# INA smartsurv

K. A. Garrett, University of Florida

## Identifying key sampling nodes, as part of an impact network analysis

Impact network analysis (INA) is designed to address multiple aspects of linked socioeconomic networks (spread of ideas, money, influence, etc.) and biophysical networks (spread of new varieties, certified seed, pathogens, pests, etc.)

Garrett, K. A., R. I. Alcala-Briseno, K. F. Andersen, C. E. Buddenhagen, R. A. Choudhury, J. C. Fulton, J. F. Hernandez Nopsa, R. Poudel, and Y. Xing. 2018. Network analysis: A systems framework to address grand challenges in plant pathology. Annual Review of Phytopathology 56: 559-580.

The example given in ths vignette addresses identifying the best locations in a network for sampling to quickly detect a spreading species: for example, a pathogen or new plant genotype. A node is better for sampling if the pathogen is likely to be detected at that node before the pathogen has spread very far through the network. For example, an isolated node would be a poor sampling choice because the pathogen would likely spread widely before it reaches the isolated node.

Examples of this type of analysis appear in the following papers.

Andersen, K. F., C. E. Buddenhagen, P. Rachkara, R. Gibson, S. Kalule, D. Phillips, and K. A. Garrett. 2019. Modeling epidemics in seed systems and landscapes to guide management strategies: The case of sweetpotato in Northern Uganda. Phytopathology 109:1519-1532.

Buddenhagen, C. E., J. F. Hernandez Nopsa, K. F. Andersen, J. Andrade-Piedra, G. A. Forbes, P. Kromann, S. Thomas-Sharma, P. Useche, and K. A. Garrett. 2017. Epidemic network analysis for mitigation of invasive pathogens in seed systems: Potato in Ecuador. Phytopathology 107:1209-1218.

In the simplest version of the analysis, a pathogen is equally likely to enter the epidemic network at any node, and then moves through the network. In this version of the analysis, it enters the network at only one node (the "introduction node"). Each node in the network (each potential "sampling node") is then evaluated to determine how early the pathogen can be detected, in terms of how many nodes remain uninfected when the pathogen's presence in the network is detected at the sampling node. A summary of the performance of a node indicates how early the pathogen is detected at that node, after entering the network at each node in turn. That is, if there are N nodes, there would be N potential introduction nodes for the pathogen and N analyses of how early the pathogen could be detected at the sampling node being evaluated.

An analysis like this, focused on the biophysical network, may be useful even when information is limited about the socioeconomic network. Or information about the socioeconomic network may be incorporated in the form of weights indicating the likelihood that a node is the point of introduction of the species in the network. Nodes associated with less-informed managers may be at higher risk of being the introduction node for an invasive species. Or environmental conditions might make initial establishment of a pathogen more likely at some nodes than others. In this weighted analysis, the performance of a sampling node would be evaluated taking the weights (likelihood of introduction) into account, so that overall performance would be weighted toward the results for more likely introduction nodes.

This vignette starts from the simplest versions of analyses and builds up to a more complete analysis. If you would prefer to jump to a full analysis, you could skip to the description of the function smartsurv, below.

## Getting the INApreliminary package from GitHub

An updated version of INApreliminary is available on GitHub.

Note that if you do not already have the package devtools, you will need to install it once (using the command commented out below)

Then you will need to install INApreliminary from GitHub (using the command commented out below)

```
# use the following three commented commands if you do not yet have
# INApreliminary or need to update the version

#install.packages("devtools") # use this command if you do not yet have devtools
#installed

#library(devtools)

#devtools::install_github("GarrettLab/INApreliminary")

library(INApreliminary)

library(igraph)
```

```
##
## Attaching package: 'igraph'

## The following objects are masked from 'package:stats':
##
##     decompose, spectrum

## The following object is masked from 'package:base':
##
##     union
```

```
library(viridis)
```

