SIR_ER_Network

Ananyo Shreya

2025-11-17

Short overview: Erdos-Renyi (ER) network

An Erdos-Renyi (ER) random graph is one of the simplest network models. In the variant, we fix the number of nodes `n` and include every possible edge independently with probability `p` known as $G(n, p)$.

Pseudo Algorithm

Step 1 Set a seed. Fix number of nodes n(we have taken n=80).

Step 2 Generate $U_1, \ldots, U_{n(n-1)/2}$ many uniform random variables independently.

Step 3 Randomly select 1 node and define its state to be `Infected`, others are `Susceptible`. We have taken number of `Recovered` person 0 initially.

Step 4 Take a sequence of probabilities (the same probability attached to the edges in ER network) and for each probability construct adjacency matrix by comparing uniform observations with probability to generate an ER network.

Step 5 Run a SIR model on this network with fixed recovery and infection rates until there are no infected people (We have taken recovery rate = 0.25 and different R_0 values). Store the 4 measures : `Total number of infected people`, `Maximum number of newly infected people on a single day`, `Day of Maximum Infection`, and `Duration of epidemic`.

Step 6 Repeat Step 5 several times (We did 10 times).

Step 7 Repeat Step 2 to 6 several times (We did 15 times) and obtain mean and standard deviations for the 4 measures at each probability using those 250 values in each case. Then plot the means with respective 95% confidence interval for various probabilities while keeping other parameters intact.

Step 8 We have repeated each of the plots for some values of R_0 (Mentioned later) and plotted all in a single frame in each case for easy comparison.

Step 9 We have also obtained the threshold probability for each R_0 which will lead to the event that at least 95% of the population gets infected and another threshold probability for each R_0 that caused longest span from starting day of epidemic to cause maximum new infection on a day.

```
## Warning: package 'knitr' was built under R version 4.3.3
```

