Risk_model_Code_and_description

Courtney Schreiner

2024-10-08

Step by step for creating risk model code

Community model parameters

```
community_pop <- 100000 #pop size of the larger community
init_conds <- c(S = community_pop-1, I = 1, R = 0)
print(init_conds)
```

```
##     S     I     R
## 99999     1     0
```

```
parms <- data.frame(bet =0.0000035 , gam = 1/21)
print(parms)
```

```
##       bet        gam
## 1 3.5e-06 0.04761905
```

```
times <- seq(from = 1, to = 144, by = 1)
print(times)
```

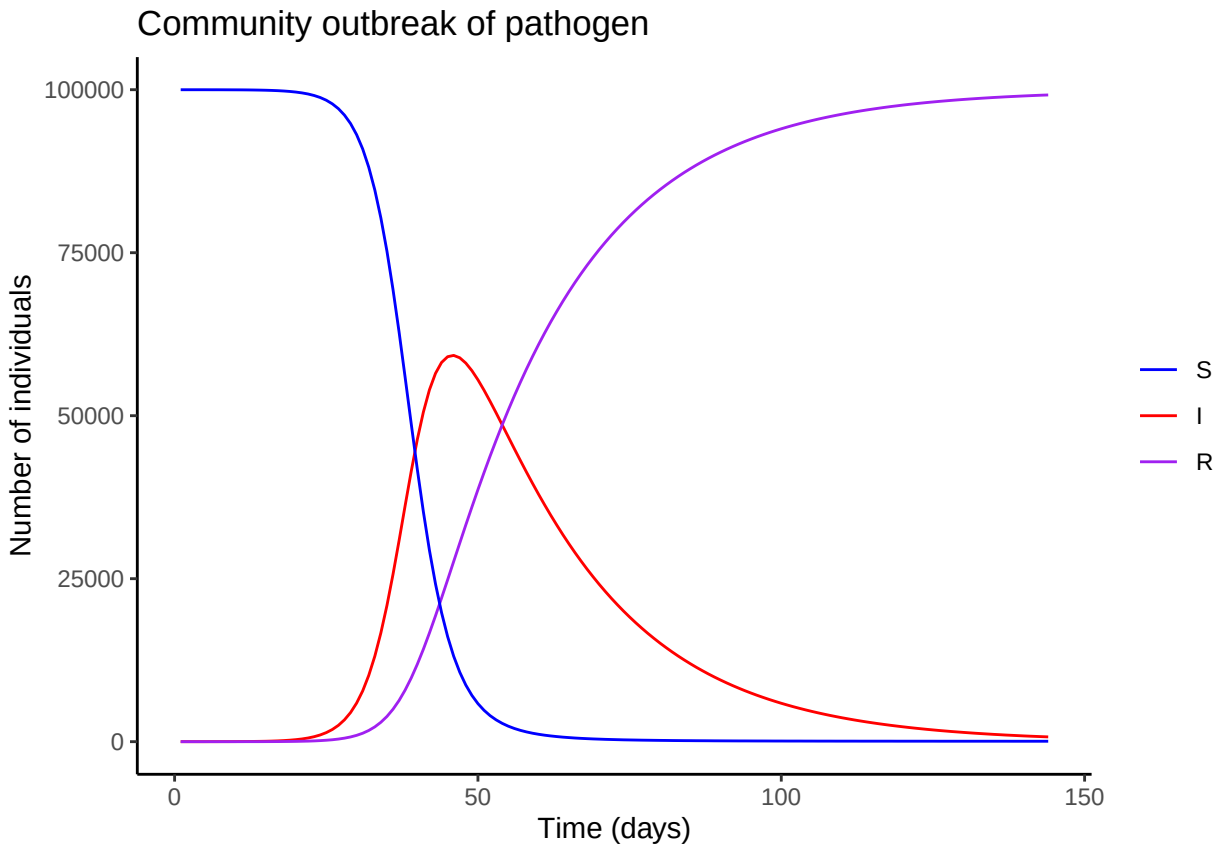
```
##   [1]   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18
##  [19]  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36
##  [37]  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54
##  [55]  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72
##  [73]  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90
##  [91]  91  92  93  94  95  96  97  98  99 100 101 102 103 104 105 106 107 108
## [109] 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126
## [127] 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144
```

Community modelfunction/equations

```
SIR_community_model <- function(t, x, parms) {
  S <- x[1]
  I <- x[2]
  R <- x[3]

  with(parms, {
    dS <- -bet*S*I
    dI <- bet*S*I - gam*I
```

```
    dR <- gam*I

    dt <- c(dS,dI,dR)
    return(list(dt))
  })}
```

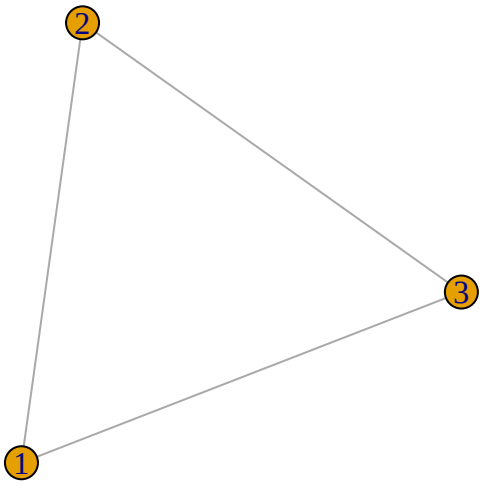
Community model output



Make building

```
small_bld_3_rooms <- matrix(c(c(0,1,1),
                              c(1,0,1),
                              c(1,1,0)),nrow = 3,ncol = 3)
small_bld_5_rooms <- matrix(c(c(0,1,1,1,1),
                                    c(1,0,1,1,0),
                                    c(1,1,0,1,0),
                                    c(1,1,1,0,0),
                                    c(1,0,0,0,0)),nrow=5, ncol = 5)

church_adjacency_matrix <- matrix(c(c(0,rep(1,3),rep(0,8),rep(1,4),rep(0,30-16)), #column 1 - main area
                                    c(1,0,0,1,rep(0,12),rep(1,4),rep(0,30-20)), # column 2 - Hallway
                                    c(1,0,0,1,rep(0,4),rep(1,4),rep(0,30-12)), # column 3 - Hallway
                                    c(1,1,1,0,1,0,1,1,rep(0,30-8)), #column 4 Main room
                                    c(rep(0,3),1,0,1,0,1,rep(0,17),1,rep(0, 30-26)), # # column 5 Hallw
                                    c(rep(0,4),1,0,1,rep(0,15),rep(1,5),0,1,1), # Column 6 Hallway
                                    c(rep(0,3),1,0,1,0,1,rep(0,30-8)), # Column 7 Hallway
                                    c(rep(0,3),1,1,0,1,rep(0,21-8),1,1,rep(0,30-22)), # Column 8 Hallwa
```

```
                                    rep(c(0,0,1,rep(0,30-3)),4), # columns 9-12
                                    rep(c(1,rep(0,30-1)),4), # columns 13-16
                                    rep(c(0,1,rep(0,30-2)),4), # columns 17-20
                                    rep(c(rep(0,7),1,rep(0, 30-8)),2), # Columns 21 and 22
                                    rep(c(rep(0,5),1,rep(0,30-6)),3), # columns 23-25
                                    c(rep(0,4),1,1,rep(0,30-6)), # Column 26
                                    c(rep(0,5),1,rep(0,21),1,rep(0,30-28)), # column 27
                                    c(rep(0,26),1,rep(0,30-27)), # column 28
                                    c(rep(0,5),1,rep(0,30-6)), # column 29
                                    c(rep(0,5),1,1,rep(0,30-7))# column 30

),nrow=30, ncol = 30)

small_bld_3_rooms_graph<- graph_from_adjacency_matrix(small_bld_3_rooms, mode = "undirected")
#take a look
plot(small_bld_3_rooms_graph)
```




```
church_graph<- graph_from_adjacency_matrix(church_adjacency_matrix, mode = "undirected")
```

```
## Warning: The ‘adjmatrix‘ argument of ‘graph_from_adjacency_matrix()‘ must be symmetric
## with mode = "undirected" as of igraph 1.6.0.
## i Use mode = "max" to achieve the original behavior.
## This warning is displayed once every 8 hours.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.
```

```
#take a look
#plot(church_graph)
```

```
graph_to_use <- small_bld_3_rooms
adjacency_matrix_to_use <- small_bld_3_rooms
#N_rooms <- ncol(adjacency_matrix_to_use) #number of rooms
# Church_C<- c(200, #column 1 - main area / Hallway
#         100, # column 2 - Hallway
#         100, # column 3 - Hallway
#         400, #column 4 Main room
```

```
#        200
# )
small_bld_3_rooms_C <- c(100,200,300)
Church_C <- c(20, #column 1 - main area / Hallway
              5, # column 2 - Hallway
              5, # column 3 - Hallway
              362, #column 4 Main room
              5, # # column 5 Hallway
              7, # Column 6 Hallway - slightly bigger hallway
              5, # Column 7 Hallway
              5, # Column 8 Hallway
              50, # Column 9 Large classroom
              30, # column 10 classroom
              30, #column 11 classroom
              2, # column 12 janitors closet
              30, # column 13 classroom
              5, # column 14 - bathroom
              30, # column 15 classroom
              5, # column 16 - bathroom
              30, # column 17 classroom
              2, # column 18 - closet
              30, # column 19 classroom
              50, # column 20 large classroom
              2, # column 21 dressing room
              2, # column 22 dressing room
              4, # column 23 office
              4, # column 24 office
              4, # column 25 office
              4, # column 26 office
              5, # column 27 office - large
              2, # column 28 small bathroom
              4, # column 29 office
              4 # column 30 office
)

length(Church_C)
```

```
## [1] 30
```