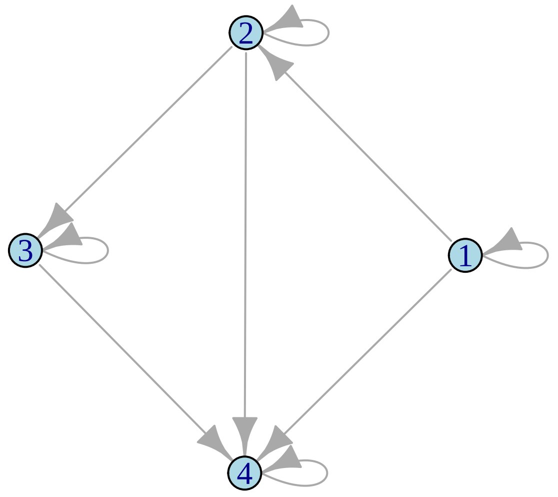
```
## Loading required package: viridisLite
```

## A simple example, illustrating the functions used by the main function smartsurv

Consider a simple adjacency matrix, which will be used to illustrate the functions.

Note that this function currently assumes that the diagonal of the adjacency matrix is composed of 1's, meaning that once a node is invaded, it stays invaded

```
Amat <- matrix(c(1,0,0,0,1,1,0,0,0,1,1,0,1,1,1,1),nrow=4,ncol=4)

#Make an igraph-formatted version of the adjacency matrix
Amati <- graph.adjacency(Amat)

#Create a layout for the matrix to make comparison of the structure easier in
#examples throughout this vignette
layout.ex <- layout.kamada.kawai(Amati)

plot(Amati, layout=layout.ex, vertex.color='lightblue')
```



```
# each potential link has probability 0.7 of existing in one realization
sAmat <- Amat * 0.7
```

### Illustration for a single starting node (onestart function)

For a given introduction node (only one at this point), onestart yields two outputs for one realization.

Output object outmat has rows=time steps, columns = nodes, and entries = invasion status for each node at each time step (1 = invaded, 0 = not invaded).

Output object sampnodes has rows = nodes, first column = time step at which invasion is detected (Inf if node is never reached), second column = the number of nodes invaded at the time of detection, third column = the number of nodes not invaded at the time of detection.

```
# in this case, the second node was selected as the starting point
onestart(adjmat=Amat, start.choice=2, stoch=F)
```

```
## $outmat
##   [,1] [,2] [,3] [,4]
## 1    0    1    0    0
## 2    0    1    1    1
## 3    0    1    1    1
##
## $sampnodes
##      firstt numinvt numnotinvt
## [1,]    Inf       3          1
## [2,]      1       1          3
## [3,]      2       3          1
## [4,]      2       3          1
```

```
onestart(adjmat=sAmat, start.choice=2, stoch=T) # stochastic version
```

```
## $outmat
##   [,1] [,2] [,3] [,4]
## 1    0    1    0    0
## 2    0    0    0    1
```

```
##
## $sampnodes
##      firstt numinvt numnotinvt
## [1,]    Inf       1          3
## [2,]      1       1          3
## [3,]    Inf       1          3
## [4,]      2       1          3
```

## The function multistart summarizes this analysis across all potential starting points.

The output is a matrix with rows = starting nodes (introduction nodes), columns = samping nodes, and entries = number of nodes not invaded by time detected at the sampling node having started at the introduction node.
For the stochastic case, the result is only based on one realization (function smartsurv, described below, evaluates multiple realizations).

Note that the diagonal contains threes, because when a node is both the starting point and the sampling point, the invasion will only have reached one node before detection and the remaining three remain free of the invasion.

```
multistart(adjmat=Amat, stoch=F)
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    3    1    0    1
## [2,]    1    3    1    1
## [3,]    2    2    3    2
## [4,]    3    3    3    3
```

```
multistart(adjmat=sAmat, stoch=T)
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    3    2    2    2
## [2,]    2    3    2    2
## [3,]    3    3    3    3
## [4,]    3    3    3    3
```

```
# For the deterministic version, let's plot the network again with the coloring
# of the nodes indicating how many nodes remain free of the invasion

# For this we take the mean across all the potential starting nodes (giving all
# the starting nodes equal weight, equal probability of being the introduction
# node, for now)

temp <- multistart(adjmat=Amat, stoch=F)
temp
```

