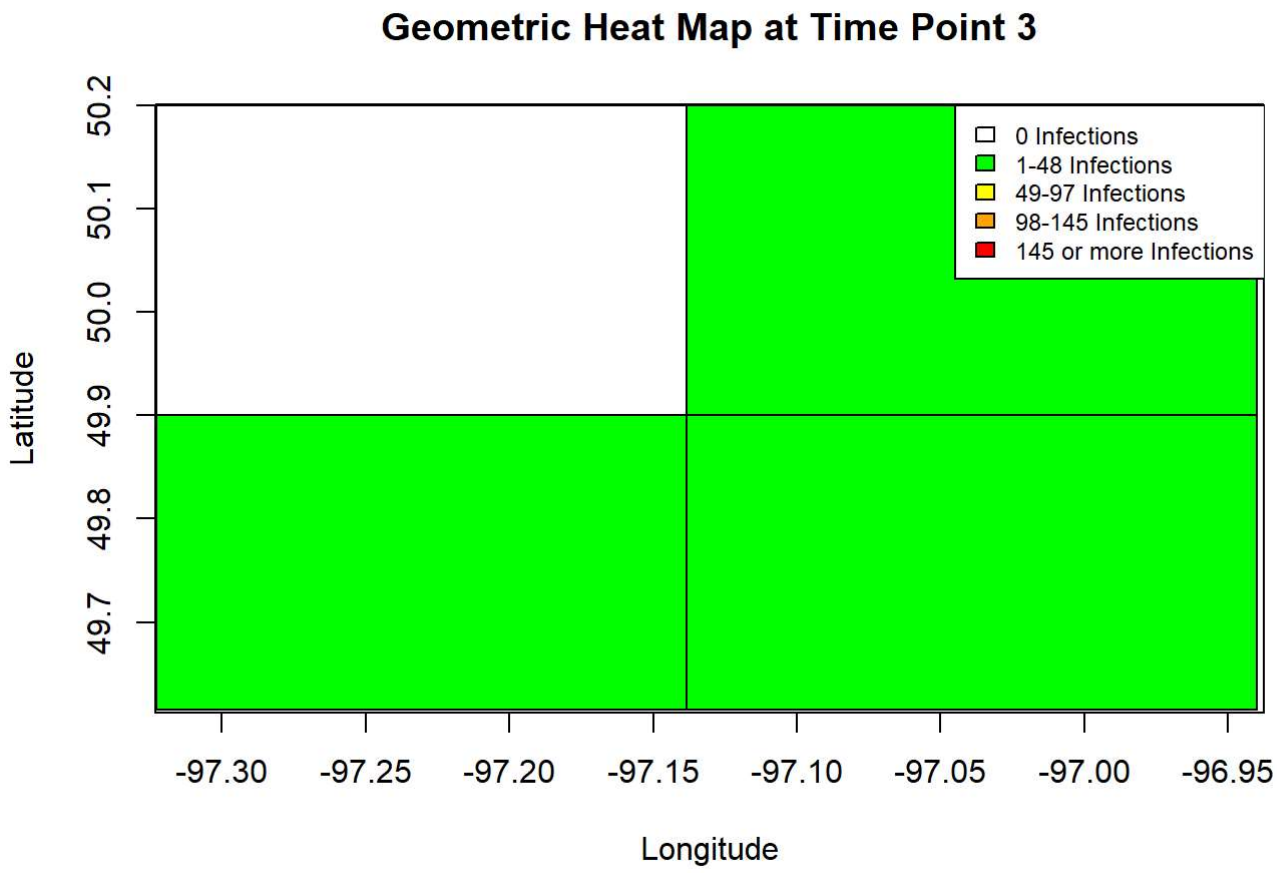

Geometric Plot of SIR Individuals at Time Point 15
Infected 2 TPA
Infected 1 TPA
Newly Infected
Recovered
Latitude
Longitude
49.7
49.8
49.9
50.0
50.1
50.2
-97.30
-97.25
-97.20
-97.15
-97.10
-97.05
-97.00
-96.95

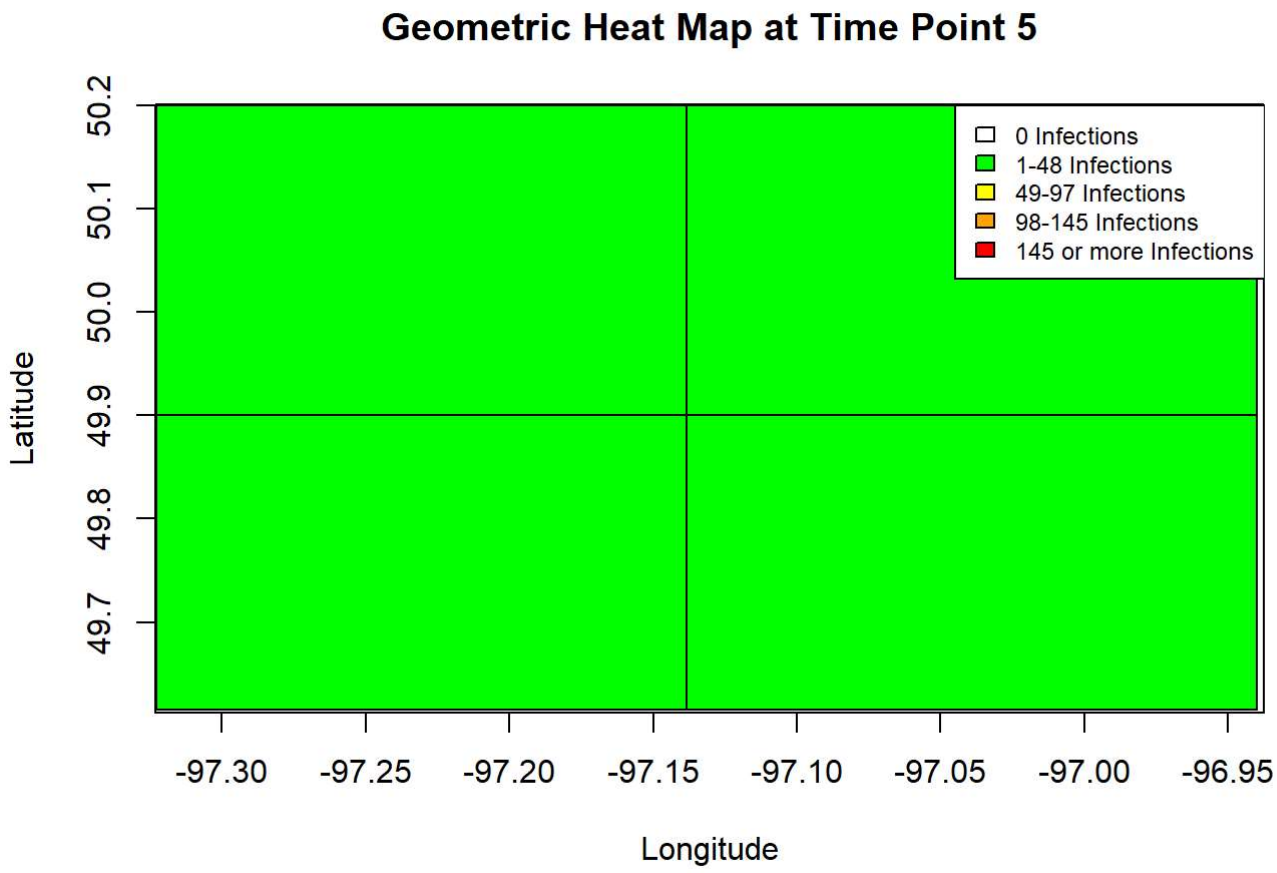

Geometric Plot of SIR Individuals at Time Point 17
Infected 2 TPA
Infected 1 TPA
Newly Infected
Recovered
Latitude
Longitude
49.7
49.8
49.9
50.0
50.1
50.2
-97.30
-97.25
-97.20
-97.15
-97.10
-97.05
-97.00
-96.95

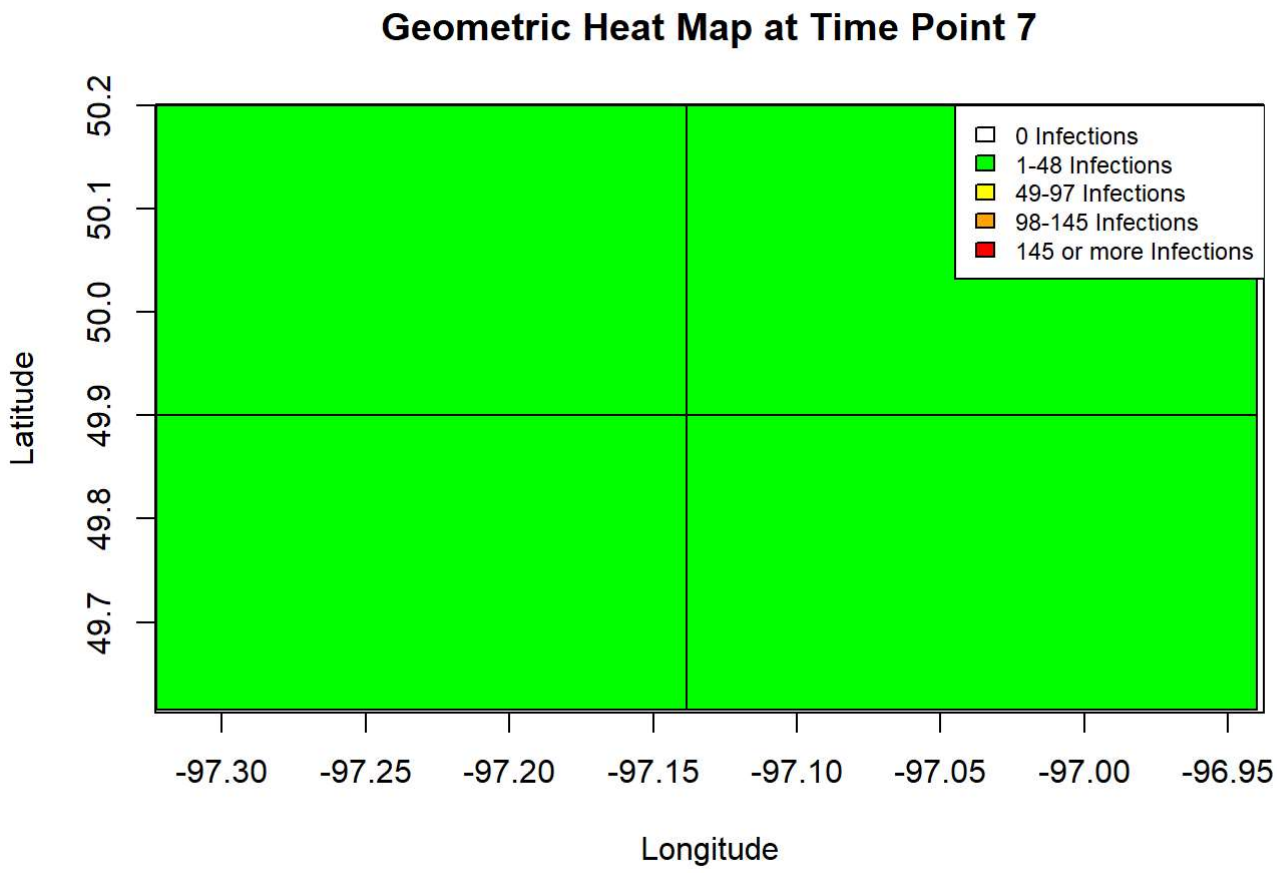

Geometric Plot of SIR Individuals at Time Point 19
Infected 2 TPA
Infected 1 TPA
Newly Infected
Recovered
Latitude
Longitude
50.2
50.1
50.0
49.9
49.8
49.7
-97.30
-97.25
-97.20
-97.15
-97.10
-97.05
-97.00
-96.95

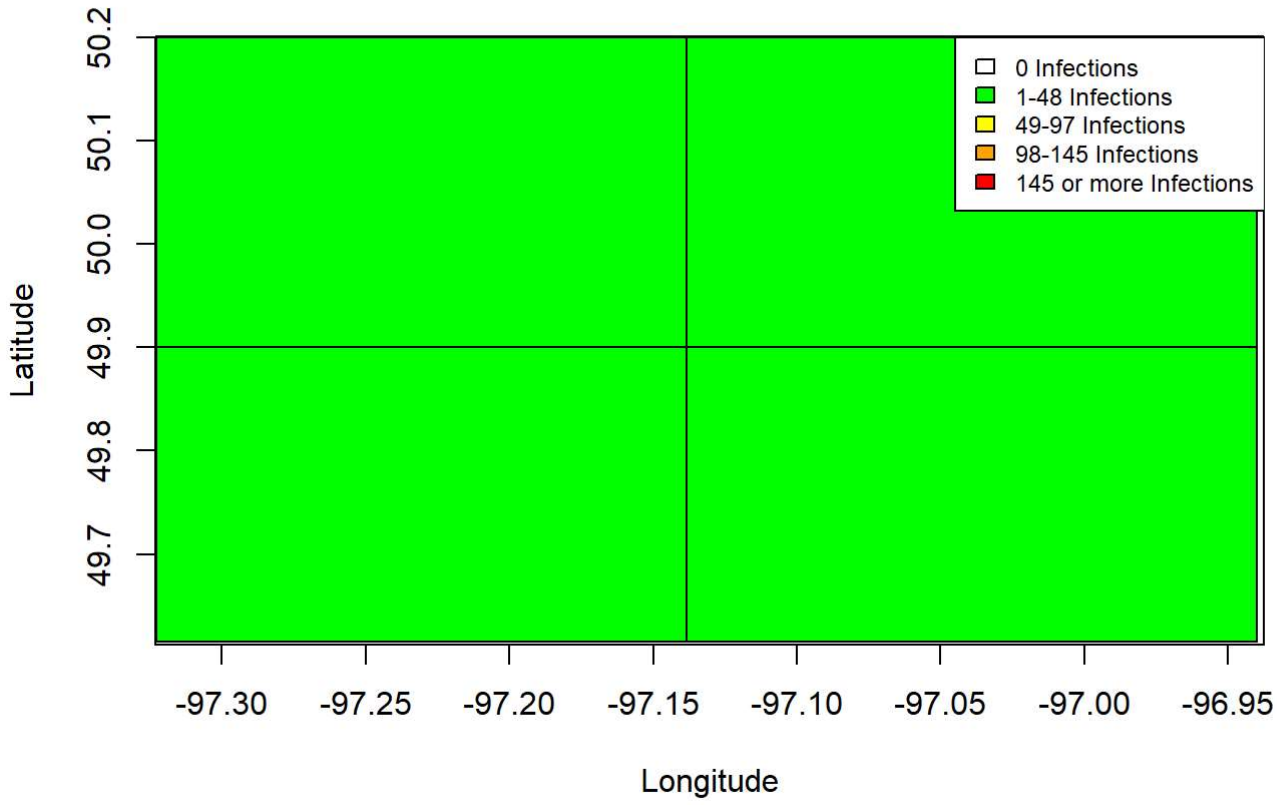

Geometric Plot of SIR Individuals at Time Point 20
Infected 2 TPA
Infected 1 TPA
Newly Infected
Recovered
Latitude
Longitude
50.2
50.1
50.0
49.9
49.8
49.7
-97.30
-97.25
-97.20
-97.15
-97.10
-97.05
-97.00
-96.95