Table 1: Experimental Design Summary

Label	Experiment_Design	Connection_Prob	R0	Recovery_Rate
Exp. 1	Monofactorial experiment with focus on change in connection probability in ER network	0.001, 0.005, 0.008, 0.01, 0.03, 0.04, 0.05, 0.06, 0.07, 0.08, 0.09, 0.1, 0.12, 0.14, 0.16, 0.18, 0.25, 0.3, 0.4, 0.5, 0.7, 0.8, 0.9, 0.97	0.8	0.25
Exp. 2	Monofactorial experiment to investigate the effect of changes in the value of R0	0.3	0.5, 0.65, 0.8, 1.0, 1.2	0.25
Exp. 3	Full factorial experiment to study the impact of interactions between parameters	0.001, 0.005, 0.008, 0.01, 0.03, 0.04, 0.05, 0.06, 0.07, 0.08, 0.09, 0.1, 0.12, 0.14, 0.16, 0.18, 0.25, 0.3, 0.4, 0.5, 0.7, 0.8, 0.9, 0.97	0.5, 0.65, 0.8, 1.0, 1.2	0.25

```r
# 1. Function to create Erdos-Renyi Network
er_graph_unif <- function(p, nodes, unif_matrix){
  adj_matrix <- matrix(0, nrow= nodes, ncol=nodes)
  adj_matrix[which(unif_matrix <= p)] <- 1
  er_graph <- graph_from_adjacency_matrix(adj_matrix, mode="undirected")
  return(er_graph)
}

# 2. Function for SIR model
SIR <- function(states, er_network, infection_rate, recovery_day){
  new_inf_vector <- c()
  start_day <- rep(-1,length(states))
  start_day[which(states == "I")] <- 0
  t=1
  while (sum(states == "I") != 0){
    # infection process
    new_infected <- c()
    infected_nodes <- which(states == "I")
    for (i in infected_nodes){
      neighbors_i <- neighbors(er_network, i)
      s_neighbors <- neighbors_i[states[neighbors_i] == "S"]
      for (s_node in s_neighbors){
        if (runif(1) < infection_rate){
          states[s_node] <- "I"
          new_infected <- c(new_infected, s_node)
        }
      }
    }
    start_day[new_infected] <- t
    new_inf_vector <- c(new_inf_vector, length(new_infected))

    # recovery process
    if (t>=recovery_day){
      recovered <- which(start_day == t-recovery_day)
      states[recovered] <- "R"
    }
```

```
    t <- t+1
  }
  total_inf <- sum(states == "R")
  max_inf <- max(new_inf_vector)
  peak_day <- min(which(new_inf_vector == max_inf))
  end_day <- t

  return(c(total_inf/length(states), max_inf/length(states), peak_day, end_day))
}


# 3. Function for each simulation
Evaluation <- function(nodes, probs, beta, gamma, initial_infected){
  #Uniform numbers generation for a simulation
  unif_matrix <- matrix(0, nodes, nodes)
  unif_matrix[lower.tri(unif_matrix, diag = FALSE)] <- runif((nodes*(nodes-1))/2)
  unif_matrix <- unif_matrix +t(unif_matrix)
  unif_matrix[which(unif_matrix==0)] <- 1

  all_sir <- list()
  for (l in 1:10){
    # initial states of nodes for a simulation
    states <- rep("S", nodes)
    states[initial_infected] <- "I"
    vec_measures <- matrix(nrow=0, ncol=4)
    for (b in beta){
      for (p in probs){
        er_graph <- er_graph_unif(p, nodes, unif_matrix) #Erdos Renyi graph
        values <- SIR(states, er_graph, b, gamma)
        vec_measures <- rbind(vec_measures, values)
      }
    }
    all_sir[[l]] <- vec_measures
  }
  return(list(list = all_sir, matrix = unif_matrix))
}


# 4. Function for generating summary measures
Simulated_df <- function(sim,nodes, probs, beta, gamma, initial_infected){

  result <- lapply(1:sim, function(x) Evaluation(nodes, probs, beta, gamma, initial_infected))

  # Extract all lists
  mat_list <- lapply(result, function(x) x$list)

  mat_array <- array(unlist(mat_list), dim = c(length(beta)*length(probs), 4, sim*10))

  mean_matrix <- apply(mat_array, c(1, 2), mean, na.rm = TRUE)
  lower_matrix <- apply(mat_array, c(1, 2), function(x) quantile(x, 0.025, na.rm = TRUE))
  upper_matrix <- apply(mat_array, c(1, 2), function(x) quantile(x, 0.975, na.rm = TRUE))

  num_r0 <- length(beta)
```

```r
split_indices <- split(1:(length(probs) * num_r0), rep(1:num_r0, each=length(probs)))
mean_list <- lapply(split_indices, function(idx) mean_matrix[idx, ])
lower_list <- lapply(split_indices, function(idx) lower_matrix[idx, ])
upper_list <- lapply(split_indices, function(idx) upper_matrix[idx, ])


#Total infected
df_total_inf <- data.frame()
for (i in 1:num_r0){
  df_temp <- data.frame(
    Probs = probs,
    R0 = beta[i],
    Mean = mean_list[[i]][,1],
    Upper = upper_list[[i]][,1],
    Lower = lower_list[[i]][,1]
  )
  df_total_inf <- rbind(df_total_inf, df_temp)
}

#Max infected
df_max_inf <- data.frame()
for (i in 1:num_r0){
  df_temp <- data.frame(
    Probs = probs,
    R0 = beta[i],
    Mean = mean_list[[i]][,2],
    Upper = upper_list[[i]][,2],
    Lower = lower_list[[i]][,2]
  )
  df_max_inf <- rbind(df_max_inf, df_temp)
}

#Peak day
df_peak_day <- data.frame()
for (i in 1:num_r0){
  df_temp <- data.frame(
    Probs = probs,
    R0 = beta[i],
    Mean = mean_list[[i]][,3],
    Upper = upper_list[[i]][,3],
    Lower = lower_list[[i]][,3]
  )
  df_peak_day <- rbind(df_peak_day, df_temp)
}

#End day
df_end_day <- data.frame()
for (i in 1:num_r0){
  df_temp <- data.frame(
    Probs = probs,
    R0 = beta[i],
    Mean = mean_list[[i]][,4],
    Upper = upper_list[[i]][,4],
```

```
      Lower = lower_list[[i]][,4]
    )
    df_end_day <- rbind(df_end_day, df_temp)
  }

  D <- list(
    total_inf = df_total_inf,
    max_inf = df_max_inf,
    peak = df_peak_day,
    end = df_end_day
  )

  return(D)
}
```