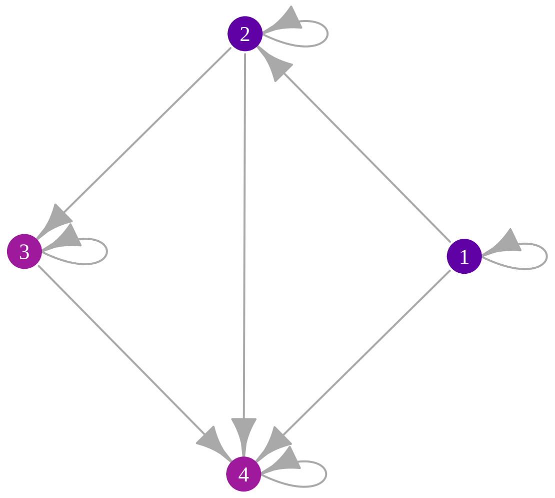
```
##      [,1] [,2] [,3] [,4]
## [1,]    3    1    0    1
## [2,]    1    3    1    1
## [3,]    2    2    3    2
## [4,]    3    3    3    3
```

```r
numinvfree <- colMeans(temp) # trait for coloring nodes
numinvfree
```

```
## [1] 2.25 2.25 1.75 1.75
```

```r
nnodes <- dim(Amat)[1] # the total number of nodes

pal1 <- viridis_pal(option="C", direction = -1, end = .8)

# create palette that incorporates the possible extremes in terms of number of
# nodes free from invasion (0 and nnodes-1) to make comparison across figures
# easier
nodecol = pal1(15)[as.numeric(cut(c(0, numinvfree, (nnodes - 1)), breaks = 15))]
nodecol <- nodecol[2:(nnodes + 1)]

plot(
  Amati,
  vertex.frame.color = nodecol,
  vertex.label.cex = 0.7,
  vertex.label.color = 'white',
  edge.curved = F,
  vertex.color = nodecol,
  layout = layout.ex,
  sub = 'Darker colors indicate better sampling nodes',
  main = 'All starting nodes equally likely (deterministic)'
  )
```

**All starting nodes equally likely (deterministic)**



Darker colors indicate better sampling nodes

## The function smartsurv performs the same analysis, for the specified number of realizations

The first component of the output from smartsurv is the response for each simulation - nrealz outputs from multistart (in this example, 10 realizations). The next component of the output is the mean across the realizations The final component is the variance across realizations

Because columns represent potential sampling nodes, the mean of a column (in the second component of the output, meanarr) is the mean number of nodes free of invasion by the time the invasion would be detected at that node. The mean of the variance in a column (in the third component of the output, vararr) indicates how consistently the mean number of nodes would have been free of invasion.

```
ss.out <- smartsurv(adjmat = sAmat,
                    stoch = T,
                    nrealz = 100)

# mean across the realizations
ss.out$meanarr
```

```
##      [,1] [,2] [,3] [,4]
## [1,] 3.00 1.91 1.26 1.61
## [2,] 1.92 3.00 1.99 1.92
## [3,] 2.54 2.54 3.00 2.54
## [4,] 3.00 3.00 3.00 3.00
```

```
# variance across the realizations
ss.out$vararr
```

```
##           [,1]      [,2]      [,3]      [,4]
## [1,] 0.0000000 0.5675758 1.4670707 0.9675758
## [2,] 0.5389899 0.0000000 0.4746465 0.5389899
## [3,] 0.2509091 0.2509091 0.0000000 0.2509091
## [4,] 0.0000000 0.0000000 0.0000000 0.0000000
```

```
# a few of the individual realizations
ss.out$outarr[,,1:3]
```

```
## , , 1
##
##      [,1] [,2] [,3] [,4]
## [1,]    3    1    0    1
## [2,]    2    3    2    2
## [3,]    3    3    3    3
## [4,]    3    3    3    3
##
## , , 2
##
##      [,1] [,2] [,3] [,4]
## [1,]    3    3    3    3
## [2,]    1    3    1    1
## [3,]    3    3    3    3
## [4,]    3    3    3    3
##
```