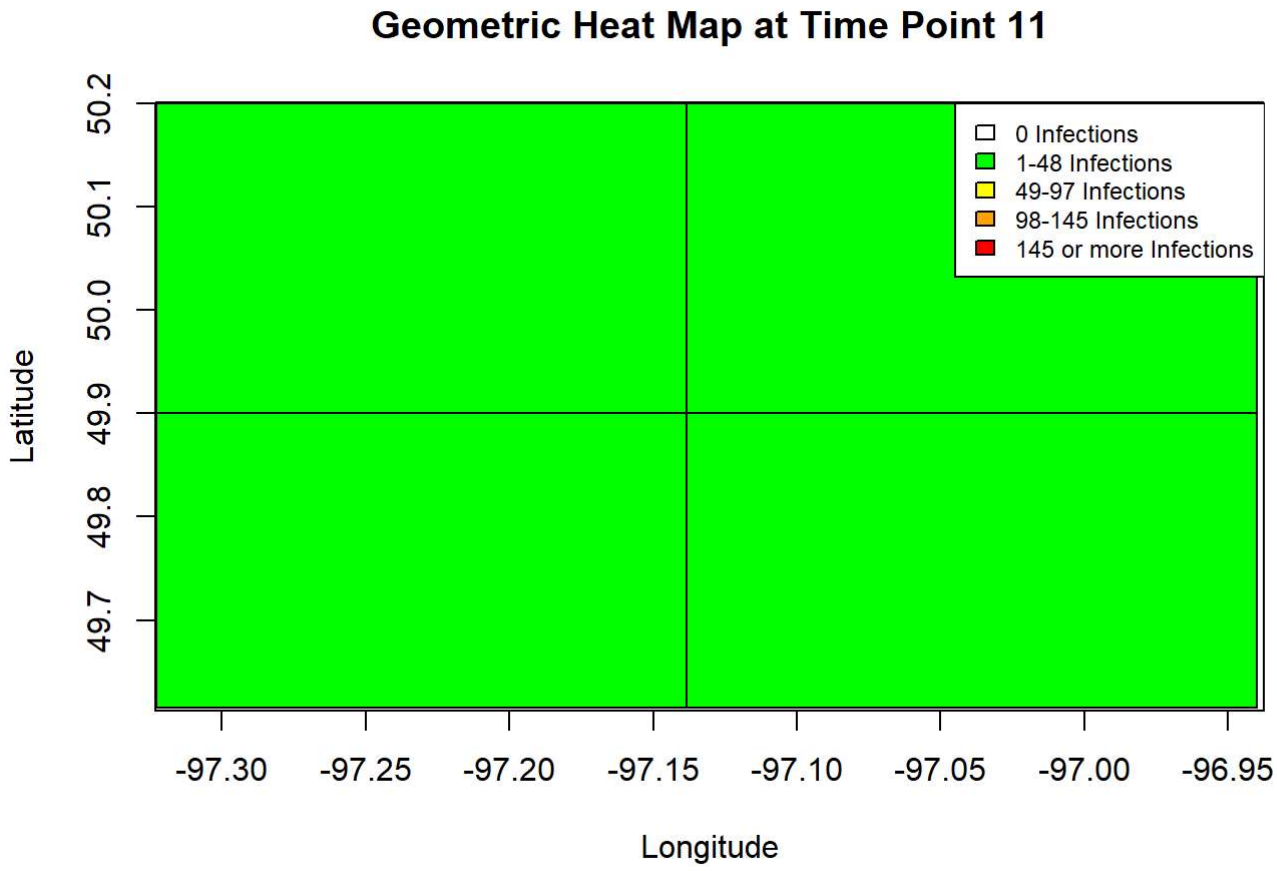
```
heatMapDisplayer(2, 1, 1, 20, epigenGeo$inftime, samples, heatMapTitleGeo)
```

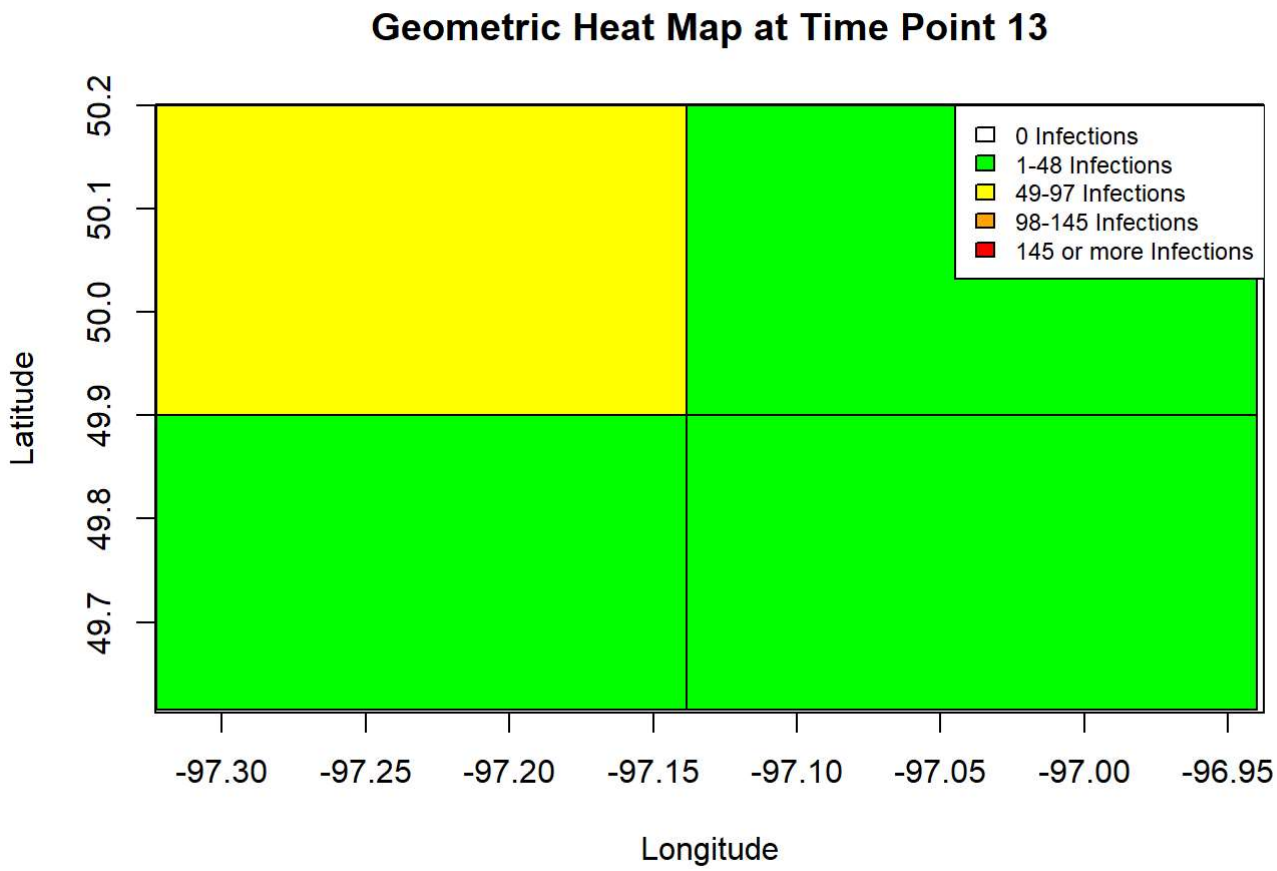

Geometric Heat Map at Time Point 3
0 Infections
1-48 Infections
49-97 Infections
98-145 Infections
145 or more Infections
Latitude
50.2
50.1
50.0
49.9
49.8
49.7
-97.30
-97.25
-97.20
-97.15
-97.10
-97.05
-97.00
-96.95
Longitude

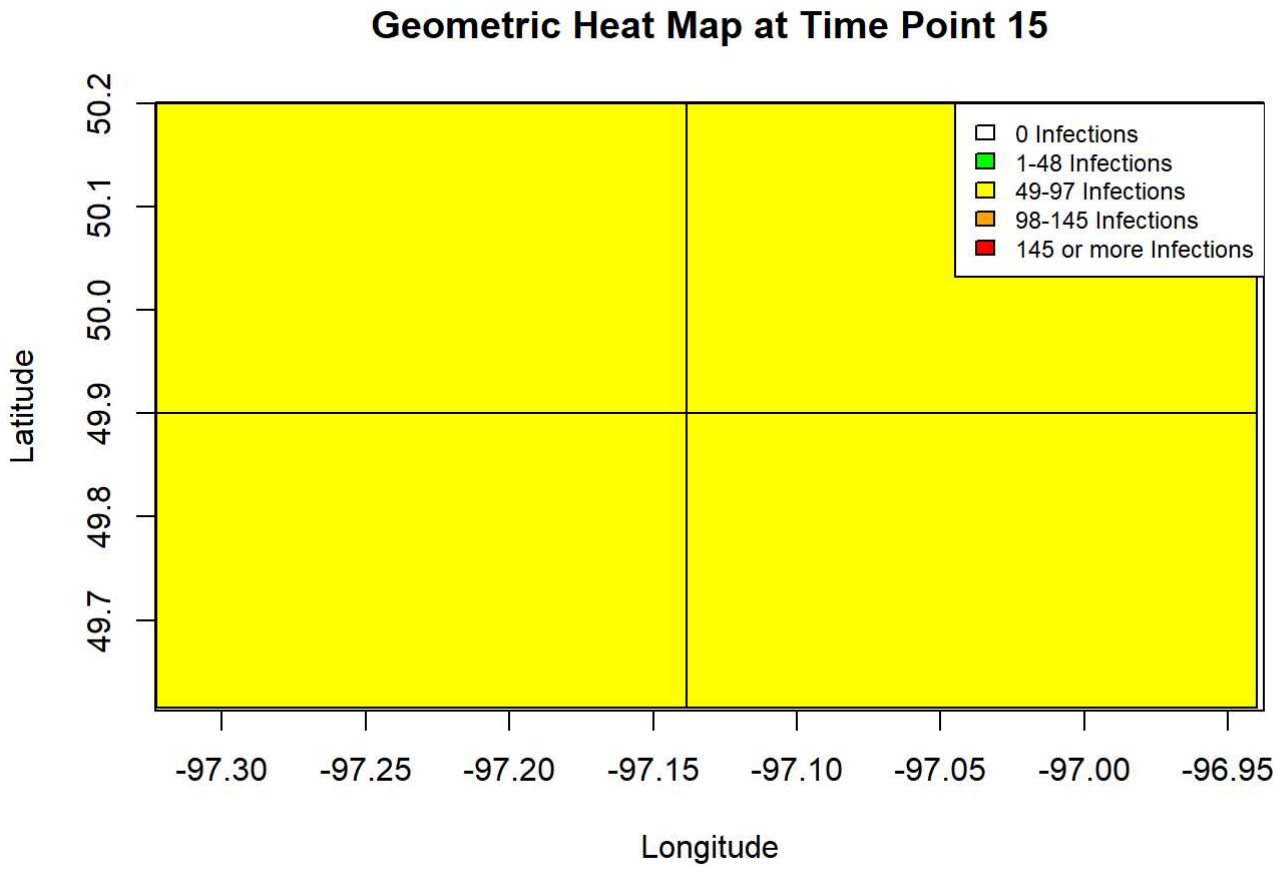

Geometric Heat Map at Time Point 5
0 Infections
1-48 Infections
49-97 Infections
98-145 Infections
145 or more Infections
Latitude
50.2
50.1
50.0
49.9
49.8
49.7
-97.30
-97.25
-97.20
-97.15
-97.10
-97.05
-97.00
-96.95
Longitude

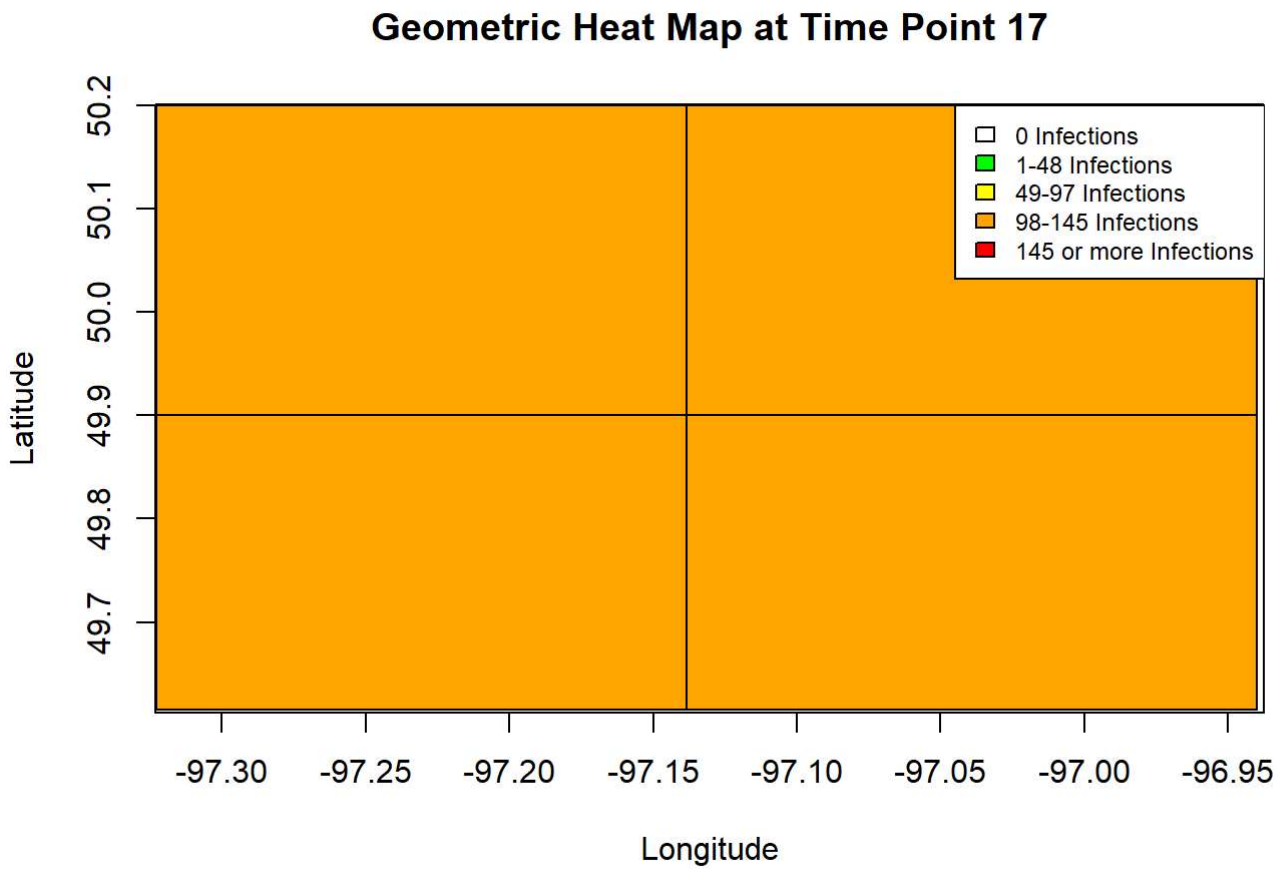

Geometric Heat Map at Time Point 7
0 Infections
1-48 Infections
49-97 Infections
98-145 Infections
145 or more Infections
Latitude
50.2
50.1
50.0
49.9
49.8
49.7
-97.30
-97.25
-97.20
-97.15
-97.10
-97.05
-97.00
-96.95
Longitude

