

Oikos

OIK-07148

Evans, J. C., Silk, M. J., Boogert, N. J. and Hodgson, D. J. 2020. Infected or informed? Social structure and the simultaneous transmission of information and infectious disease. – Oikos doi: [10.1111/oik.07148](https://doi.org/10.1111/oik.07148)

Appendix 1

Table A1. Summary of examples of theoretical models of infectious disease or information transmission through a social network.

Susceptible - Infected (SI)	States:	<i>Susceptible</i>	<i>Infected</i>		
	Disease description	Can be infected	Has disease and is contagious		
	Information description	Can receive info	Has accepted info and is spreading it		
Susceptible – Infected – Recovered/Refractor (SIR)	States:	Susceptible	Infected	Recovered/Refractory/Vaccinated	
	Disease description	Can be infected	Has disease and is contagious	Becomes immune/dies – is no longer contagious Is Vaccinated and therefore immune	
	Information description	Can receive info	Has received info and is using it to change its behaviour	Has lost interest in sharing/ has forgotten/no longer uses info to change behaviour – no longer involved in spreading info	
Susceptible – Infected – Susceptible (SIS)	States:	<i>Susceptible</i>	<i>Infected</i>	<i>Susceptible</i>	<i>Recovered/Refractory/Vaccinated</i>
	Disease description	Can be infected	Has disease and is contagious	Recovers and returns to being Susceptible	Becomes immune/dies – is no longer contagious Is Vaccinated and therefore immune
	Information description	Can receive info	Has received info and is using it to change its behaviour	Has forgotten/no longer uses info – returns to being Susceptible	Has forgotten and/or no longer pays attention to or does not use info.
Susceptible – Exposed – Infected - Recovered (SEIR)	States:	<i>Susceptible</i>	<i>Exposed/Contacted</i>	<i>Infected</i>	<i>Recovered/Refractory</i>
	Disease description	Can be infected	Has disease but is not contagious	Has disease and is contagious	Is immune/ dead – is no longer contagious
	Information description	Can receive info	Has received info but is not yet using it or has stopped using it	Has received info and is using it	Has forgotten and/or no longer pays attention to or does not use info.
Susceptible – Exposed – Infected - Sceptic (SEIZ)	States:	<i>Susceptible</i>	<i>Exposed</i>	<i>Infected</i>	<i>Sceptic</i>
	Information description	Can receive info	Has received info but is not yet spreading	Has accepted info and is spreading it	Refuses to accept info – spreads Scepticism about that info
Susceptible – Aware – Uninformed – Informed – Recovered/Susceptible (SAUIR/S)	States:	<i>Susceptible/Aware</i>	<i>Uninformed</i>	<i>Informed</i>	<i>Sceptic/Recovered</i>
	Disease description	Can be infected Can become Aware that neighbours are infected – allowing avoidance behaviour – reducing infection risk	Has disease but does not know it	Has disease and knows it	Recovers and returns to Susceptible state (S) Individual recovers or dies – no longer contagious (R)