Parameter Initializations

```
### Now comes the main code:
nodes <- 500
probs <- c(0.001, 0.005, 0.008, 0.01, 0.03, 0.04, 0.05, 0.06, 0.07, 0.08, 0.09, 0.1,
           0.12, 0.14, 0.16, 0.18,0.25, 0.3, 0.4, 0.5, 0.7, 0.8, 0.9, 0.97)
beta <- c(0.125, 0.1625,0.2,0.25,0.3) # Infection rate
recovery_day <- 4

initial <- 1
initial_infected <- sample(1:nodes, initial)

sim <- 15 #No of simulations

sim_df1 <- Simulated_df(sim, nodes, probs, beta, recovery_day, initial_infected)
```

Plots and Observations

We compute mean and 95% confidence intervals across simulations for each `p` and plot.

Threshold Value of probability exceeding which 95% of the individuals got infected for every β

```
##     Beta    p
## 1 0.1250 0.03
## 2 0.1625 0.03
## 3 0.2000 0.03
## 4 0.2500 0.03
## 5 0.3000 0.01
```

Threshold Value of probability at which day of maximum infection peaked for every β

```
##     Beta     p
## 1 0.1250 0.008
## 2 0.1625 0.008
```

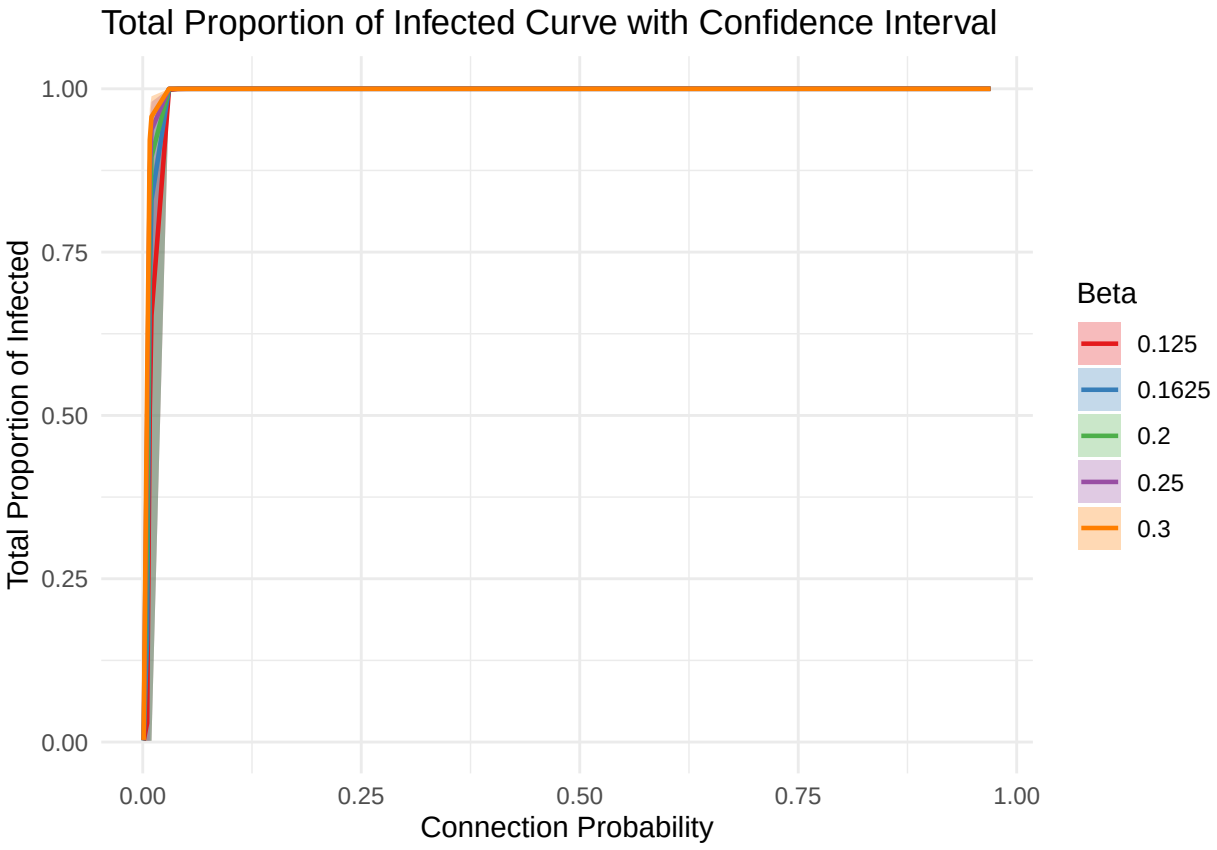

Figure 1: Total Proportion of Infected Curve with Confidence Interval (Recovery days=4)