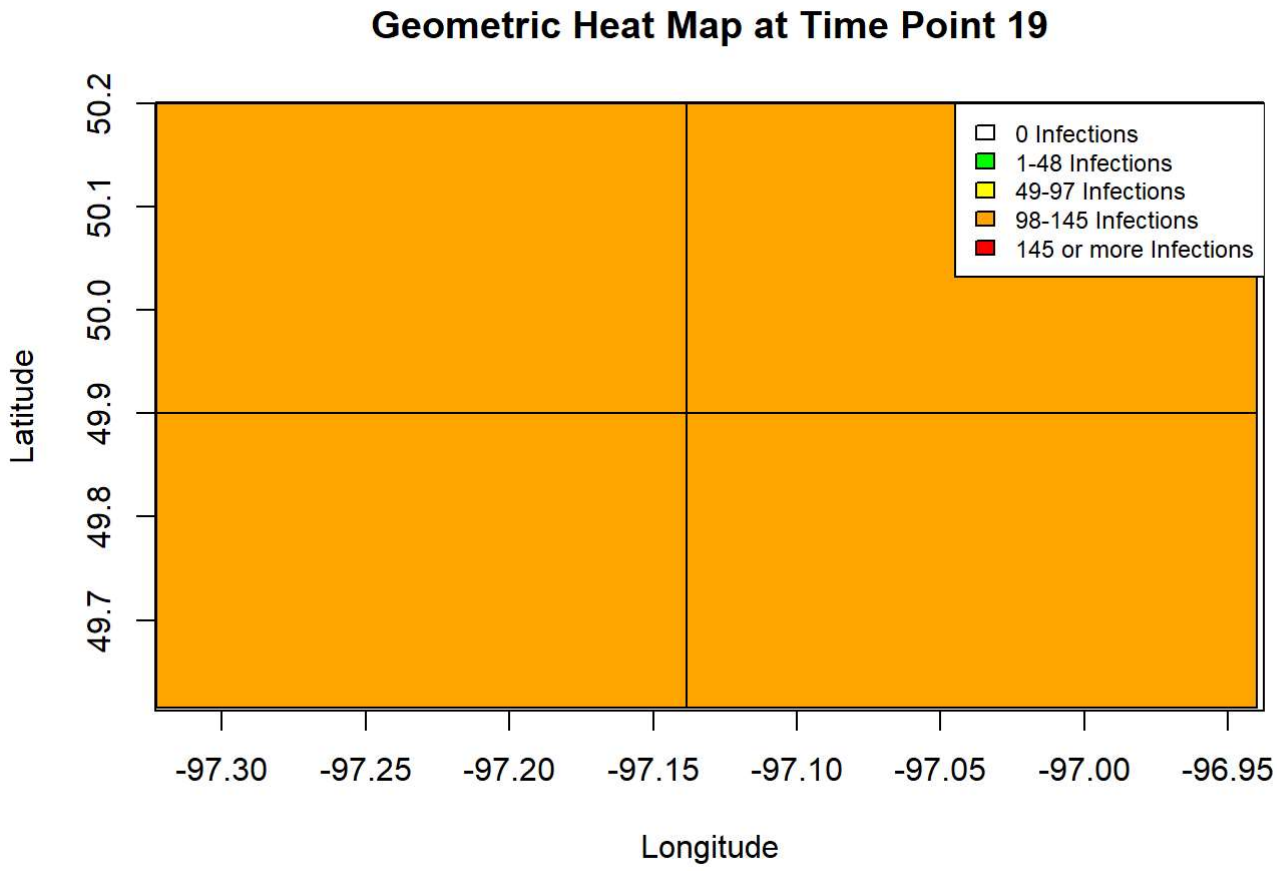

Geometric Heat Map at Time Point 9
0 Infections
1-48 Infections
49-97 Infections
98-145 Infections
145 or more Infections
Latitude
50.2
50.1
50.0
49.9
49.8
49.7
-97.30
-97.25
-97.20
-97.15
-97.10
-97.05
-97.00
-96.95
Longitude

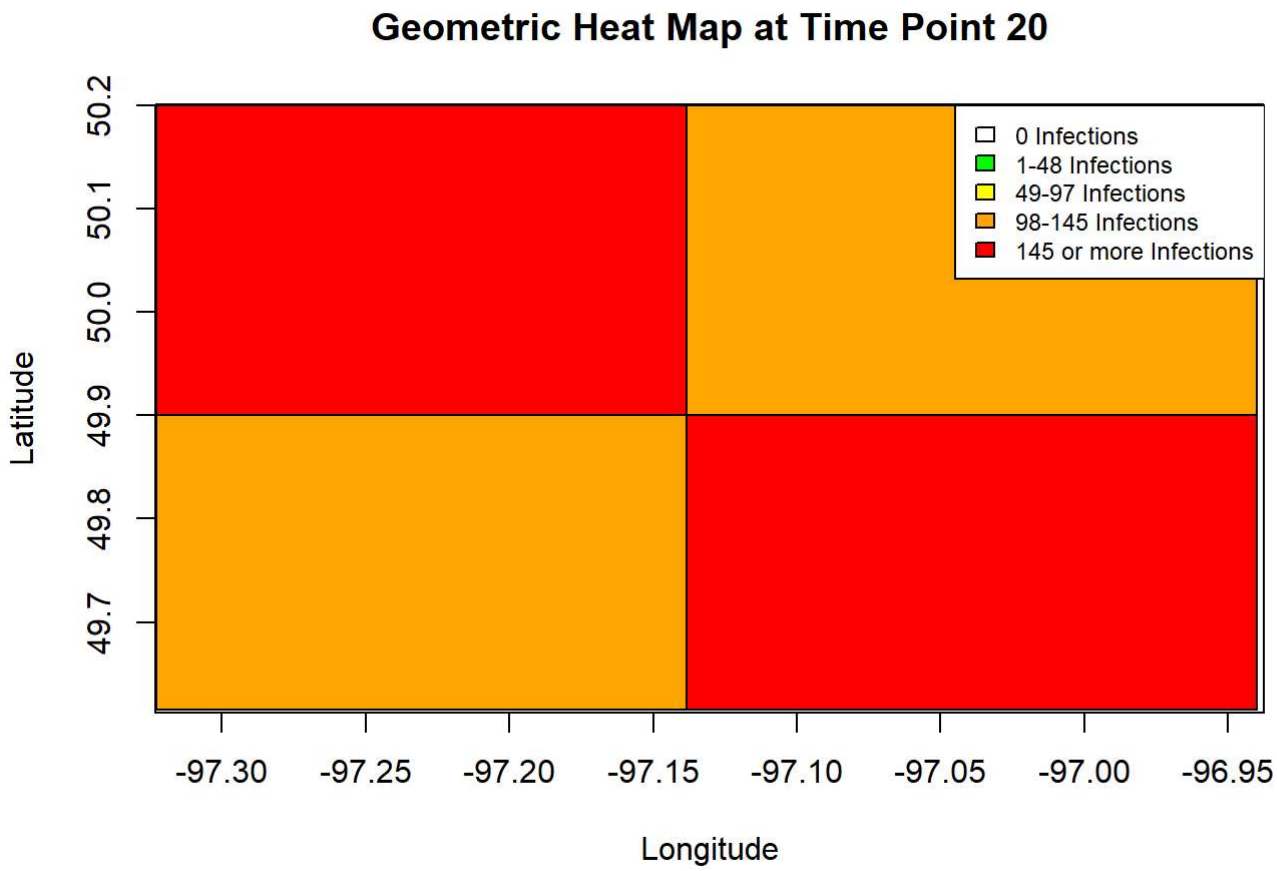

Geometric Heat Map at Time Point 11
0 Infections
1-48 Infections
49-97 Infections
98-145 Infections
145 or more Infections
Latitude
50.2
50.1
50.0
49.9
49.8
49.7
-97.30
-97.25
-97.20
-97.15
-97.10
-97.05
-97.00
-96.95
Longitude


Geometric Heat Map at Time Point 13
0 Infections
1-48 Infections
49-97 Infections
98-145 Infections
145 or more Infections
Latitude
50.2
50.1
50.0
49.9
49.8
49.7
-97.30
-97.25
-97.20
-97.15
-97.10
-97.05
-97.00
-96.95
Longitude


Geometric Heat Map at Time Point 15
0 Infections
1-48 Infections
49-97 Infections
98-145 Infections
145 or more Infections
Latitude
50.2
50.1
50.0
49.9
49.8
49.7
-97.30
-97.25
-97.20
-97.15
-97.10
-97.05
-97.00
-96.95
Longitude

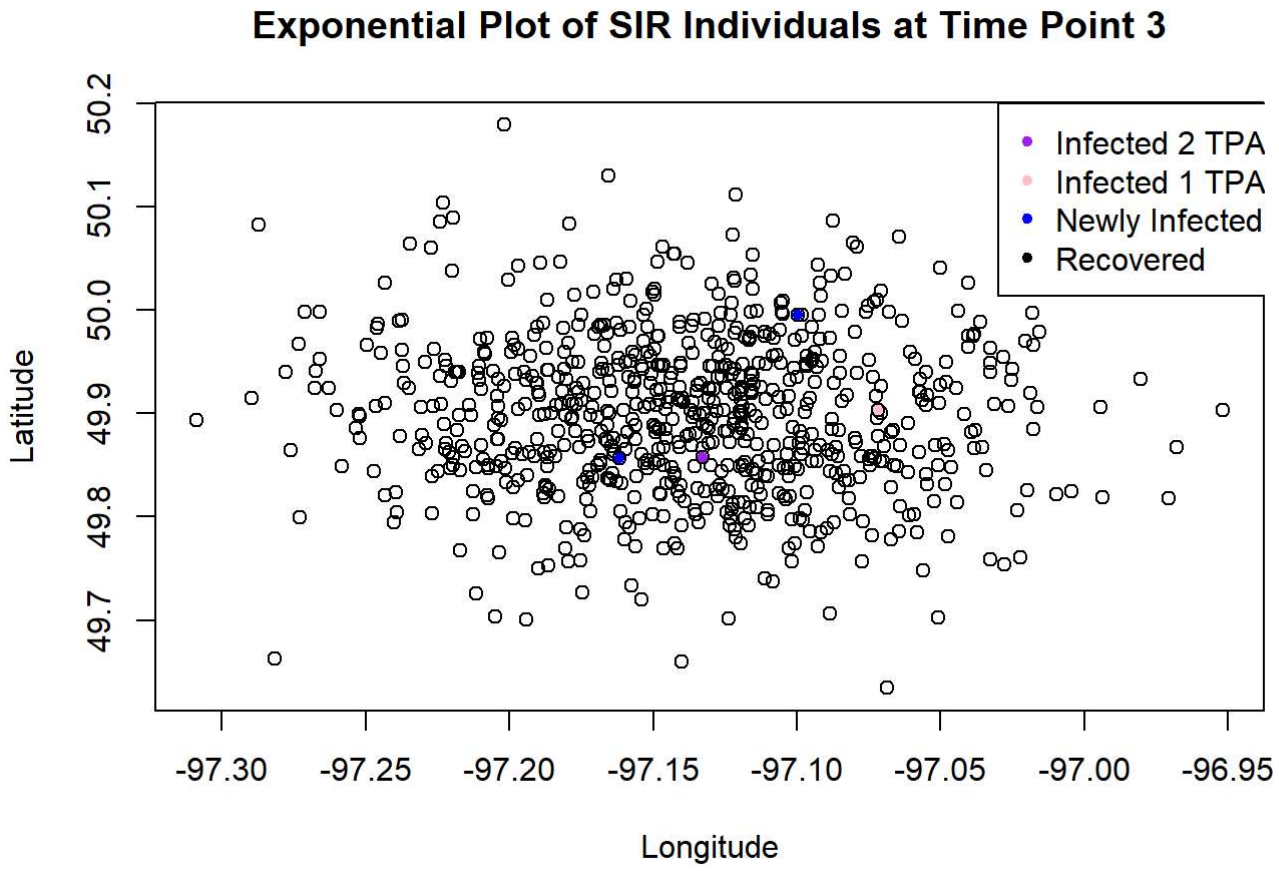

Geometric Heat Map at Time Point 17
0 Infections
1-48 Infections
49-97 Infections
98-145 Infections
145 or more Infections
Latitude
50.2
50.1
50.0
49.9
49.8
49.7
-97.30
-97.25
-97.20
-97.15
-97.10
-97.05
-97.00
-96.95
Longitude