Now that we have building defined by the adjacency matrix, we need to specify the conditions in the building

```
Max_Building_Capacity_small_bld <- sum(small_bld_3_rooms_C)
Prop_full <- 0.8 # how full do we want our building capacity to be
Adj_Max_Building_Capacity_small_bld <- round(Prop_full*Max_Building_Capacity_small_bld,0) #Adjusted cap
delt_small_bld <- Adj_Max_Building_Capacity_small_bld/community_pop # proportionality constant ( what p
```

Next, we need to put people in our building. These will be our initial conditions for the building level model.

```
day <- 31 #What day to extract results from the community level model
#N_rooms <- length(Church_C)
# delt is the proportionality constant
# C_x is the room capacity for each room
```

```r
# N_total is the total number of individuals that we want in the building
Bld_setup_func_v2 <- function(Community_output,day, delt,N_rooms, C_x, N_total){
  Building_ICs <- Community_output[day,] #Retrieves the number of S, I, and R individuals in the commun

  N_x  <- c(rep(0, N_rooms)) #initialize a vector
  for (i in 1:length(C_x)) {
    if (sum(N_x) >= N_total) break  # Stop if all individuals are distributed

    # Maximum we can place in this room, constrained by remaining individuals and room capacity
    max_for_room <- min(C_x[i], N_total - sum(N_x))

    # Randomly assign a number between 0 and max_for_room
    N_x[i] <- sample(0:max_for_room, 1)
  }

  # If after the first pass, there are still individuals left, distribute the remaining in a while loop
  while (sum(N_x) < N_total) {
    # Randomly select a room index
    room_index <- sample(1:length(N_x), 1)

    # Check if the selected room can accommodate more individuals
    if (N_x[room_index] < C_x[room_index] && sum(N_x) < N_total) {
      N_x[room_index] <- N_x[room_index] + 1
    }

    # Break if we've distributed all individuals or hit the building limit
    if (sum(N_x) >= N_total || sum(N_x) >= N_total) {
      break
    }
  }

  #Now we need to find how many S, I, and R individuals we should have in the building based on the pro
  Sb <- Community_output$S[day]*delt # number of susceptible in building
  Ib <- Community_output$I[day]*delt # number of Infectious in building
  Rb <- Community_output$R[day]*delt # number of Recovered in building

  # calculate the proportion of S,I, and R that should be in the building
  Sb_prop <- Sb/N_total
  Ib_prop <- Ib/N_total
  Rb_prop <- Rb/N_total

  # make the number of S, I, and R in each room is proportional to that in the building
  S_x <- N_x*Sb_prop
  I_x <- N_x*Ib_prop
  R_x <- N_x*Rb_prop

  S_x_prop <- S_x/N_total
  I_x_prop <- I_x/N_total
  R_x_prop <- R_x/N_total

  # we start with no particles in the building
  P_x <- c(rep(0,N_rooms))
```

```
  return(data.frame(S=S_x_prop,I=I_x_prop,R=R_x_prop, P=P_x, N_x = N_x))
}

Small_bld_setup <- Bld_setup_func_v2(Community_output = Community_output,day = day,delt = delt_small_blo
Small_bld_setup
```

```
##           S          I           R P N_x
## 1 0.1174288 0.01006380 0.001674100 0  62
## 2 0.2443276 0.02093919 0.003483207 0 129
## 3 0.5473696 0.04691029 0.007803464 0 289
```

```
r1_pop <-(Small_bld_setup$S[1]+Small_bld_setup$I[1]+Small_bld_setup$R[1])*Adj_Max_Building_Capacity_smal
r2_pop <-(Small_bld_setup$S[2]+Small_bld_setup$I[2]+Small_bld_setup$R[2])*Adj_Max_Building_Capacity_smal
r3_pop<- (Small_bld_setup$S[3]+Small_bld_setup$I[3]+Small_bld_setup$R[3])*Adj_Max_Building_Capacity_smal
r1_pop+r2_pop+r3_pop
```

```
## [1] 480
```

Checking initial conditions Now lets check that our building setup function did what I wanted it to. Each room's proportion should sum to 1. And if we convert back to numbers using N_x the numbers should equal

```
Small_bld_setup %>% summarise(total_prop = sum(S)+sum(I)+sum(R),build_pop = sum(N_x))
```

```
##   total_prop build_pop
## 1          1       480
```

props sum to 1 so that is good

```
#prop of susceptible in building
sum(Small_bld_setup$S)
```

```
## [1] 0.909126
```

```
#prop of susceptible in community
Community_output$S[day]/(Community_output$S[day]+Community_output$I[day]+Community_output$R[day])
```

```
## [1] 0.909126
```

```
#prop of infectious in building
sum(Small_bld_setup$I)
```

```
## [1] 0.07791328
```

```
#prop of infectious in community
Community_output$I[day]/(Community_output$S[day]+Community_output$I[day]+Community_output$R[day])
```

```
## [1] 0.07791328
```

```r
#prop of recovered in building
sum(Small_bld_setup$R)
```

```
## [1] 0.01296077
```

```r
#prop of susceptible in community
Community_output$R[day]/(Community_output$S[day]+Community_output$I[day]+Community_output$R[day])
```

```
## [1] 0.01296077
```