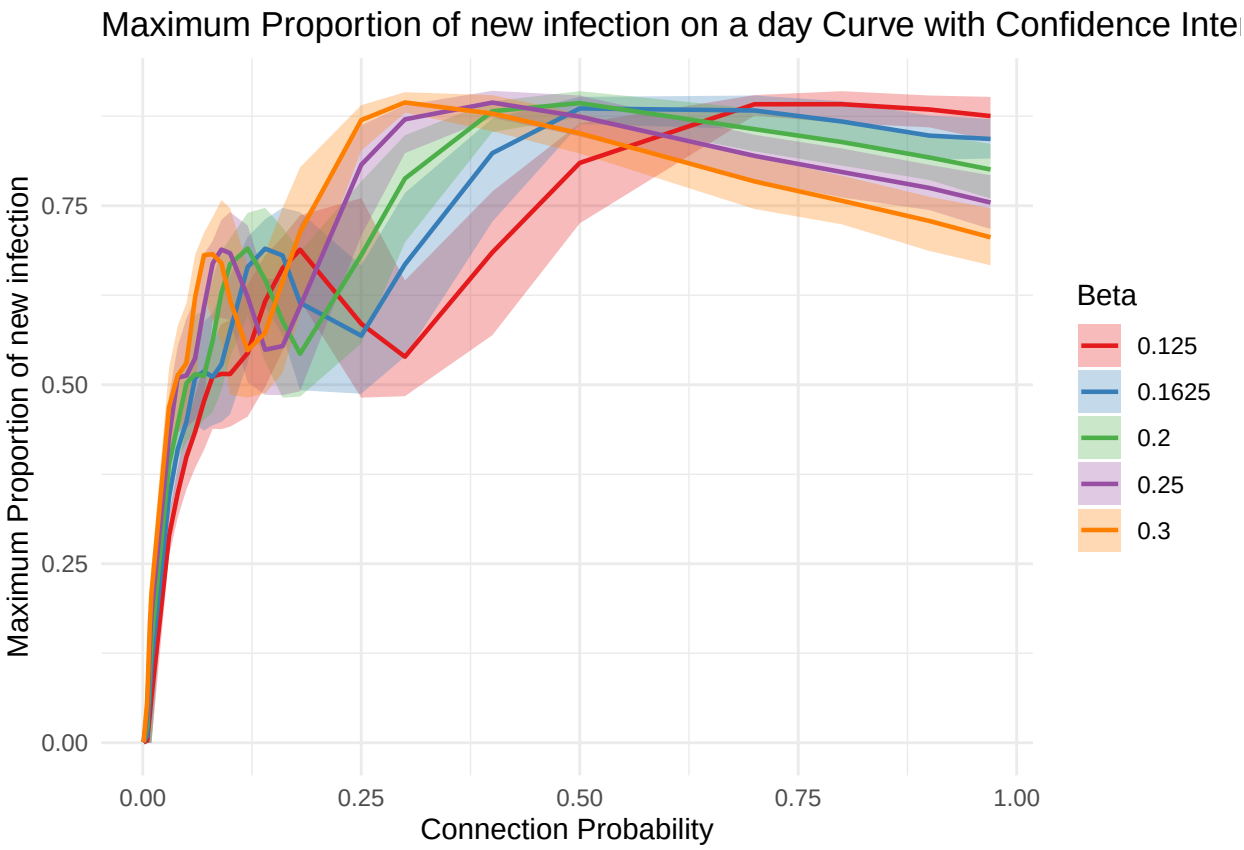

Figure 2: Maximum Proportion of Infection on a day Curve with Confidence Interval (Recovery days=4)