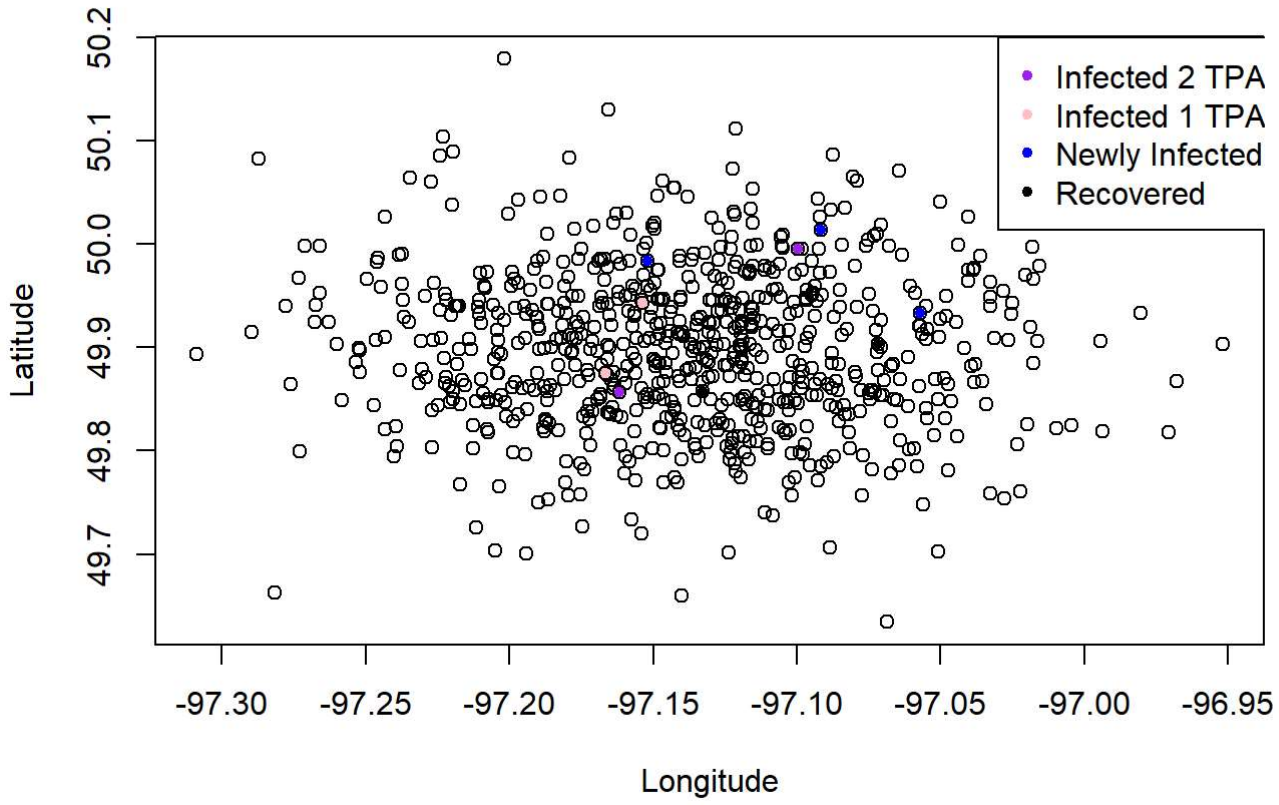

Geometric Heat Map at Time Point 19
0 Infections
1-48 Infections
49-97 Infections
98-145 Infections
145 or more Infections
Latitude
50.2
50.1
50.0
49.9
49.8
49.7
-97.30
-97.25
-97.20
-97.15
-97.10
-97.05
-97.00
-96.95
Longitude

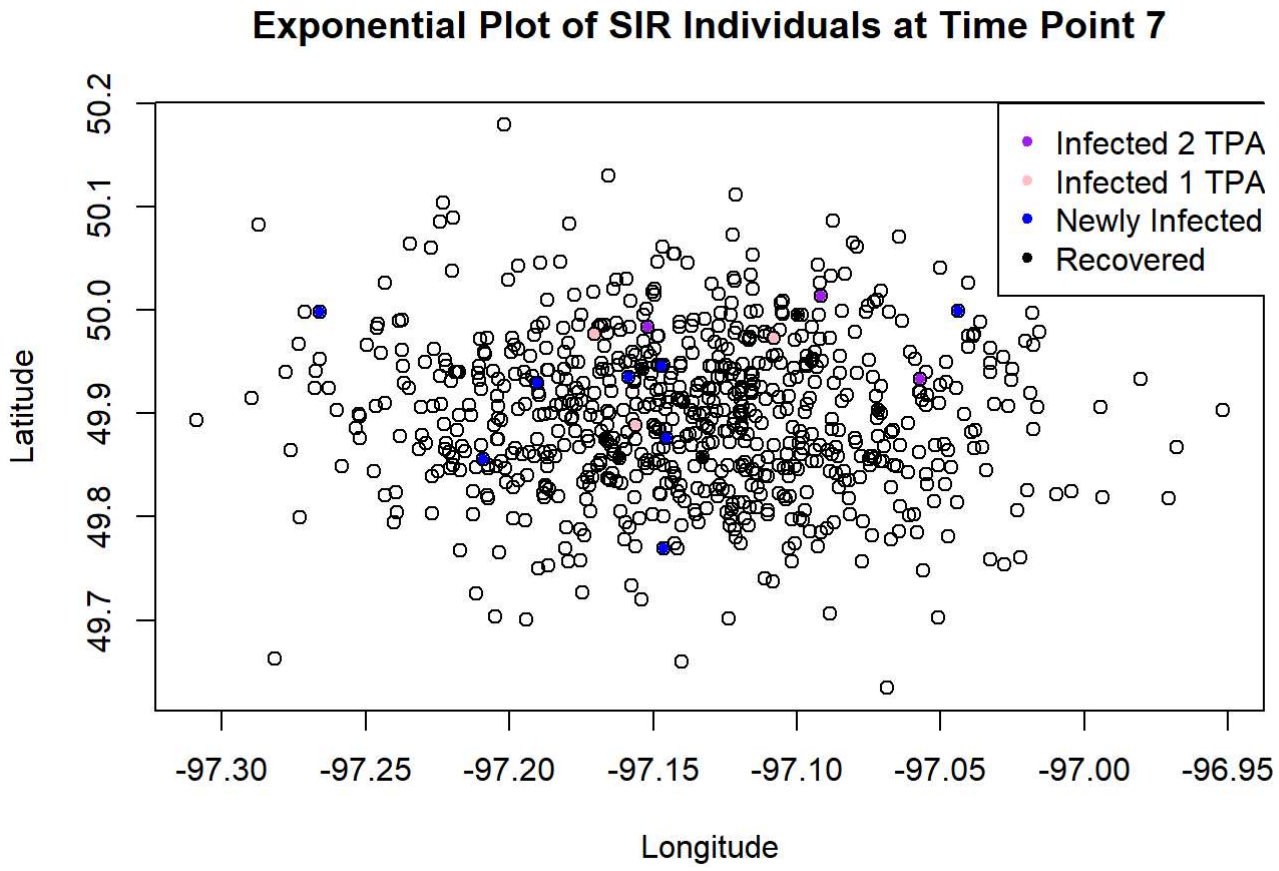

Geometric Heat Map at Time Point 20
0 Infections
1-48 Infections
49-97 Infections
98-145 Infections
145 or more Infections
Latitude
50.2
50.1
50.0
49.9
49.8
49.7
-97.30
-97.25
-97.20
-97.15
-97.10
-97.05
-97.00
-96.95
Longitude

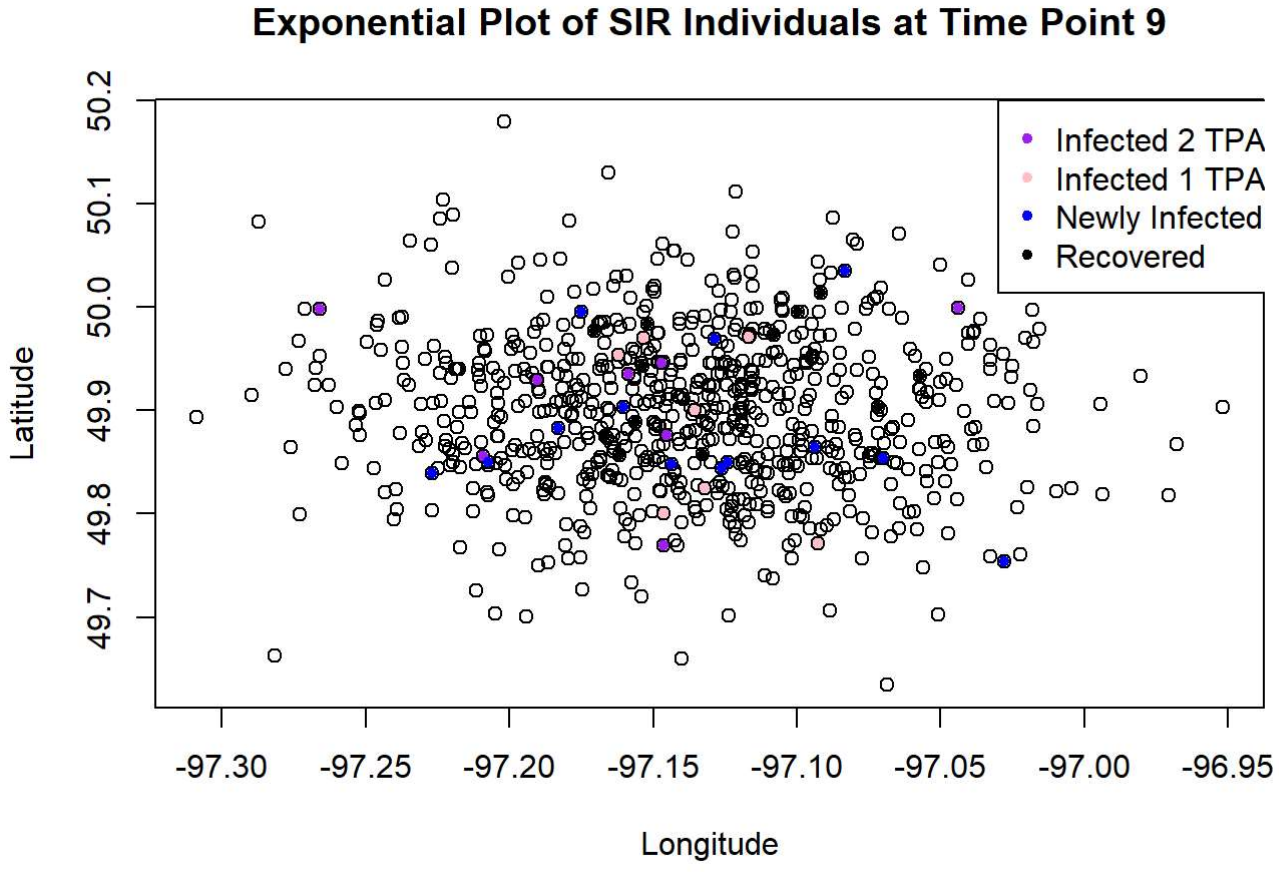
Exponential Heat Maps and Maps of Infected Individuals

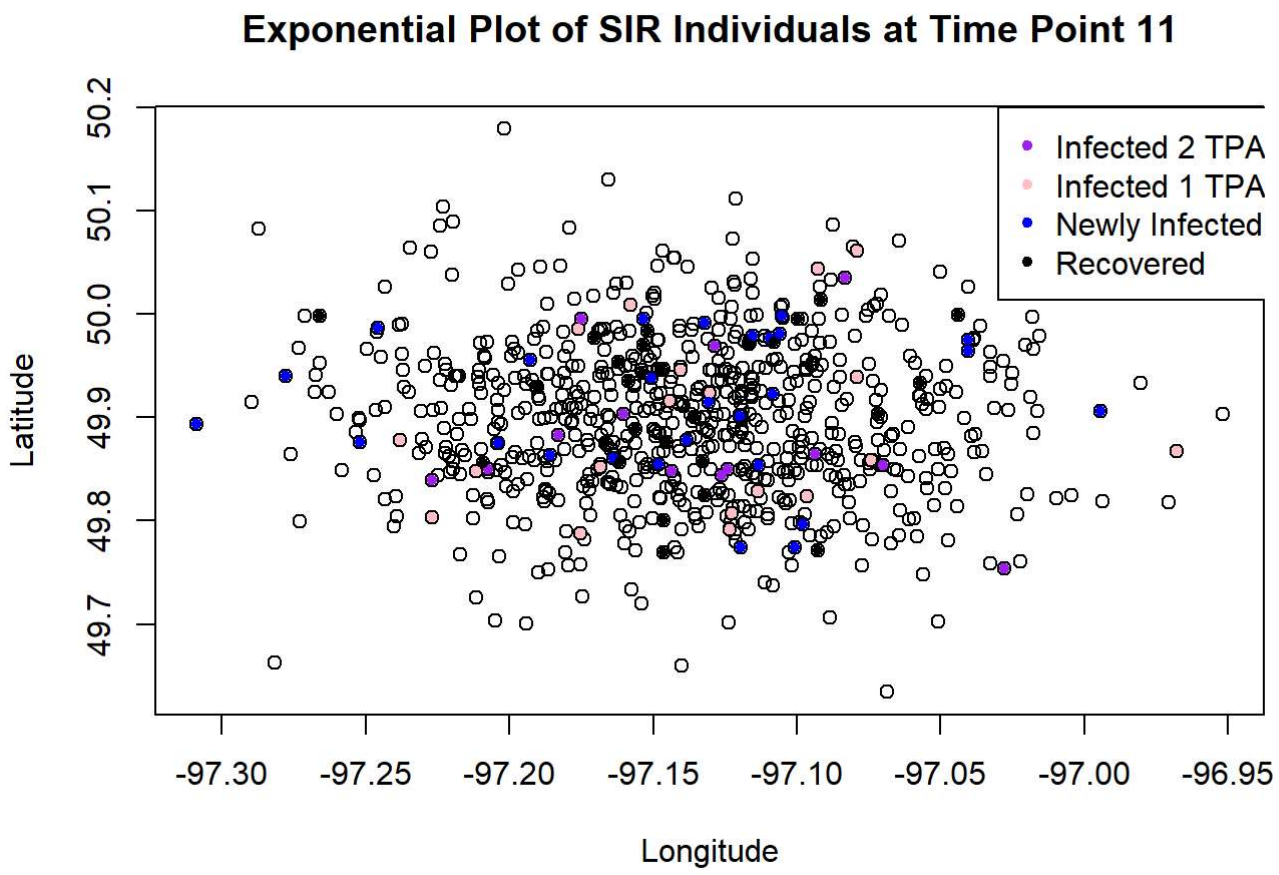
```
#Setting the titles that we will use for the plots
infectedTitleExp = title_generator(T, F, 3, 20)
heatMapTitleExp = title_generator(T, T, 3, 20)

set.seed(3001)
epigenExp=epigenExponential(samples, 0.001, 0.0005, 20)

#TPA stands for time point ago
activeInfectionsDisplayer(2, 1, 1, 20, epigenExp$inftime, samples, infectedTitleExp)
```