Okay, great all of that checks out

Now people and particles need to move (Transition matrices)

```r
Create_T_Matrix <-function(adjacency_matrix_to_use, N_rooms){
  set.seed(123145) # <- easier for debugging
  T_mov <- data.frame(matrix(runif(N_rooms^2), nrow = N_rooms)) #populates a square matrix/dataframe wi
  diag(T_mov) <- 0 #set diagonal to 0
  T_mov <- adjacency_matrix_to_use*T_mov #restrict the movement according to our network/adjacency matr
  T_mov_norm <- t(apply(T_mov, 1, function(x) x / sum(x))) # normalize so that there aren't more people
  T_mov <-T_mov_norm
  return(T_mov)
}

#Church_T_mov <- Create_T_Matrix(adjacency_matrix_to_use = church_adjacency_matrix, N_rooms = length())
Small_bld_T_mov <- Create_T_Matrix(adjacency_matrix_to_use = small_bld_3_rooms,N_rooms = length(small_bl

sum(Small_bld_T_mov[1,])
```

```
## [1] 1
```

```r
sum(Small_bld_T_mov[2,])
```

```
## [1] 1
```

```r
sum(Small_bld_T_mov[3,])
```

```
## [1] 1
```

```r
# sum(Church_T_mov[4,])
# sum(Church_T_mov[5,])
r1_in <- sum(Small_bld_T_mov[,1])
r1_out <- sum(Small_bld_T_mov[1,])
r1_change <- r1_in-r1_out
r2_in <- sum(Small_bld_T_mov[,2])
r2_out <- sum(Small_bld_T_mov[2,])
r2_change <- r2_in-r2_out
r3_in <- sum(Small_bld_T_mov[,3])
r3_out <- sum(Small_bld_T_mov[3,])
r3_change <- r3_in - r3_out
# We can use the same function for the particle matrix
#Church_theta_mov <- Create_T_Matrix(adjacency_matrix_to_use = church_adjacency_matrix)
Small_bld_theta_mov <- Create_T_Matrix(adjacency_matrix_to_use = small_bld_3_rooms,N_rooms = length(sma
```

Now set parameters for the model

```
parms <-data.frame(s=100,a=5, d=3,lam = 1)
# s = shedding, a = absorption, d = decay, lam = scalar for room capacities
#N_b <- Adj_Max_Building_Capacity #Building population size
Maxtime <- 24*3
times <- seq(from = 0, to = Maxtime, by = 0.2)
m <- 5 # number of equations per room
```

For the model we are going to have to sum over some of the vectors/matrices so we will put those steps in functions. This is needed for the movement of individuals and particles. Particles and people will have to have a different function because there are capacities on rooms for people but not particles.

```
flux_in_people <- function(N_rooms, Transition_matrix,State,Room_pops,Carrying_capacity,t){
  # Transition_matrix <- matrix(c(0,0.5,1,0.7,0,0,0.3,0.5,0), nrow = 3,ncol = 3)
  # State <- Church_setup$S_prop
  # Room_pops <- Church_setup$N_x
  # Carrying_capacity <- Church_C
  all_room_change <- c(seq(N_rooms))
  for(x in 1:N_rooms){ #flow in to room x from other rooms (j)
    flux_in_temp <- 0
    for(j in 1:N_rooms){
      flux_in_temp <- flux_in_temp + State[j]*Transition_matrix[j,x]*(1-(Room_pops[x]/Carrying_capacity
    }
    all_room_change[x] <- flux_in_temp
  }

  test <- c(M = as.vector(all_room_change))
  return(test)
}

flux_out_people <- function(N_rooms, Transition_matrix,State,Room_pops,Carrying_capacity){
  all_room_change <- c(seq(N_rooms))
  for(x in 1:N_rooms){
    flux_out_temp <- 0
    for(j in 1:N_rooms){
      flux_out_temp <- flux_out_temp + State[x]*Transition_matrix[x,j]*(1-(Room_pops[j]/Carrying_capaci
    }
    all_room_change[x] <- flux_out_temp
  }
  test <- c(M = as.vector(all_room_change))
  return(test)
}

flux_in_particles <- function(N_rooms, Transition_matrix,State){
  all_room_change <- c(seq(N_rooms))
  for(x in 1:N_rooms){
    flux_in_temp <- 0
    for(j in 1:N_rooms){
      flux_in_temp <- flux_in_temp + State[j]*Transition_matrix[j,x]
    }
    all_room_change[x] <- flux_in_temp
  }
```

```
  test <- c(M = as.vector(all_room_change))
  return(test)
}

flux_out_particles <- function(N_rooms, Transition_matrix,State){
  all_room_change <- c(seq(N_rooms))
  for(x in 1:N_rooms){
    flux_out_temp <- 0
    for(i in 1:N_rooms){
      flux_out_temp <- flux_out_temp + State[x]*Transition_matrix[x,i]
    }
    all_room_change[x] <- flux_out_temp
  }
  test <- c(M = as.vector(all_room_change))
  return(test)
}
```

Function for the whole model

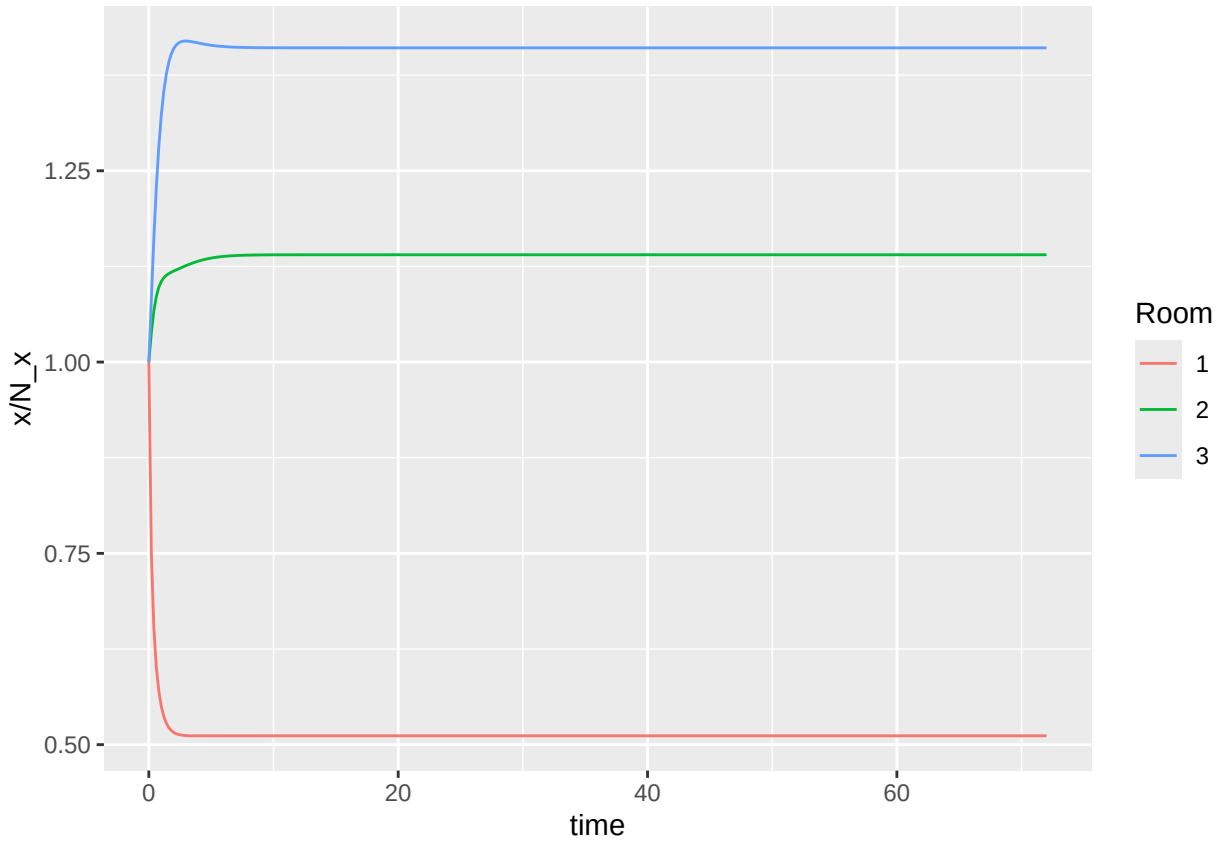
Checking what the model should do based on the initial conditions and stepping through manually

```
Particle_model_v2 <- function(t, x, parms,T_mov, theta_mov, adjacency_matrix_to_use,C_x,N_b,N_rooms){

  # x <- Church_Init_conds
  # T_mov <- Church_T_mov
  #
  # theta_mov <- Church_theta_mov
  # adjacency_matrix_to_use <- small_bld_3_rooms
  # C_x <- small_bld_3_rooms_C
  # N_b <- N_b
  N_rooms <- N_rooms
  ncompartment <- 5
  n_rooms <- length(x)/ncompartment
  S <- as.matrix(x[1:n_rooms])
  I <- as.matrix(x[(n_rooms+1):(2*n_rooms)])
  R <- as.matrix(x[(2*n_rooms+1):(3*n_rooms)])
  P <- as.matrix(x[(3*n_rooms+1):(4*n_rooms)])
  N_x <- as.matrix(x[(4*n_rooms+1):(5*n_rooms)])


  with(parms,{

    dS <- as.matrix((flux_in_people(N_rooms, Transition_matrix =T_mov, State=S,Room_pops = (S+I+R)*N_x,(
    # s1_in <- S[1]*T_mov[1,1]*(1-(((S[1]+I[1]+R[1])*N_b)/C_x[1]))
    # s2_in <- S[2]*T_mov[2,1]*(1-(((S[1]+I[1]+R[1])*N_b)/C_x[1]))
    # s3_in <- S[3]*T_mov[3,1]*(1-(((S[1]+I[1]+R[1])*N_b)/C_x[1]))
    #
    # s1_out <- S[1]*T_mov[1,1]*(1-(((S[1]+I[1]+R[1])*N_b)/C_x[1]))
    # s2_out <- S[1]*T_mov[1,2]*(1-(((S[2]+I[2]+R[2])*N_b)/C_x[2]))
    # s3_out <- S[1]*T_mov[1,3]*(1-(((S[3]+I[3]+R[3])*N_b)/C_x[3]))
    #
    # Total_Sx <- s1_in+s2_in+s3_in - (s1_out+s2_out+s3_out)
    dI <- as.matrix((flux_in_people(N_rooms=N_rooms, Transition_matrix =T_mov, State=I,Room_pops = (S+I-
```