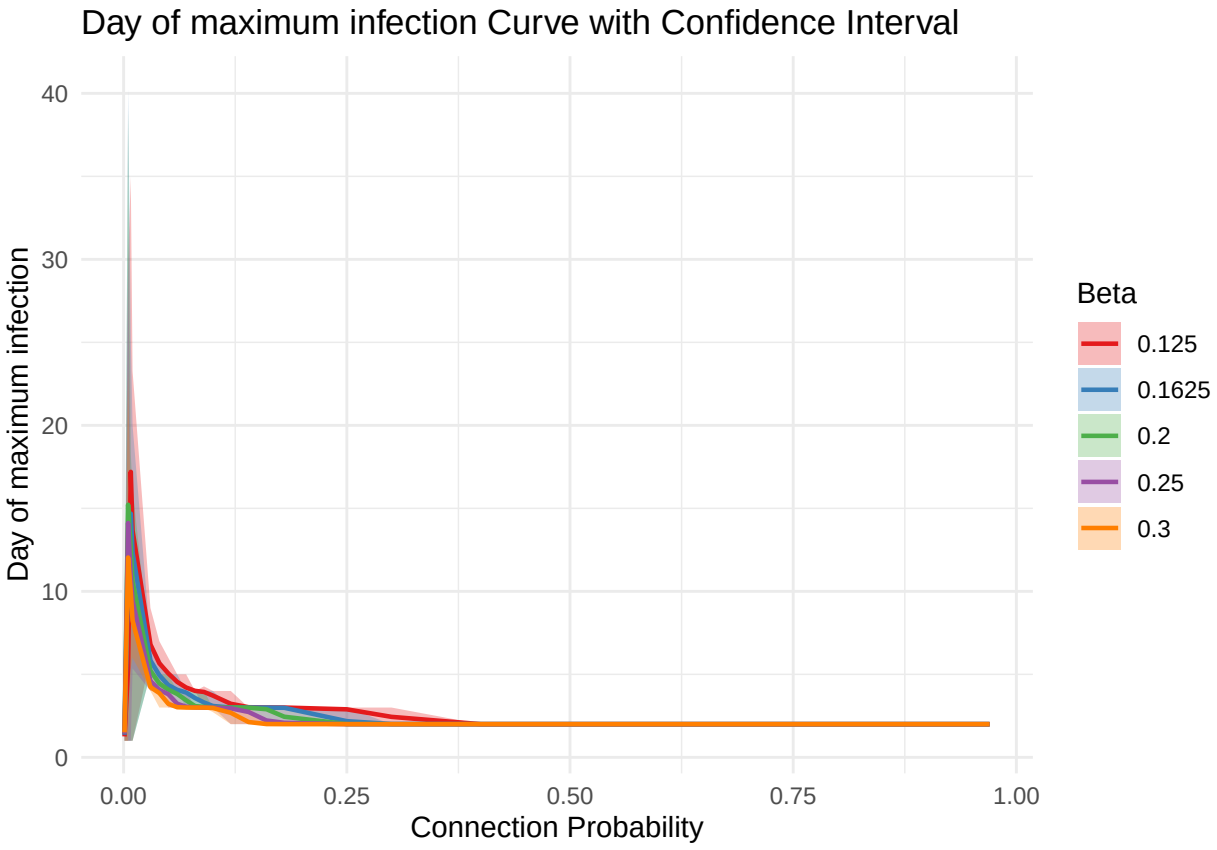

Figure 3: Day of maximum infection Curve with Confidence Interval (Recovery days=4)