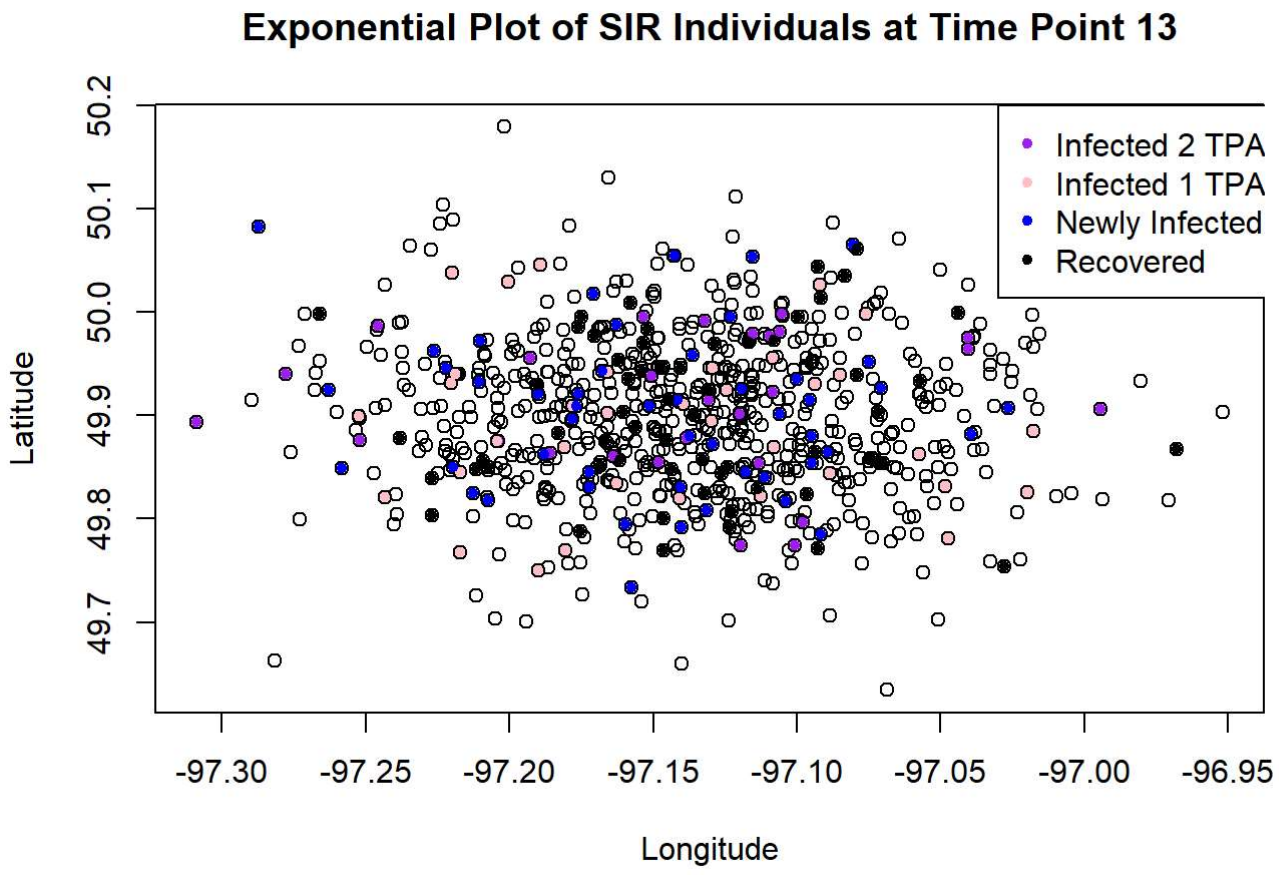

Exponential Plot of SIR Individuals at Time Point 3
Infected 2 TPA
Infected 1 TPA
Newly Infected
Recovered
Latitude
Longitude
49.7
49.8
49.9
50.0
50.1
50.2
-97.30
-97.25
-97.20
-97.15
-97.10
-97.05
-97.00
-96.95

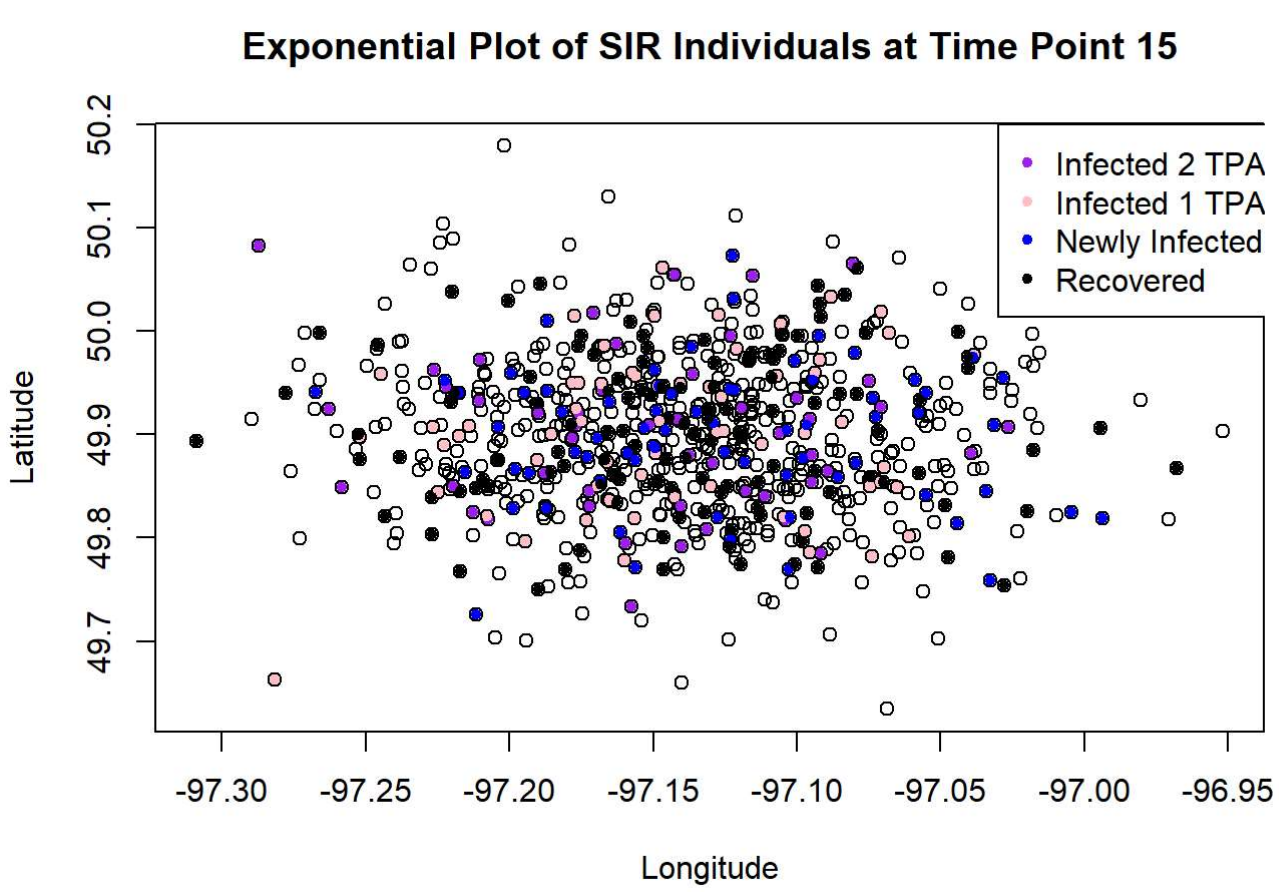

Exponential Plot of SIR Individuals at Time Point 5
Infected 2 TPA
Infected 1 TPA
Newly Infected
Recovered
Latitude
Longitude
49.7
49.8
49.9
50.0
50.1
50.2
-97.30
-97.25
-97.20
-97.15
-97.10
-97.05
-97.00
-96.95

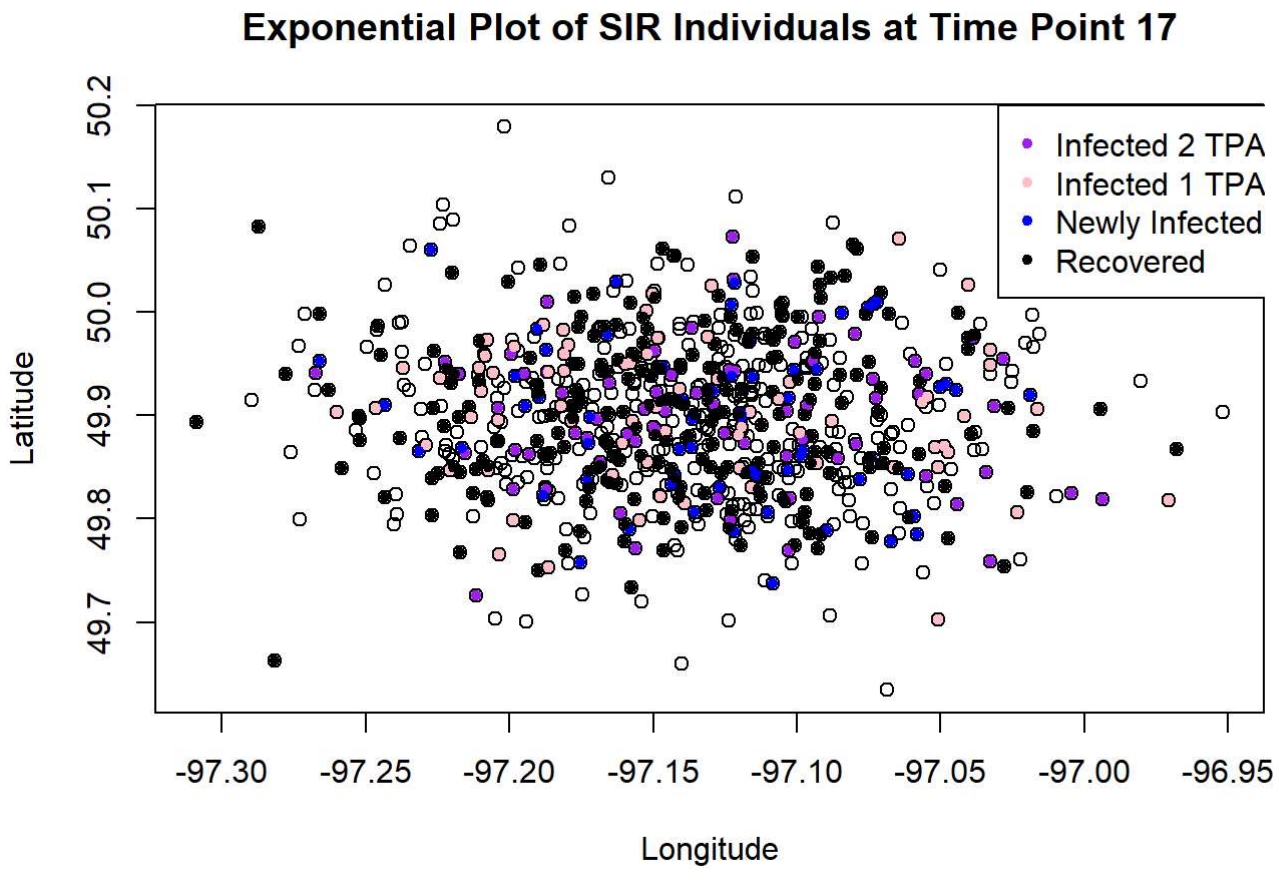

Exponential Plot of SIR Individuals at Time Point 7
Infected 2 TPA
Infected 1 TPA
Newly Infected
Recovered
Latitude
Longitude
49.7
49.8
49.9
50.0
50.1
50.2
-97.30
-97.25
-97.20
-97.15
-97.10
-97.05
-97.00
-96.95

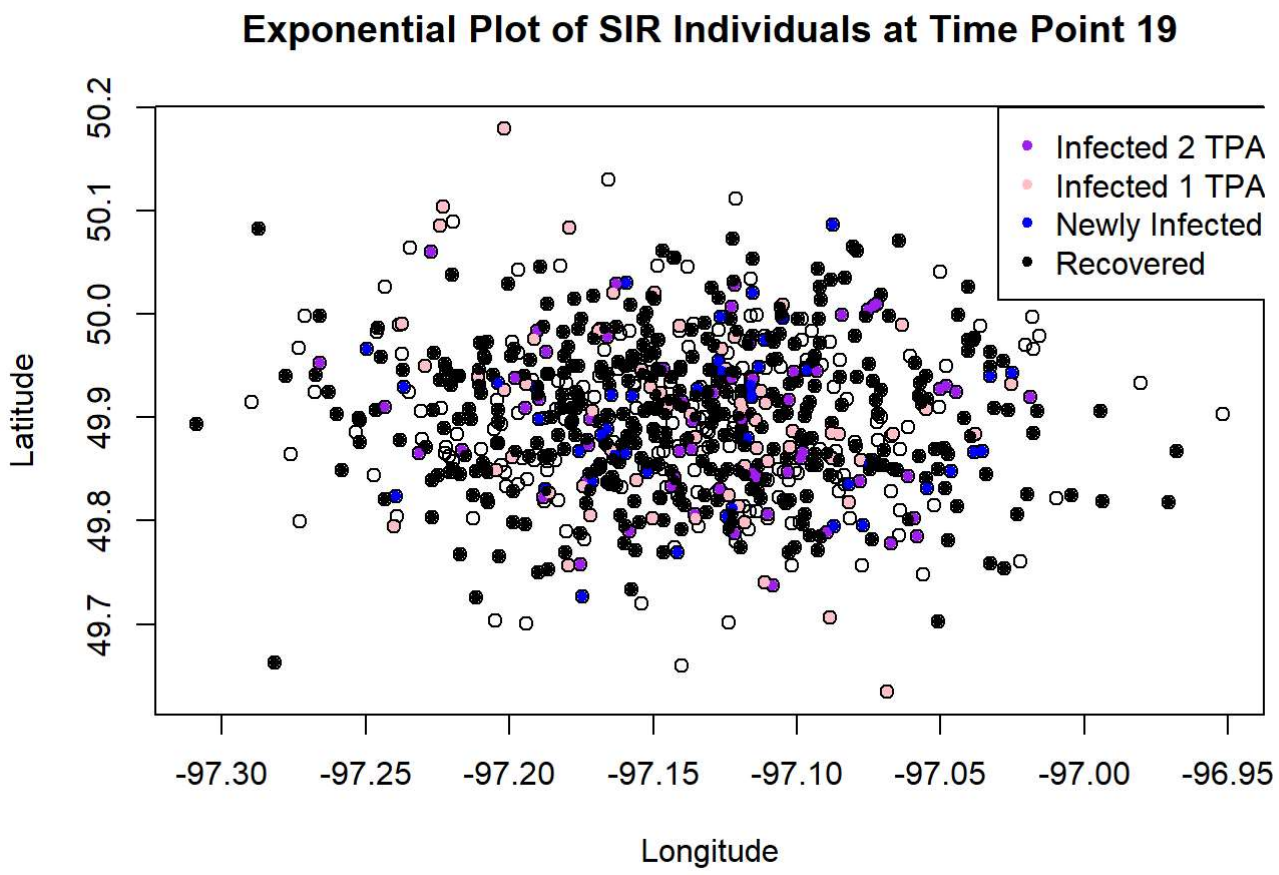

Exponential Plot of SIR Individuals at Time Point 9
Infected 2 TPA
Infected 1 TPA
Newly Infected
Recovered
Latitude
Longitude
49.7
49.8
49.9
50.0
50.1
50.2
-97.30
-97.25
-97.20
-97.15
-97.10
-97.05
-97.00
-96.95