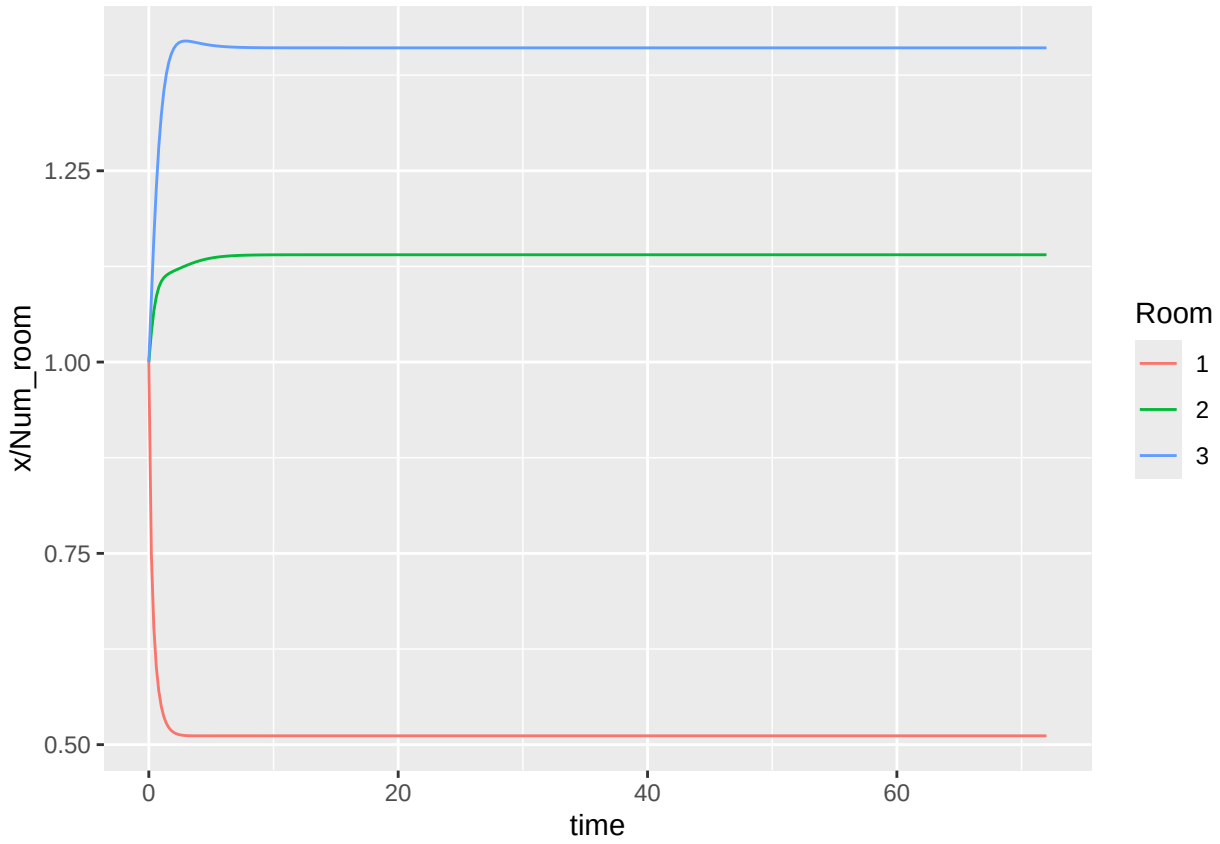
```
    dR <- as.matrix((flux_in_people(N_rooms=N_rooms, Transition_matrix =T_mov, State=R,Room_pops = (S+I+
    #last step will be the particle EQ

    dP <- s*as.matrix(I)*as.matrix((S+I+R)*N_x) - as.matrix(a*P/(lam*C_x*((S+I+R)*N_x)))-d*as.matrix(P)

    dN_x <- as.matrix((flux_in_people(N_rooms=N_rooms, Transition_matrix =T_mov, State=N_x,Room_pops = N
    #
    dt <- c(dS,dI,dR,dP,dN_x)

    return(list(dt))})

}
```

```
#Church_Init_conds_v1 <-c(S=Church_setup$S, I = Church_setup$I, R = Church_setup$R, P = Church_setup$P)

Small_bld_Init_conds_v2 <-c(S=Small_bld_setup$S, I = Small_bld_setup$I, R = Small_bld_setup$R, P = Small

# Church_output_v1 <- data.frame(lsoda(y = Church_Init_conds_v1, func = Particle_model_v1,times = times
#                                      parms = parms,
#                                      adjacency_matrix_to_use=church_adjacency_matrix,
#                                      theta_mov =Church_theta_mov,
#                                      T_mov = Church_T_mov,
#                                      C_x=Church_C,N_b = N_b))
Small_bld_output_v2 <- data.frame(lsoda(y = Small_bld_Init_conds_v2, func = Particle_model_v2,times = ti
                                     parms = parms,
                                     adjacency_matrix_to_use=small_bld_3_rooms,
                                     theta_mov =Small_bld_theta_mov,
                                     T_mov = Small_bld_T_mov,
                                     C_x=small_bld_3_rooms_C,
                                     N_b = Adj_Max_Building_Capacity_small_bld,
                                     N_rooms = length(small_bld_3_rooms_C)))
```

```
data_clean <- Small_bld_output_v2%>% pivot_longer(cols = !time,
                                                  names_to = c("State", "Room"),
                                                  names_pattern = "([A-Za-z]+)(\\d+)",
                                                  values_to = "Number")
data_ratios <- data_clean %>% pivot_wider(names_from = c(State),values_from = c(Number)) %>% group_by(ti
  mutate(N_x = (S+I+R)*Adj_Max_Building_Capacity_small_bld, ratio = x/N_x) #%>% group_by(time) %>% summ
# %>% group_by(time,Room) %>%
#   mutate(K_x = ((parms$s)*I)/((parms$a)*N_x), prop_to_K = P/K_x)
# Church_P_x_K_x_plot<-ggplot(Church_data_ratios,aes(x =time, y =prop_to_K,group= Room,color=Room))+geo
ggplot(data_ratios,aes(x=time, color = Room))+geom_line(aes(x = time, y = x/N_x))#+geom_line(aes(x = ti
```




```
data_clean %>% pivot_wider(names_from = c(State),values_from = c(Number)) %>%
   mutate(Prop_room = S+I+R, Num_room =Prop_room*Adj_Max_Building_Capacity_small_bld) %>%
   ggplot(aes(x = time,y = x/Num_room,color = Room ))+geom_line()
```




```
p1_v2 <- data_clean %>% filter(State == "S") %>%
  group_by(State,Room)%>%
  ggplot( aes(x=time, y = Number, group =Room, color = Room))+
  geom_line()+theme_classic()+
  labs(x = "Time (hours)", y= "Proportion of Susceptible individuals")+ggtitle("Proportion of Susceptibl

p2_v2 <- data_clean %>% filter(State == "I") %>%
  group_by(State,Room)%>%
  ggplot( aes(x=time, y = Number, group =Room, color = Room))+
  geom_line()+theme_classic()+
  labs(x = "Time (hours)", y= "Proportion of Infectious individuals")+ggtitle("Proportion of Infectious
p3_v2 <- data_clean %>% filter(State == "R") %>%
  group_by(State,Room)%>%
  ggplot( aes(x=time, y = Number, group =Room, color = Room))+
  geom_line()+theme_classic()+
  labs(x = "Time (hours)", y= "Proportion of Recovered individuals")+ggtitle("Proportion of Recovered in
p4_v2 <- data_clean %>% filter(State == "P") %>%
  group_by(State,Room)%>%
  ggplot( aes(x=time, y = Number, group =Room, color = Room))+
  geom_line()+theme_classic()+
  labs(x = "Time (hours)", y= "Number of infectious particles")+ggtitle("Infectious particles in rooms w
p5_v2 <- data_clean %>% filter(State == "x") %>%
  group_by(State,Room)%>%
  ggplot( aes(x=time, y = Number, group =Room, color = Room))+
  geom_line()+theme_classic()+
  labs(x = "Time (hours)", y= "Number of individuals")+ggtitle("Number of individuals in rooms within a
```