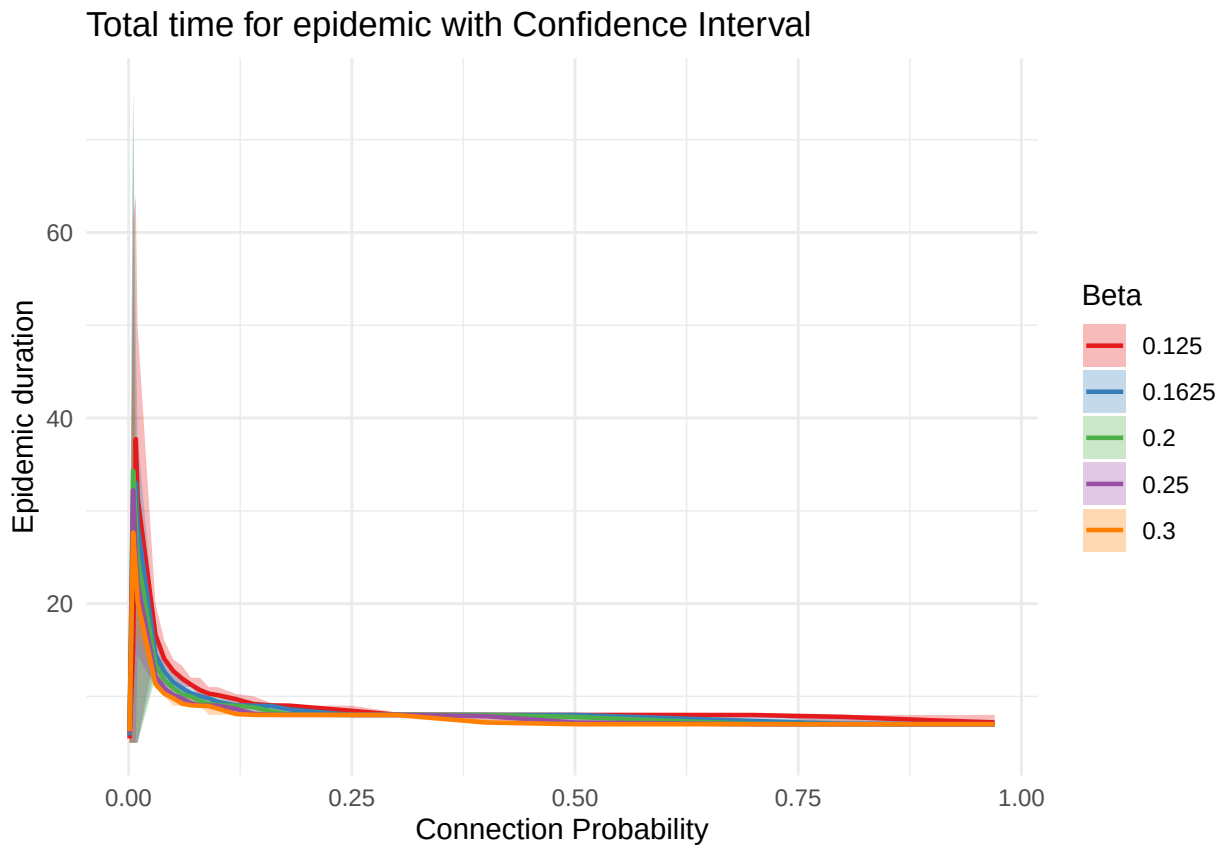

Figure 4: Epidemic duration with Confidence Interval (Recovery days=4)

```
## 3 0.2000 0.005
## 4 0.2500 0.005
## 5 0.3000 0.005
```