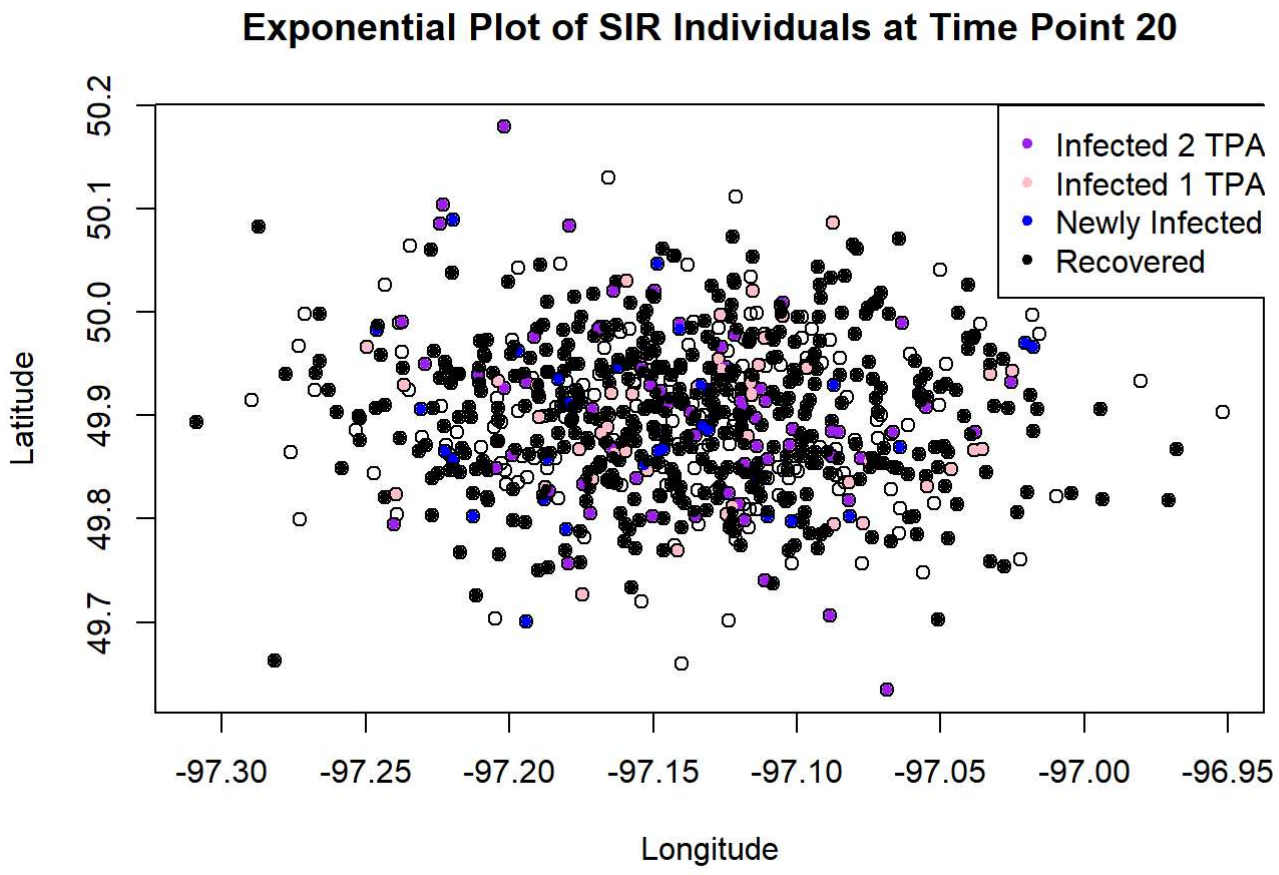

Exponential Plot of SIR Individuals at Time Point 11
Infected 2 TPA
Infected 1 TPA
Newly Infected
Recovered
Latitude
Longitude
49.7
49.8
49.9
50.0
50.1
50.2
-97.30
-97.25
-97.20
-97.15
-97.10
-97.05
-97.00
-96.95


Exponential Plot of SIR Individuals at Time Point 13
Infected 2 TPA
Infected 1 TPA
Newly Infected
Recovered
Latitude
Longitude
49.7
49.8
49.9
50.0
50.1
50.2
-97.30
-97.25
-97.20
-97.15
-97.10
-97.05
-97.00
-96.95

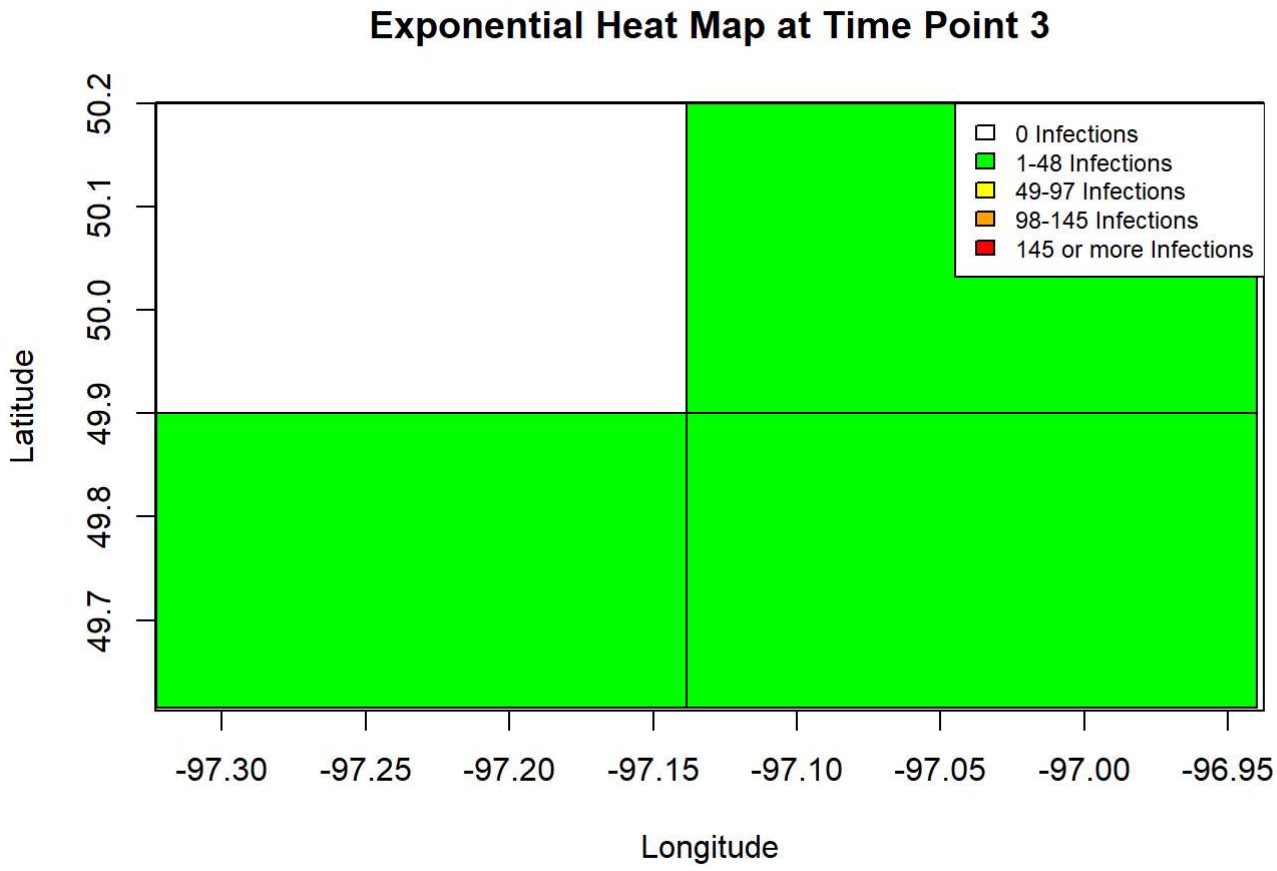

Exponential Plot of SIR Individuals at Time Point 15
Infected 2 TPA
Infected 1 TPA
Newly Infected
Recovered
Latitude
Longitude
49.7
49.8
49.9
50.0
50.1
50.2
-97.30
-97.25
-97.20
-97.15
-97.10
-97.05
-97.00
-96.95

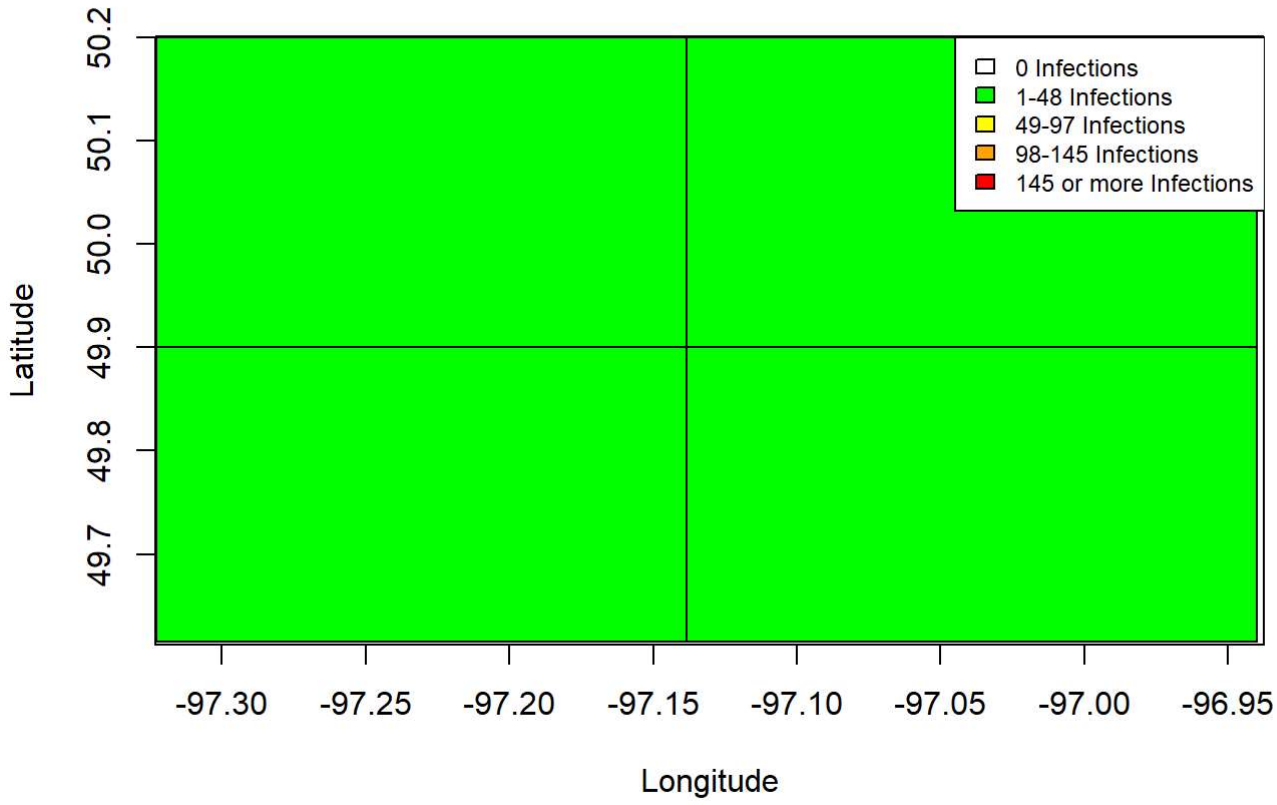

Exponential Plot of SIR Individuals at Time Point 17
Infected 2 TPA
Infected 1 TPA
Newly Infected
Recovered
Latitude
Longitude
49.7
49.8
49.9
50.0
50.1
50.2
-97.30
-97.25
-97.20
-97.15
-97.10
-97.05
-97.00
-96.95