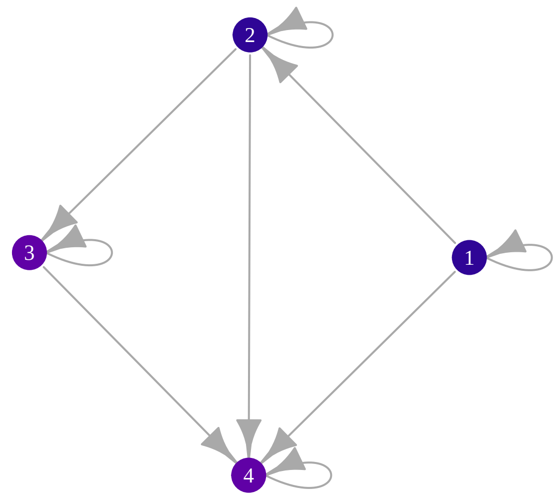
```
## , , 3
##
##      [,1] [,2] [,3] [,4]
## [1,]    3    1    0    1
## [2,]    2    3    2    2
## [3,]    2    2    3    2
## [4,]    3    3    3    3
```

```
# the mean number of nodes free of invasion for each sampling node (averaged
# across all the potential starting nodes)
numinvfree <- colMeans(ss.out$meanarr) # trait for coloring nodes
numinvfree
```

```
## [1] 2.6150 2.6125 2.3125 2.2675
```

```
# Coloring nodes based on their role in sampling: the mean number of nodes free
# of invasion when invasion detected at sampling node

nnodes <- nrow(Amat) # the total number of nodes

pal1 <- viridis_pal(option="C", direction = -1, end = 0.8)

# create palette that incorporates the possible extremes in terms of number of
# nodes free from invasion (0 and nnodes-1) to make comparison across figures
# easier
nodecol = pal1(15)[as.numeric(cut(c(0, numinvfree, (nnodes - 1)), breaks = 15))]
nodecol <- nodecol[2:(nnodes + 1)]

plot(
  Amati,
  vertex.frame.color = nodecol,
  vertex.label.cex = 0.7,
  vertex.label.color = 'white',
  edge.curved = F,
  vertex.color = nodecol,
  layout = layout.ex,
  sub = 'Darker colors indicate better sampling nodes',
  main = 'All starting nodes equally likely (realz=100)'
  )
```

## All starting nodes equally likely (realz=100)



Darker colors indicate better sampling nodes

### Weighted likelihood of a node being an entry point into network

The calculations of means up to this point have been based on the assumption that each node is equally likely to be the starting node.

The function smartsurv.weight uses output from snartsurv to evaluate the mean number of nodes free of invasion when invasion would be detected at each potential sampling node, where potential starting nodes may have different probabilities of functioning as starting nodes. For example, higher risk of being the starting node might result from a node's role as a port, weather conditions associated with a node, or lack of management information at the node.