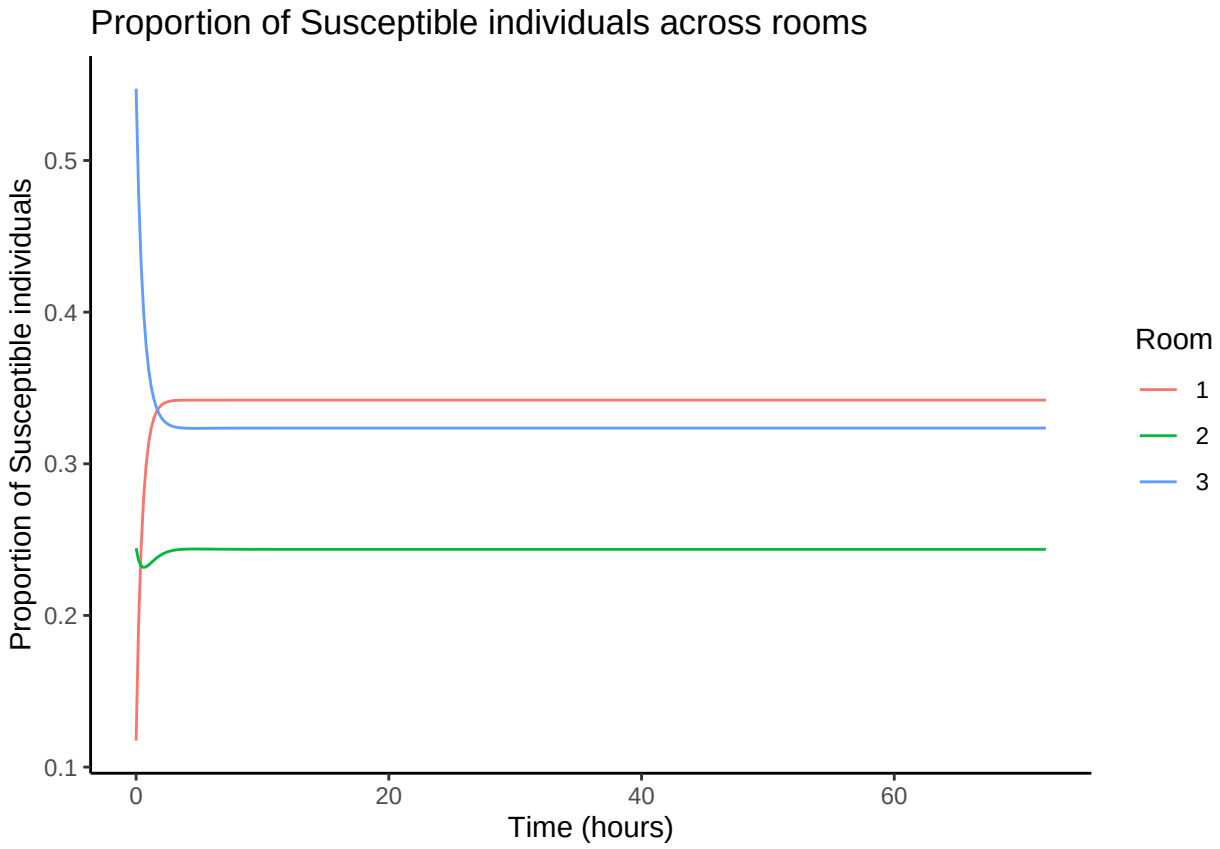
```
# Church_data_clean %>% filter(State != "P") %>%
#   group_by(time,Room)%>% summarise(total_room_pop=sum(Number)) %>% group_by(time) %>% summarise(tot_b
#   ggplot( aes(x=time, y = tot_bld_pop))+
#   geom_line()+theme_classic()+
#   labs(x = "Time (hours)", y= "Number of people")+ggtitle("number of people in building")

p6_v2 <- data_clean %>% filter(State == "S"| State == "I" | State == "R") %>% ungroup() %>% group_by(tir
  ggplot(aes(x=time, y=Total_prop, color=Room))+geom_line()+
  labs(x = "Time (hours)", y= "Proportion of individuals")+ggtitle("Proportion of individuals in rooms
```

```
## `summarise()` has grouped output by 'time'. You can override using the
## `.groups` argument.
```