R code for simulation from Fig. 1c

```
set.seed(1)

library(igraph)

#-----

##Define a Bernoulli draw function for convenience

rbern<-function(n,prob){
  return(rbinom(n,1,prob))
}

#-----

###Transmission functions

##Cascade

ts<-function(network,indiv.info,S_I,I_R,plot=T){

  S<-indiv.info$S
  I<-indiv.info$I
  R<-indiv.info$R

  t.mat<-array(0,dim=dim(network))
  for(i in 1:nrow(network)){
    for(j in 1:nrow(network)){
      t.mat[i,j]<-rbern(1,S_I)*network[i,j]
    }
  }

  diag(t.mat)<-0

  danger<-t.mat[which(I>0),]
  ifelse(is.vector(danger)==TRUE,infected<-which(danger>0),infected<-
  which(colSums(danger)>0))

  if(length(infected)>0){
    for(i in 1:length(infected)){
      if(R[infected[i]]==0){
        I[infected[i]]<-1
        S[infected[i]]<-0
      }
    }
  }

  for(i in 1:nrow(indiv.info)){
    if(I[i]==1){
      R[i]<-rbern(1,I_R)
      if(R[i]==1){
        I[i]<-0
      }
    }
  }

  indiv.info2<-indiv.info
  indiv.info2$S<-S
  indiv.info2$I<-I
  indiv.info2$R<-R

  res<-list(indiv.info2)
  return(res)

} #end function
```

```

##Threshold

tts<-function(network, indiv.info, S_I_min, S_I_max, S_I_slope, S_I_pivot, I_R, plot=T) {

  S<-indiv.info$S
  I<-indiv.info$I
  R<-indiv.info$R

  ##calculate the proportion of infected direct neighbours an individual has
  inf.net<-I*network
  inf.deg<-colSums(inf.net)
  deg<-colSums(network)
  prop.inf<-inf.deg/deg

  p.inf<-S_I_min+(S_I_max-S_I_min)*1/(1+exp(-S_I_slope*(prop.inf-S_I_pivot)))
  inf<-rep(NA, length(p.inf))
  for(i in 1:length(p.inf)){
    inf[i]<-rbern(1,p.inf[i])
  }

  for(i in 1:length(inf)){
    if(inf[i]==1&R[i]==0&S[i]==1){
      I[i]<-1
      S[i]<-0
    }
  }

  for(i in 1:nrow(indiv.info)){
    if(I[i]==1){
      R[i]<-rbern(1,I_R)
      if(R[i]==1){
        I[i]<-0
      }
    }
  }

  indiv.info2<-indiv.info
  indiv.info2$S<-S
  indiv.info2$I<-I
  indiv.info2$R<-R

  res<-list(indiv.info2)
  return(res)

} #end function

##-----
##-----

net<-erdos.renyi.game(100,0.1,directed=FALSE)
plot(net)

mat<-as_adjacency_matrix(net,sparse=FALSE)

id<-seq(1,100,1)
S<-rep(1,100)
I<-rep(0,100)
R<-rep(0,100)

indiv.info<-data.frame(id,S,I,R)

#Cascade params
S_I<-0.008
I_R<-0

#Threshold params
S_I_min<-0.002

```

```

S_I_max<-0.3
S_I_slope<-15
S_I_pivot<-0.5

#check shape of threshold model generated
x<-seq(0,1,0.001)
y<-S_I_min+(S_I_max-S_I_min)*1/(1+exp(-S_I_slope*(x-S_I_pivot)))
dev.new()
plot(x,y,type="l")

##-----
##-----

##Simulations
###BOTH COMPARED FOR MODULAR NETWORKS

#Modular

pop.info<-list()

pop.info[[1]]<-indiv.info
pop.info[[1]]$I[1:3]<-rep(1,3)
pop.info[[1]]$S[1:3]<-rep(0,3)

network<-mat
Cres4=as.list(rep(NA,20))
for(rep in 1:20){

  tmp.up<-ts(network=network,indiv.info=pop.info[[1]],S_I=S_I,I_R=I_R,plot=F)

  pop.info[[2]]<-tmp.up[[1]]
  network<-network

  for(t in 3:200){
    tmp.up<-ts(network=network,indiv.info=tmp.up[[1]],S_I=S_I,I_R=I_R,plot=F)
    pop.info[[t]]<-tmp.up[[1]]
  }

  Cres4[[rep]]<-matrix(unlist(lapply(pop.info,colSums)),nr=200,nc=4,byrow=TRUE)

}
#----
#Modular

pop.info<-list()

pop.info[[1]]<-indiv.info
pop.info[[1]]$I[1:3]<-rep(1,3)
pop.info[[1]]$S[1:3]<-rep(0,3)

network<-mat

Cres3=as.list(rep(NA,20))
for(rep in 1:20){

  tmp.up<-
  tts(network=network,indiv.info=pop.info[[1]],S_I_min=S_I_min,S_I_max=S_I_max,S_I_sl
  ope=S_I_slope,S_I_pivot=S_I_pivot,I_R=I_R,plot=F)

  pop.info[[2]]<-tmp.up[[1]]
  network<-network

  for(t in 3:200){

```

```
tmp.up<-
tts(network=network, indiv.info=tmp.up[[1]], S_I_min=S_I_min, S_I_max=S_I_max, S_I_slope=S_I_slope, S_I_pivot=S_I_pivot, I_R=I_R, plot=F)
  pop.info[[t]]<-tmp.up[[1]]
}

Cres3[[rep]]<-matrix(unlist(lapply(pop.info, colSums)), nr=200, nc=4, byrow=TRUE)
}
```