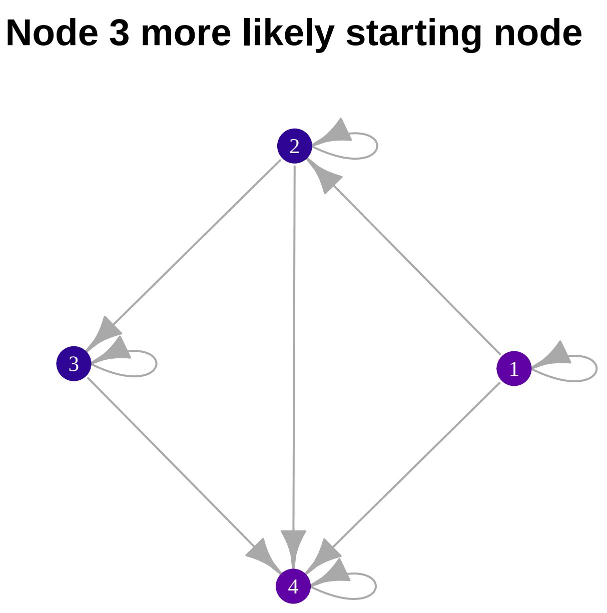
Suppose the weights indicating the probability that each of four nodes is the introduction node are as follows.

```
# weights indicating the relative likelihood that each of four nodes is the
# introduction node for an invasion
wtvec.ex <- c(0.001, 0.2, 0.798, 0.001)

msf.outex <- smartsurv(adjmat = sAmat,
                       stoch = T,
                       nrealz = 10)

temp <-
  smartsurv.weight(ss.out = msf.outex,
  adjmat = sAmat,
  wtvec = wtvec.ex)

# Note that smartsurv.weight gives output that includes number of nodes invasion
# free
```

```
numinvfree <- temp$tsampfree$sampfree # trait for coloring nodes
numinvfree
```

```
## [1] 2.3808 2.6800 2.7588 2.3798
```

```
nnodes <- dim(Amat)[1] # the total number of nodes

pal1 <- viridis_pal(option = "C",
                    direction = -1,
                    end = .8)

# create palette that incorporates the possible extremes in terms of number of
# nodes free from invasion (0 and nnodes-1) to make comparison across figures
# easier
nodecol = pal1(15)[as.numeric(cut(c(0, numinvfree, (nnodes - 1)), breaks = 15))]
nodecol <- nodecol[2:(nnodes + 1)]

plot(
  Amati,
  vertex.frame.color = nodecol,
  vertex.label.cex = 0.7,
  vertex.label.color = 'white',
  edge.curved = F,
  vertex.color = nodecol,
  layout = layout.ex,
  sub = 'Darker colors indicate better sampling nodes',
  main = 'Node 3 more likely starting node'
  )
```



Darker colors indicate better sampling nodes

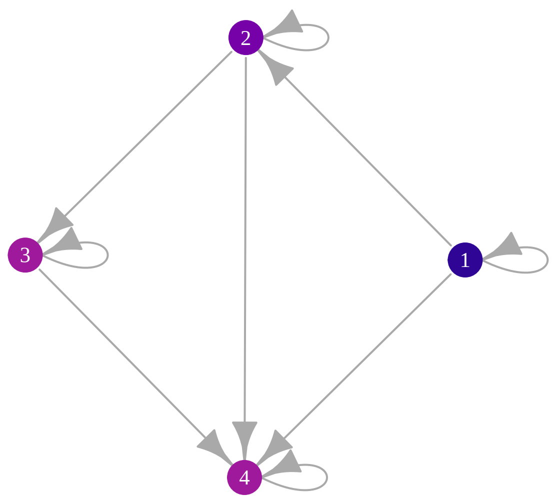
```
# Suppose node 1 is more likely to be the starting node

# weights indicating the relative likelihood that each of four nodes is the
# introduction node for an invasion
wtvec.ex <- c(0.793, 0.2, 0.001, 0.001)

msf.outex <- smartsurv(adjmat = sAmat,
                       stoch = T,
                       nrealz = 10)

temp <-
  smartsurv.weight(ss.out = msf.outex,
  adjmat = sAmat,
  wtvec = wtvec.ex)

# Note that startwt gives output that includes number of nodes invasion free
numinvfree <- temp$tsampfree$sampfree # trait for coloring nodes
numinvfree
```

```
## [1] 2.7046 2.1123 1.7141 1.7530
```

```
# create palette that incorporates the possible extremes in terms of number of
# nodes free from invasion (0 and nnodes-1) to make comparison across figures
# easier
nodecol = pal1(15)[as.numeric(cut(c(0, numinvfree, (nnodes - 1)), breaks = 15))]
nodecol <- nodecol[2:(nnodes + 1)]
```

```r
plot(
  Amati,
  vertex.frame.color = nodecol,
  vertex.label.cex = 0.7,
  vertex.label.color = 'white',
  edge.curved = F,
  vertex.color = nodecol,
  layout = layout.ex,
  sub = 'Darker colors indicate better sampling nodes',
  main = 'Node 1 more likely starting node'
  )
```

Node 1 more likely starting node



Darker colors indicate better sampling nodes

```r
#smartsurv.weight(ss.out=msf.outex, adjmat5=sAmat, wtvec=wtvec.ex,
#nodenam=c("KS","NE","ND","SD"))
```

Note that weighting has changed the outcome in this example. When node 3 is much more likely to be the starting location, the importance for sampling shifted.