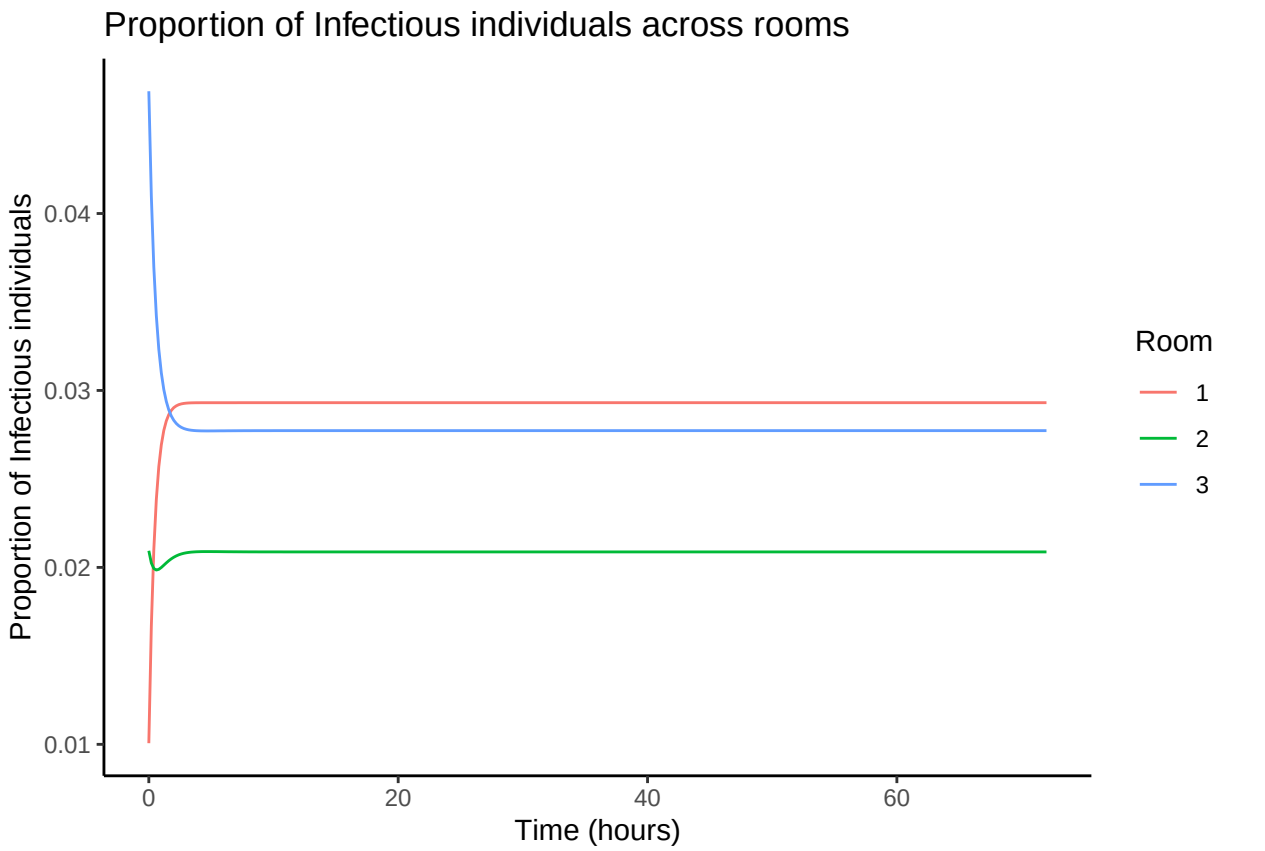
```
p7_v2 <- data_clean %>% filter(State == "S"| State == "I" | State == "R") %>% ungroup() %>% group_by(tir
  ggplot(aes(x=time, y=Total_prop))+geom_line()+
  labs(x = "Time (hours)", y= "Proportion of individuals")+ggtitle("Proportion of individuals in the bui


p8_v2 <- data_clean %>% filter(State == "x") %>% ungroup() %>% group_by(time, Room) %>% summarise(Total_
  ggplot(aes(x=time, y=Total_prop, color=Room))+geom_line()+
  labs(x = "Time (hours)", y= "Number of individuals")+ggtitle("Number of individuals in the building")
```

```
## `summarise()` has grouped output by 'time'. You can override using the
## `.groups` argument.
```