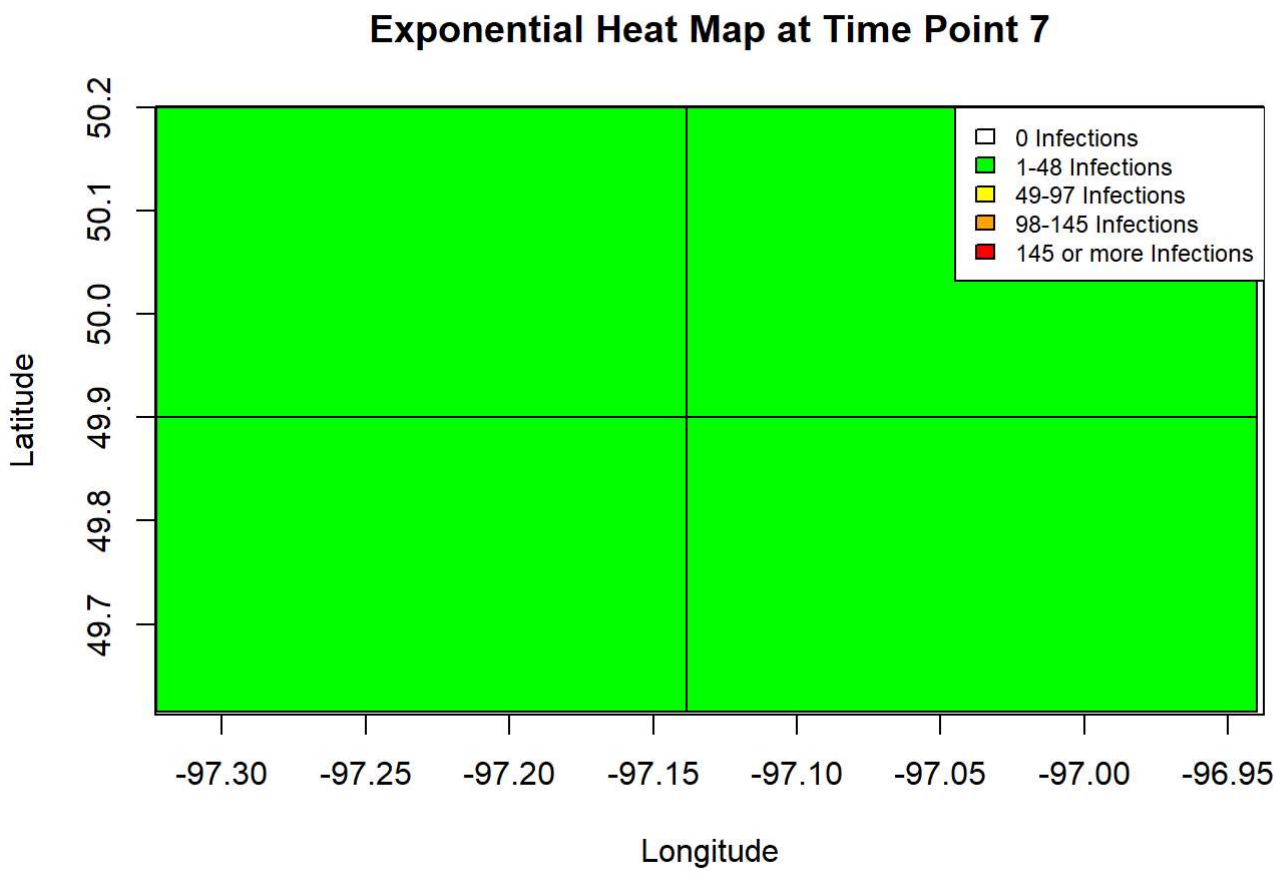

Exponential Plot of SIR Individuals at Time Point 19
Infected 2 TPA
Infected 1 TPA
Newly Infected
Recovered
Latitude
Longitude
49.7
49.8
49.9
50.0
50.1
50.2
-97.30
-97.25
-97.20
-97.15
-97.10
-97.05
-97.00
-96.95

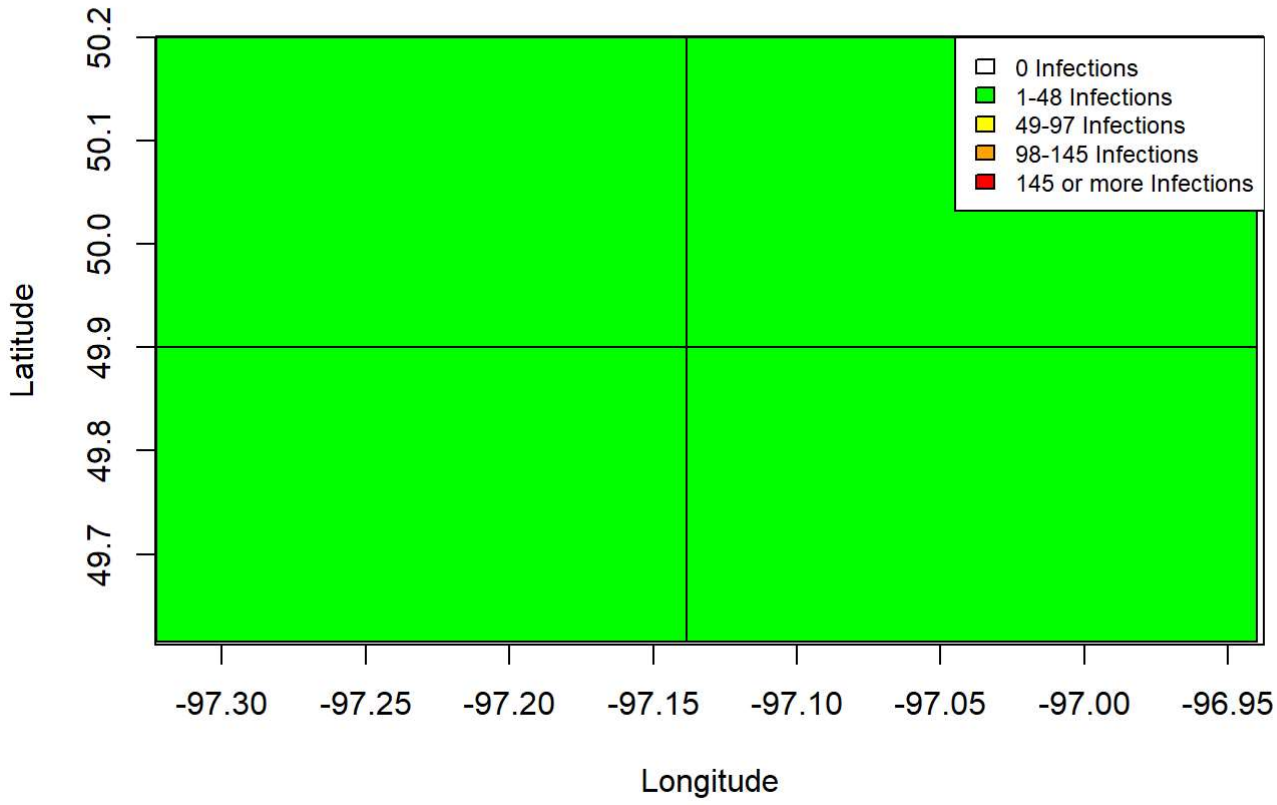

Exponential Plot of SIR Individuals at Time Point 20
Infected 2 TPA
Infected 1 TPA
Newly Infected
Recovered
Latitude
Longitude
49.7
49.8
49.9
50.0
50.1
50.2
-97.30
-97.25
-97.20
-97.15
-97.10
-97.05
-97.00
-96.95

```
heatMapDisplayer(2, 1, 1, 20, epigenExp$inftime, samples, heatMapTitleExp)
```

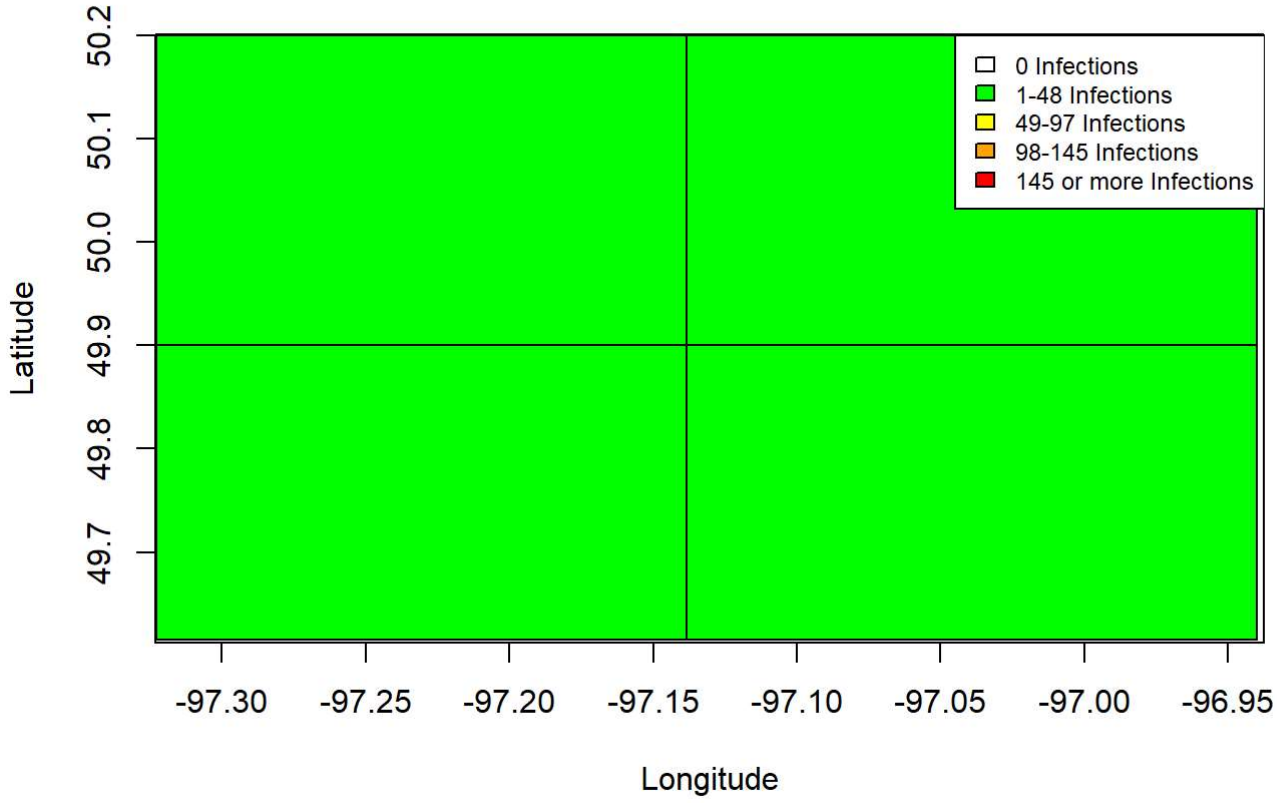

Exponential Heat Map at Time Point 3
0 Infections
1-48 Infections
49-97 Infections
98-145 Infections
145 or more Infections
Latitude
50.2
50.1
50.0
49.9
49.8
49.7
-97.30
-97.25
-97.20
-97.15
-97.10
-97.05
-97.00
-96.95
Longitude


Exponential Heat Map at Time Point 5
0 Infections
1-48 Infections
49-97 Infections
98-145 Infections
145 or more Infections
Latitude
50.2
50.1
50.0
49.9
49.8
49.7
-97.30
-97.25
-97.20
-97.15
-97.10
-97.05
-97.00
-96.95
Longitude

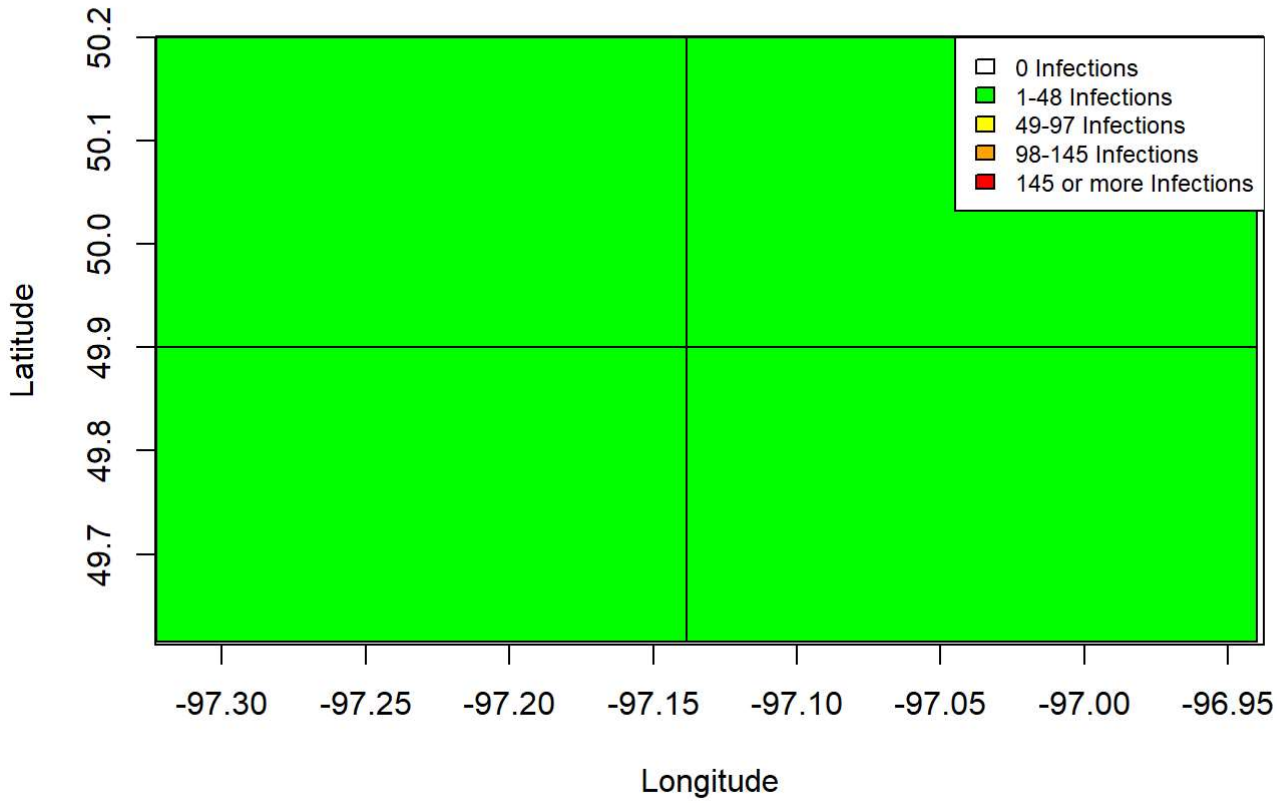

Exponential Heat Map at Time Point 7
0 Infections
1-48 Infections
49-97 Infections
98-145 Infections
145 or more Infections
Latitude
50.2
50.1
50.0
49.9
49.8
49.7
-97.30
-97.25
-97.20
-97.15
-97.10
-97.05
-97.00
-96.95
Longitude