## Applying the analyses to some larger networks

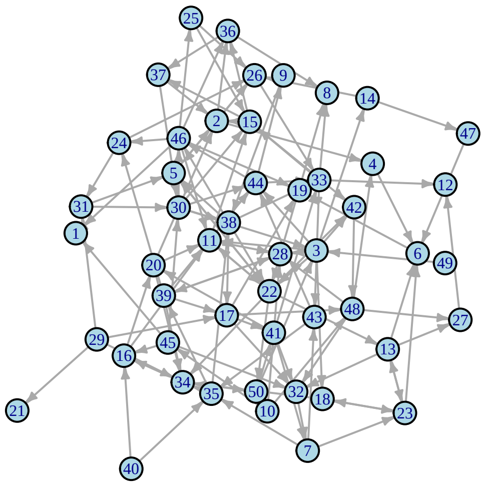
In this experiment, key nodes for sampling are identified for a set of biophysical network types.

The importance of nodes for sampling by the criteria developed here is evaluated for nine scenarios, each combination of three types of networks and three types of weighting. The three types of networks considered here are random (Erdos-Renyi), small world, and power law (Barabasi). The three types of weighting of potential starting nodes are unweighted, weights proportional to node degree, and weights inversely proportional to node degree.

## Random network

### Random network: nodes equally likely to be the starting node

```
# generate a directed random network with 50 nodes
random.eq <- sample_gnp(
  n = 50,
  p = 0.05,
  directed = T,
  loops = F
  )

layout.random <- layout_with_kk(random.eq)

plot(
  random.eq,
  edge.arrow.size = 0.3,
  vertex.size = 10,
  vertex.label.cex = 0.5,
  layout = layout.random,
  vertex.color = 'lightblue'
  )
```

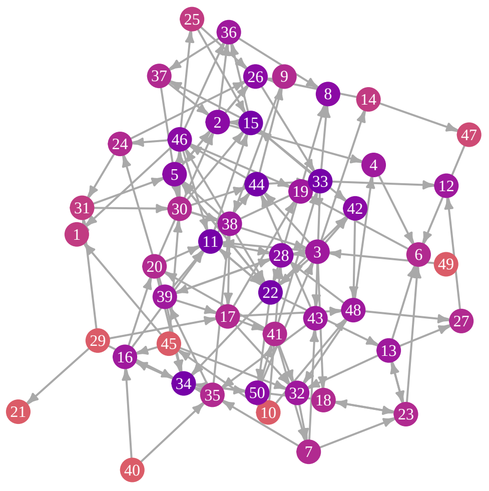


```
# find the value of nodes for sampling based on a deterministic model

temp1 <- as_adjacency_matrix(random.eq, sparse = F)
diag(temp1) <- 1 # nodes invaded stay invaded
temp2 <- multistart(adjmat = temp1, stoch = F)

numinvfree <- colMeans(temp2) # trait for coloring nodes
numinvfree
```

```
##  [1] 21.48 31.34 28.52 26.26 29.82 25.66 23.70 30.36 25.24 14.38 33.38 26.58
## [13] 26.72 21.20 33.58 26.48 26.08 23.94 27.20 23.24 14.30 33.58 23.54 25.30
## [25] 21.00 29.64 23.68 32.28 14.38 24.78 20.02 28.88 33.06 33.22 25.40 27.54
## [37] 25.54 29.24 28.42 14.36 24.22 30.52 27.16 34.42 15.20 30.80 16.50 26.36
## [49] 14.36 29.44
```

```r
nnodes <- dim(temp1)[1] # the total number of nodes


pal1 <- viridis_pal(option = "C",
                    direction = -1,
                    end = .8)

# create palette that incorporates the possible extremes in terms of number of
# nodes free from invasion (0 and nnodes-1) to make comparison across figures
# easier
nodecol = pal1(15)[as.numeric(cut(c(0, numinvfree, (nnodes - 1)), breaks = 15))]
nodecol <- nodecol[2:(nnodes + 1)]

plot(
  random.eq,
  vertex.frame.color = nodecol,
  vertex.label.cex = 0.5,
  vertex.label.color = 'white',
  vertex.size = 10,
  edge.arrow.size = 0.3,
  edge.curved = F,
  vertex.color = nodecol,
  layout = layout.random,
  sub = 'Darker colors indicate better sampling nodes',
  main = 'Random graph: all starting nodes equally likely (deterministic)'

  )
```