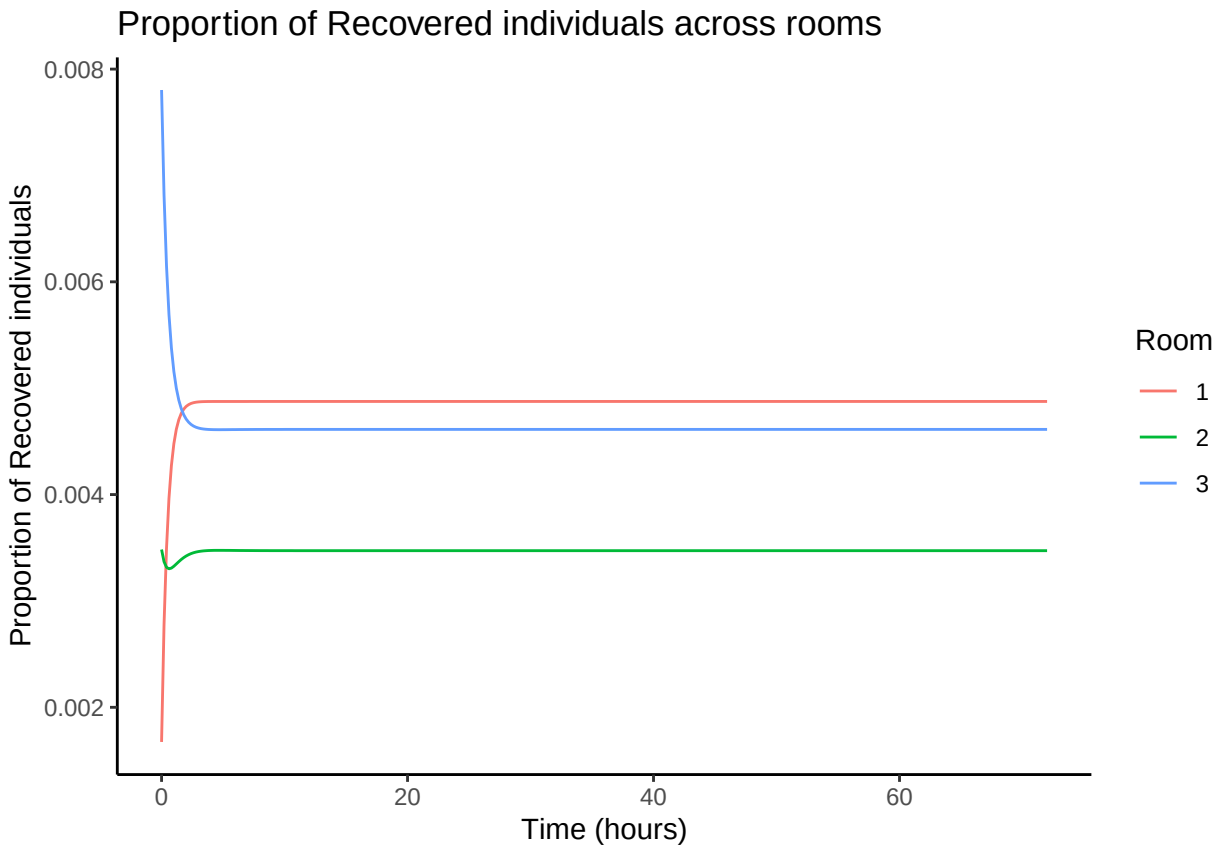
```
p1_v2
```



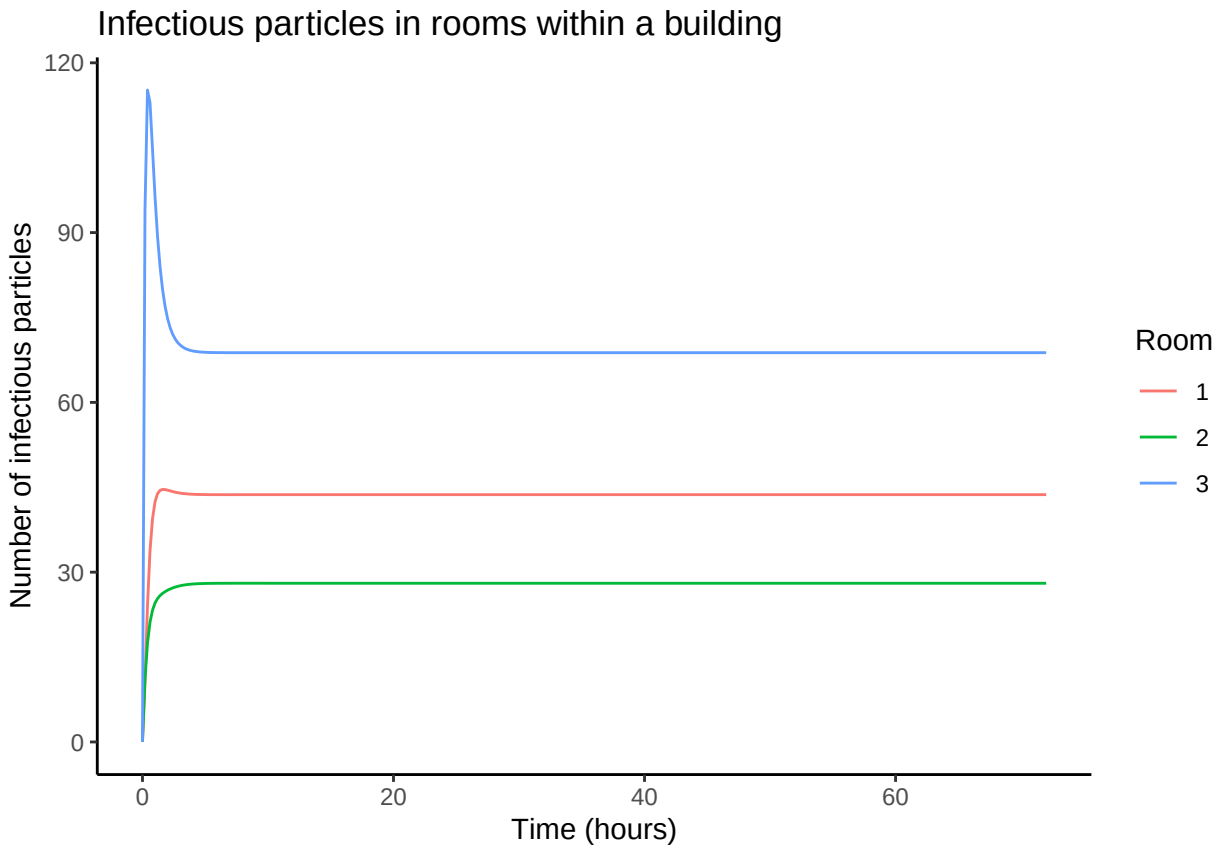
```
p2_v2
```


Proportion of Infectious individuals across rooms
Proportion of Infectious individuals
0.04
0.03
0.02
0.01
0
20
40
60
Time (hours)
Room
1
2
3

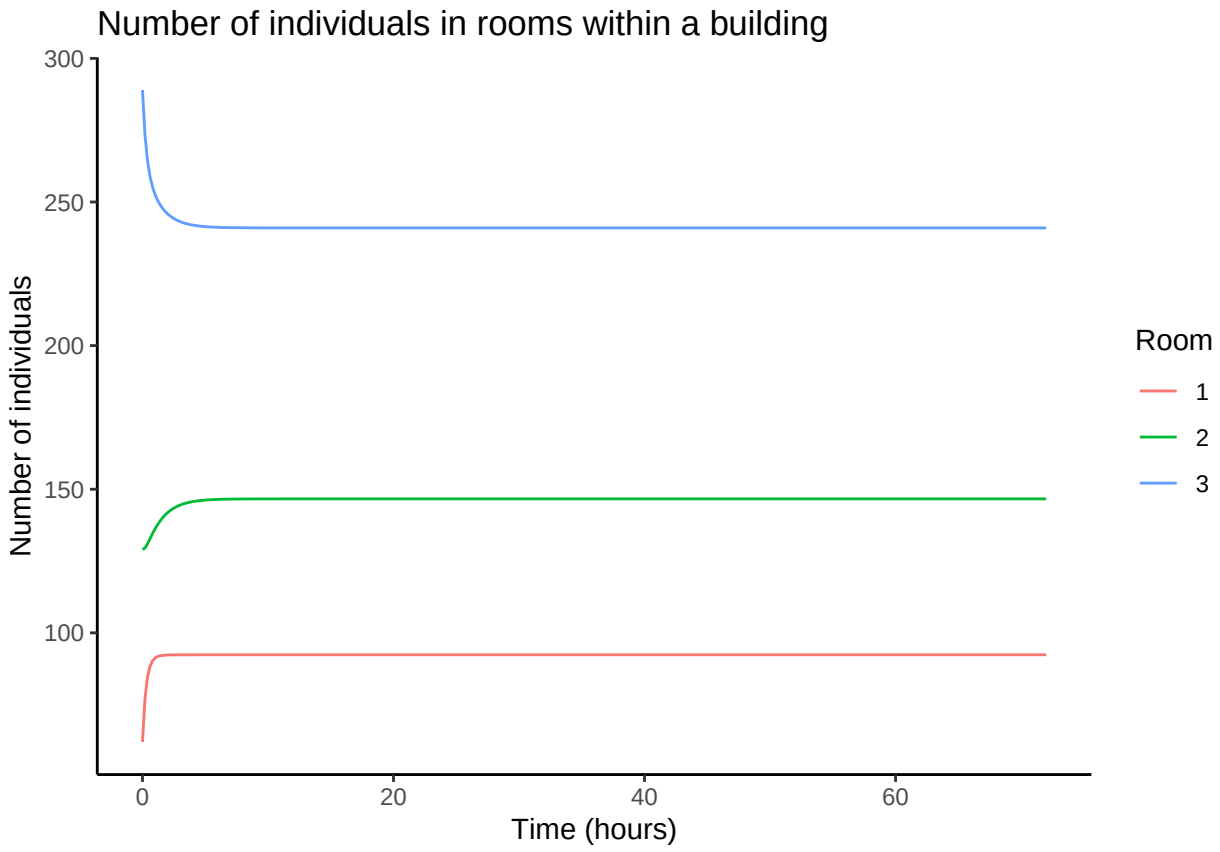
```
p3_v2
```


Proportion of Recovered individuals across rooms
Proportion of Recovered individuals
0.008
0.006
0.004
0.002
0
20
40
60
Time (hours)
Room
1
2
3

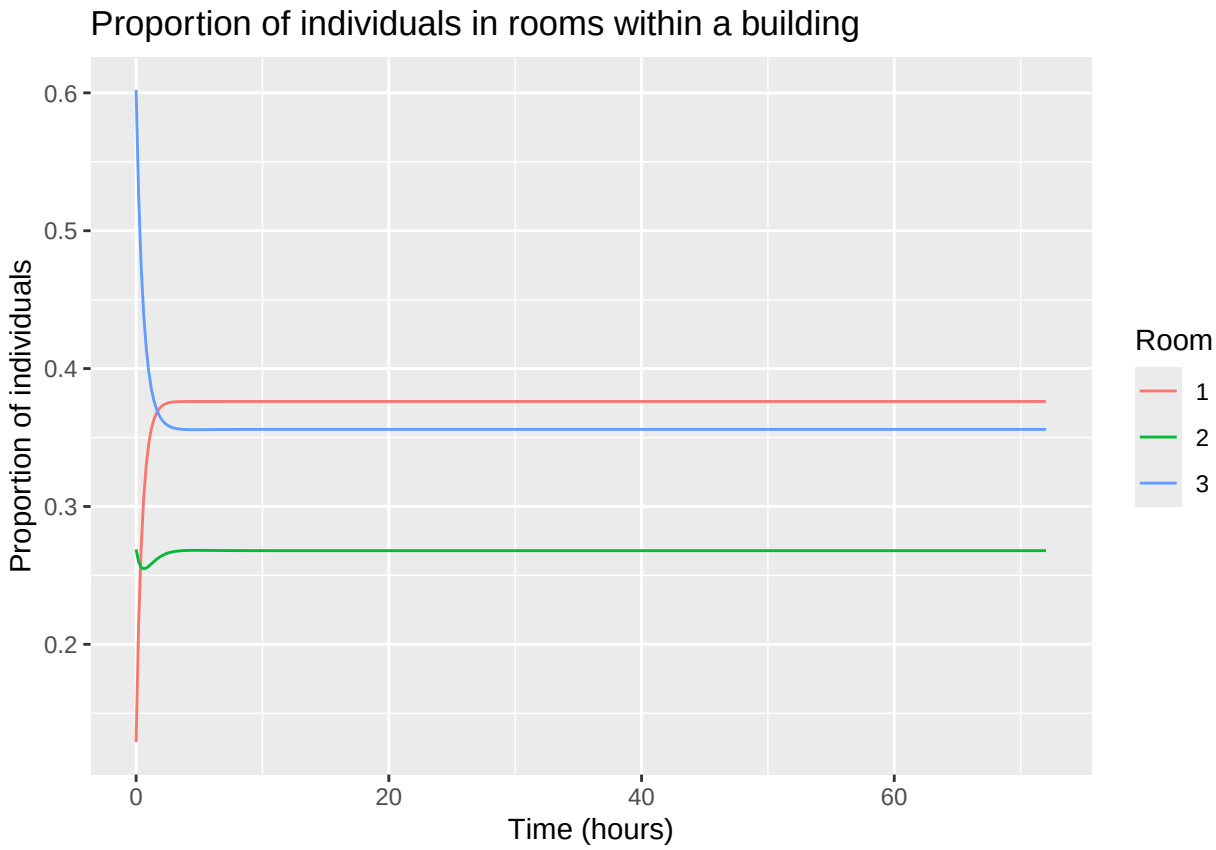
```
p4_v2
```


Infectious particles in rooms within a building
120
90
60
30
0
Number of infectious particles
0
20
40
60
Time (hours)
Room
1
2
3