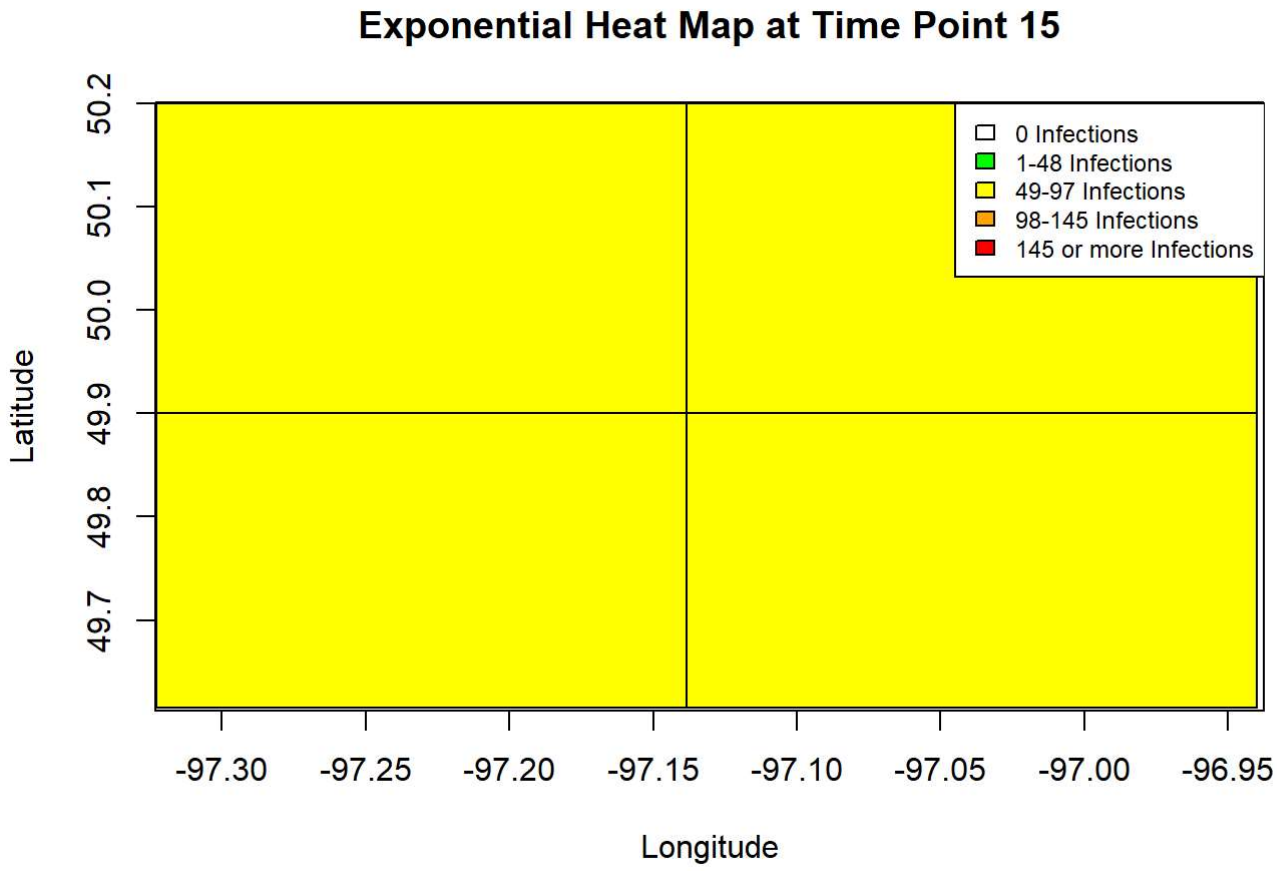

Exponential Heat Map at Time Point 9
0 Infections
1-48 Infections
49-97 Infections
98-145 Infections
145 or more Infections
Latitude
50.2
50.1
50.0
49.9
49.8
49.7
-97.30
-97.25
-97.20
-97.15
-97.10
-97.05
-97.00
-96.95
Longitude

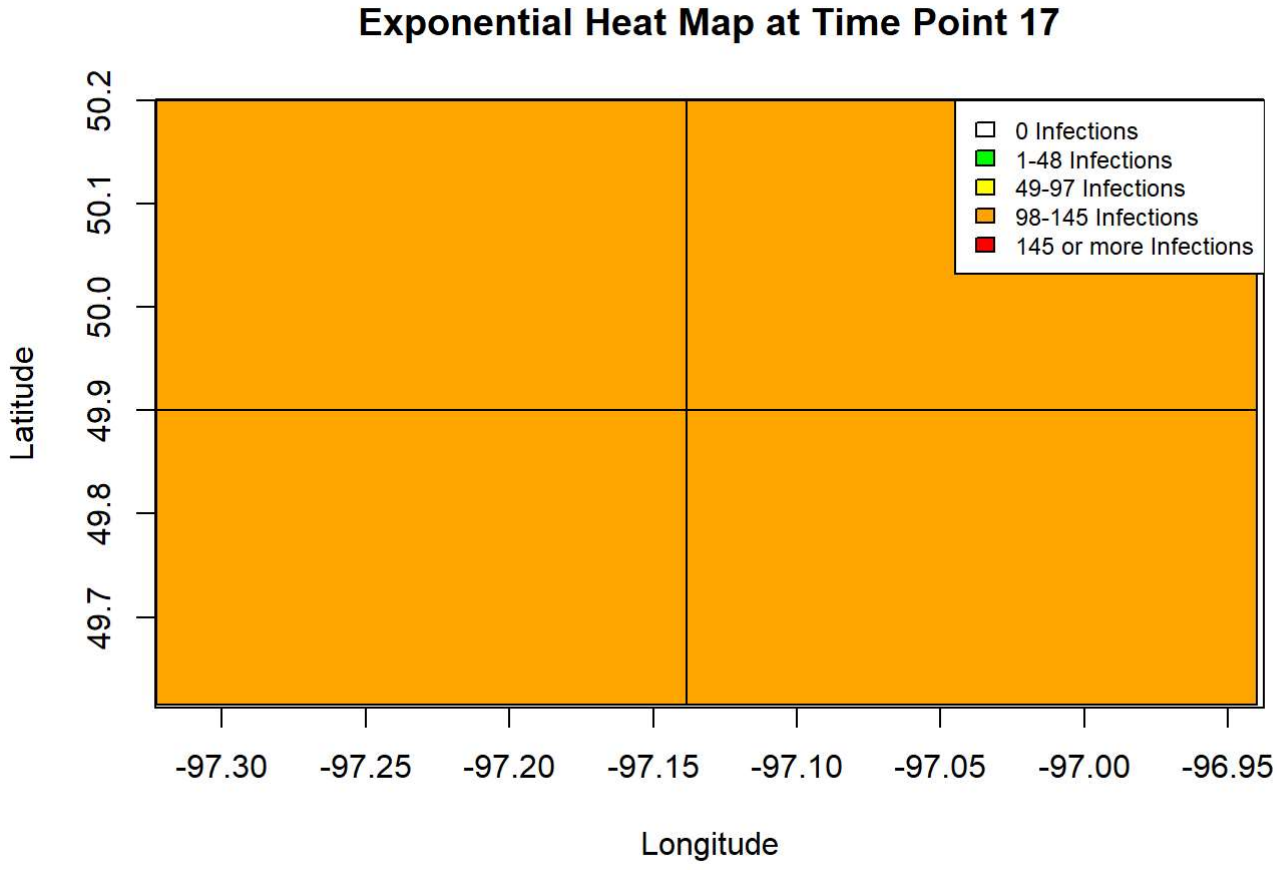
Exponential Heat Map at Time Point 11

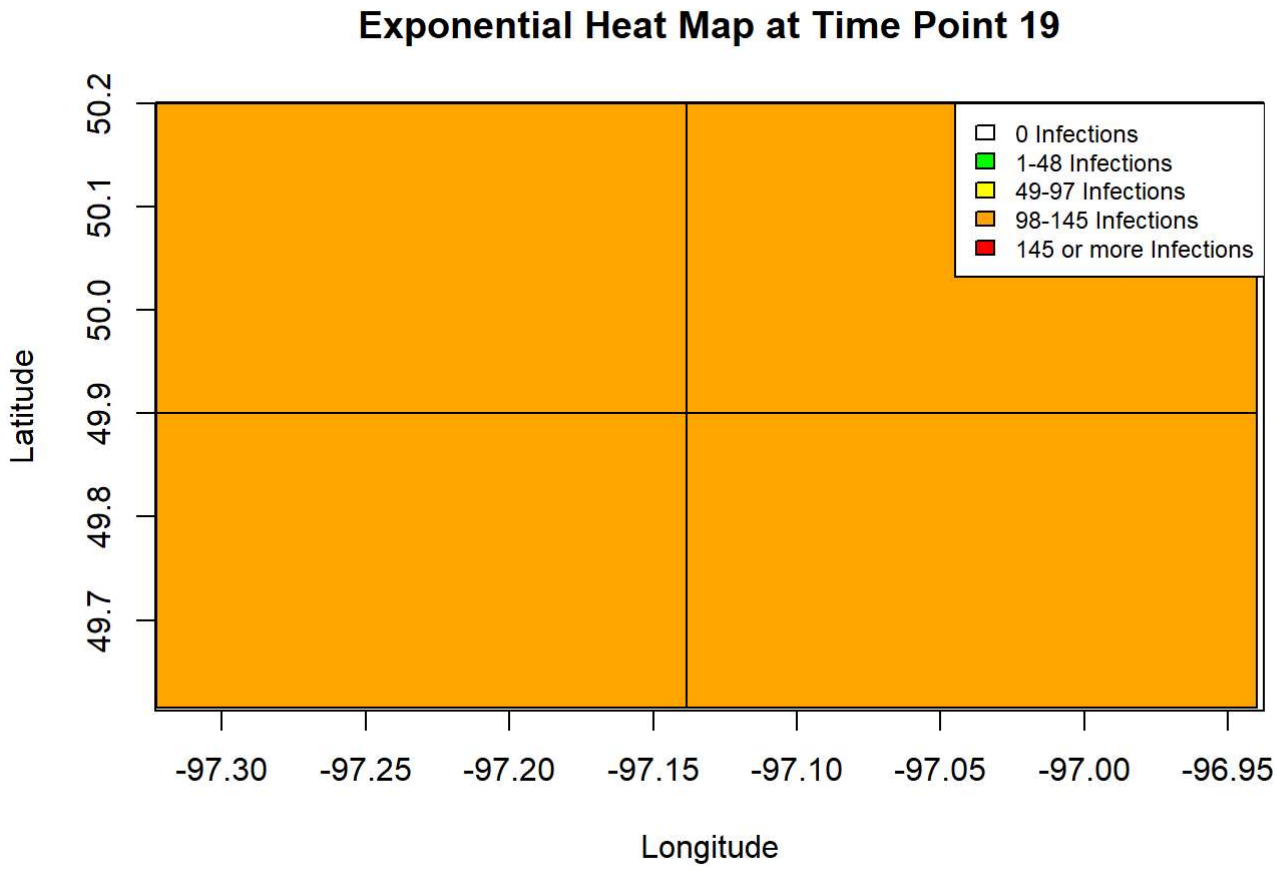


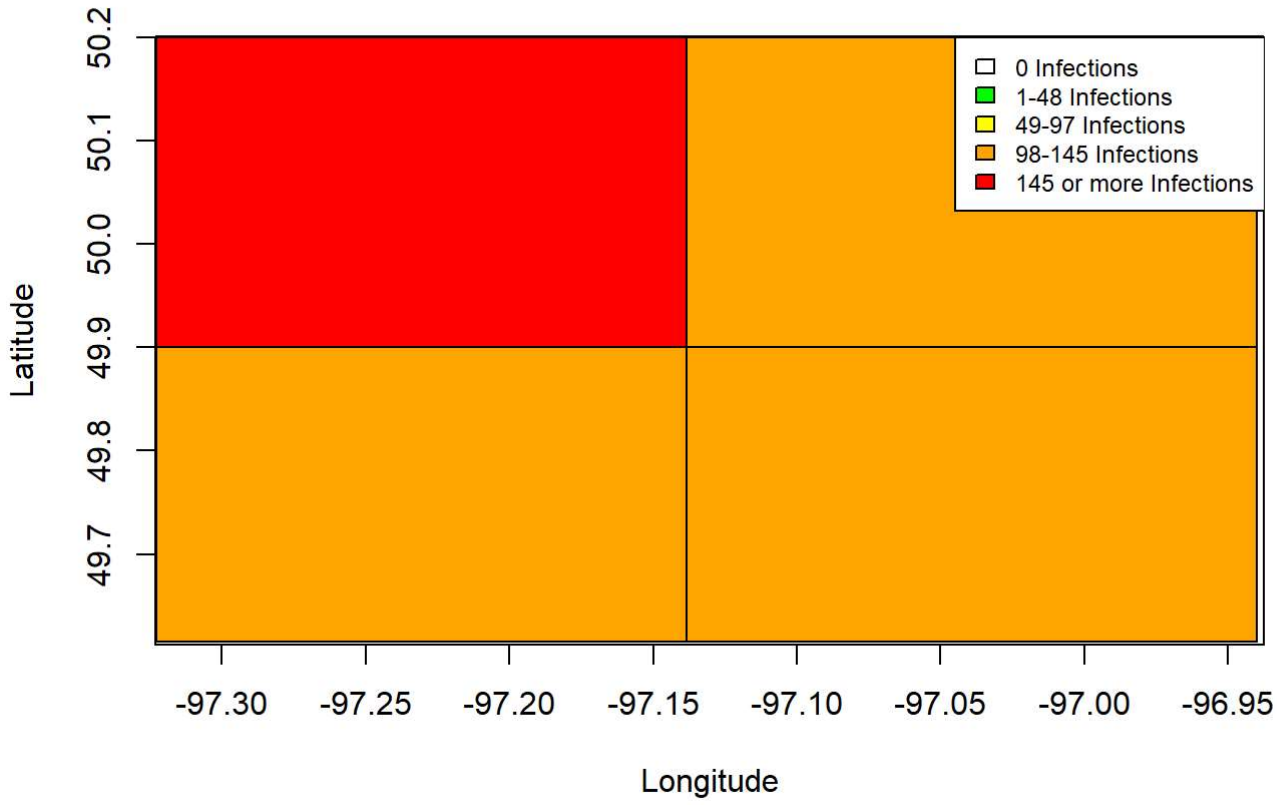
Exponential Heat Map at Time Point 13




Exponential Heat Map at Time Point 15
0 Infections
1-48 Infections
49-97 Infections
98-145 Infections
145 or more Infections
Latitude
50.2
50.1
50.0
49.9
49.8
49.7
-97.30
-97.25
-97.20
-97.15
-97.10
-97.05
-97.00
-96.95
Longitude


Exponential Heat Map at Time Point 17
0 Infections
1-48 Infections
49-97 Infections
98-145 Infections
145 or more Infections
Latitude
50.2
50.1
50.0
49.9
49.8
49.7
-97.30
-97.25
-97.20
-97.15
-97.10
-97.05
-97.00
-96.95
Longitude


Exponential Heat Map at Time Point 19
0 Infections
1-48 Infections
49-97 Infections
98-145 Infections
145 or more Infections
Latitude
50.2
50.1
50.0
49.9
49.8
49.7
-97.30
-97.25
-97.20
-97.15
-97.10
-97.05
-97.00
-96.95
Longitude


Exponential Heat Map at Time Point 20
0 Infections
1-48 Infections
49-97 Infections
98-145 Infections
145 or more Infections
Latitude
50.2
50.1
50.0
49.9
49.8
49.7
-97.30
-97.25
-97.20
-97.15
-97.10
-97.05
-97.00
-96.95
Longitude