**Random graph: all starting nodes equally likely (deterministic)**



Darker colors indicate better sampling nodes

**Random network: likelihood a node is the starting node is proportional to node degree**

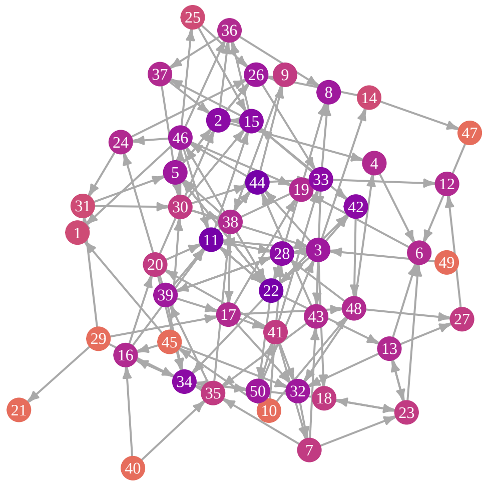
As an example, suppose that for the same network, the likelihood that the species enters the network at a given node is proportional to node degree. This might be reasonable if, for example, nodes of high degree in the known network are also more likely to be linked in other, unknown networks.

```r
# find the degree for each node and contruct a proportional weight from it
wtintermed <- degree(random.eq, mode = "all", loops = F)
wtvec1 <- wtintermed / sum(wtintermed)

temp1 <- as_adjacency_matrix(random.eq, sparse = F)
diag(temp1) <- 1 # nodes invaded stay invaded
msf.outex <- smartsurv(adjmat = temp1,
                       stoch = F,
                       nrealz = 1)

temp <-
  smartsurv.weight(ss.out = msf.outex,
  adjmat = temp1,
  wtvec = wtvec1)

# Note that startwt gives output that includes number of nodes invasion free
numinvfree <- temp$tsampfree$sampfree # trait for coloring nodes
numinvfree
```

```
##  [1] 18.461538 29.815385 26.242308 23.607692 27.969231 23.280769 21.057692
##  [8] 29.034615 22.673077 10.334615 32.861538 24.069231 25.303846 17.857692
## [15] 32.573077 23.661538 23.357692 21.203846 25.250000 19.880769 10.446154
## [22] 32.673077 21.434615 22.930769 17.976923 27.553846 21.357692 31.276923
## [29] 10.507692 21.719231 16.776923 26.903846 31.284615 31.950000 22.103846
## [36] 26.080769 23.803846 25.953846 26.576923 10.153846 22.207692 29.565385
## [43] 24.473077 33.296154 11.146154 28.788462 12.603846 24.157692  9.984615
## [50] 27.900000
```

```r
nnodes <- dim(temp1)[1] # the total number of nodes

pal1 <- viridis_pal(option = "C",
                    direction = -1,
                    end = .8)

# create palette that incorporates the possible extremes in terms of number of
# nodes free from invasion (0 and nnodes-1) to make comparison across figures
# easier
nodecol = pal1(15)[as.numeric(cut(c(0, numinvfree, (nnodes - 1)), breaks = 15))]
nodecol <- nodecol[2:(nnodes + 1)]

plot(
  random.eq,
  vertex.frame.color = nodecol,
  vertex.label.cex = 0.5,
  vertex.label.color = 'white',
  vertex.size = 10,
  edge.arrow.size = 0.3,
  edge.curved = F,
  vertex.color = nodecol,
  layout = layout.random,
  sub = 'Darker colors indicate better sampling nodes'
  )
title(main = 'Random graph: higher degree nodes more likely starting nodes (deterministic links)',
      cex.main = 0.9)
```