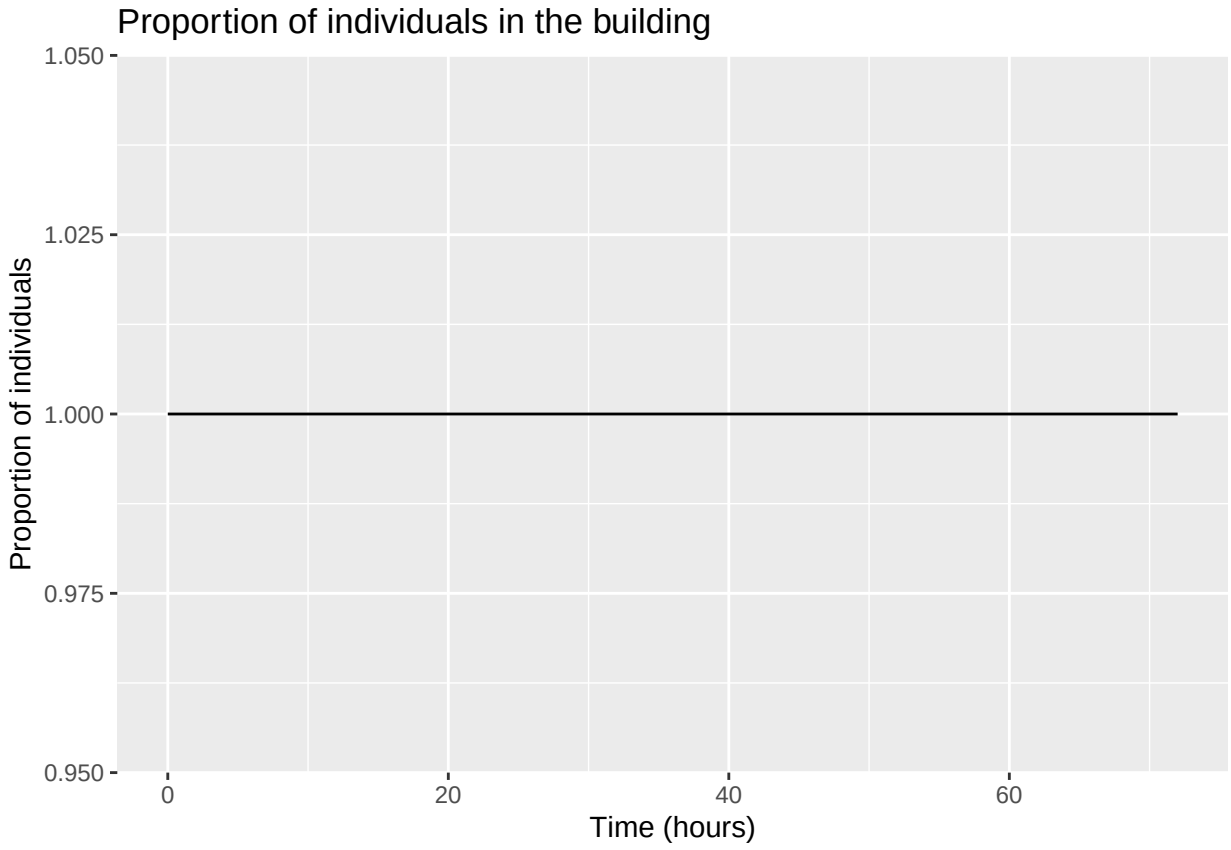
```
p5_v2
```


Number of individuals in rooms within a building
300
250
200
150
100
Number of individuals
0
20
40
60
Time (hours)
Room
1
2
3

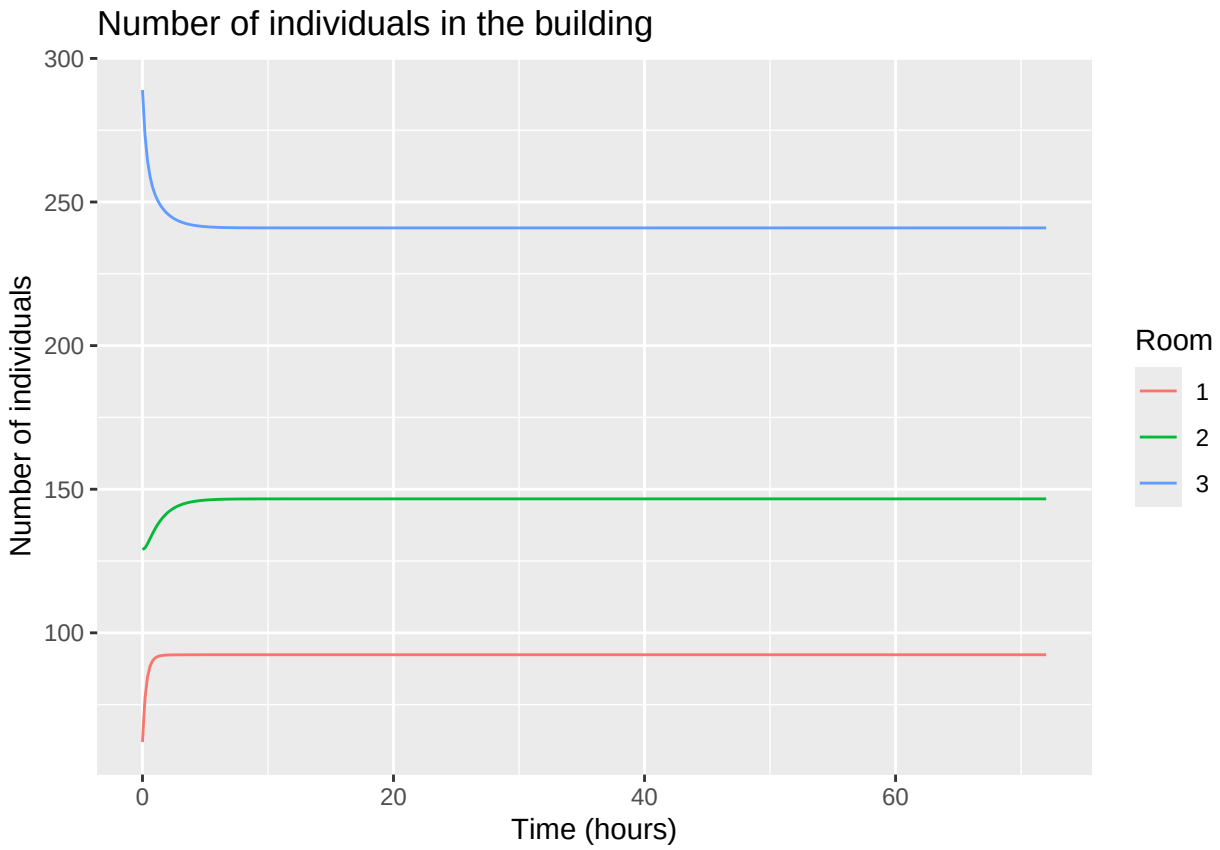
```
p6_v2
```


Proportion of individuals in rooms within a building
Proportion of individuals
0.6
0.5
0.4
0.3
0.2
0
20
40
60
Time (hours)
Room
1
2
3

```
p7_v2
```


Proportion of individuals in the building
1.050
1.025
1.000
0.975
0.950
Proportion of individuals
0
20
40
60
Time (hours)

```
p8_v2
```


Number of individuals in the building
300
250
200
150
100
Number of individuals
0
20
40
60
Time (hours)
Room
1
2
3