**Random graph: higher degree nodes more likely starting nodes (deterministic links)**



Darker colors indicate better sampling nodes

Note: “deterministic links” here refers to how the links generated when the random network was formed are maintained rather than having some probability of disappearing as they would in the stochastic option for the model.

## Random network: likelihood a node is the starting node is inversely proportional to node degree

In this case, suppose the likelihood that a node is the introduction point for an invasive in the systems is inversely proportional to node degree. This could be logical if higher node degree for potential invasive movement is associated with higher node degree for information flow, for example.

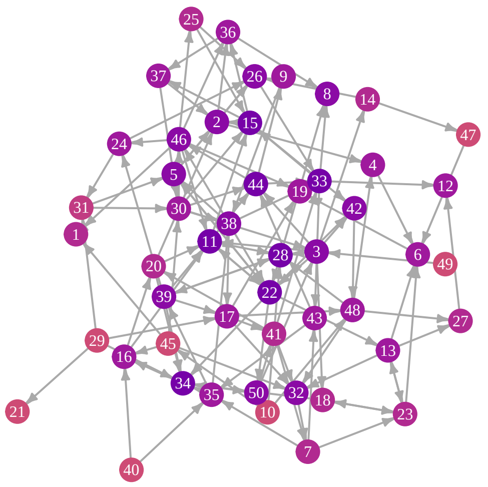
```
# find the degree for each node and contruct a weight inversely proportional to
# it
wtintermed <- degree(random.eq, mode = "all", loops = F)
hilo <- sum(range(wtintermed))
wtintermed <- hilo - wtintermed

wtvec1 <- wtintermed / sum(wtintermed)
```

```r
temp1 <- as_adjacency_matrix(random.eq, sparse = F)
diag(temp1) <- 1 # nodes invaded stay invaded
msf.outex <- smartsurv(adjmat = temp1,
                       stoch = F,
                       nrealz = 1)

temp <-
  smartsurv.weight(ss.out = msf.outex,
  adjmat = temp1,
  wtvec = wtvec1)

# Note that startwt gives output that includes number of nodes invasion free
numinvfree <- temp$tsampfree$sampfree # trait for coloring nodes
numinvfree
```

```
##  [1] 23.78824 32.50588 30.26176 28.28824 31.23529 27.47941 25.72059 31.37353
##  [9] 27.20294 17.47353 33.77647 28.50000 27.80294 23.75588 34.35000 28.63529
## [17] 28.16176 26.03235 28.69118 25.80882 17.24706 34.27353 25.15000 27.11176
## [25] 23.31176 31.23529 25.45588 33.04706 17.34118 27.12059 22.50000 30.39118
## [33] 34.41765 34.19118 27.92059 28.65588 26.86765 31.75294 29.82941 17.57647
## [41] 25.75882 31.25000 29.21471 35.27941 18.30000 32.33824 19.47941 28.04412
## [49] 17.70588 30.61765
```

```r
nnodes <- dim(temp1)[1] # the total number of nodes

pal1 <- viridis_pal(option = "C",
                    direction = -1,
                    end = .8)

# create palette that incorporates the possible extremes in terms of number of
# nodes free from invasion (0 and nnodes-1) to make comparison across figures
# easier
nodecol = pal1(15)[as.numeric(cut(c(0, numinvfree, (nnodes - 1)), breaks = 15))]
nodecol <- nodecol[2:(nnodes + 1)]

plot(
  random.eq,
  vertex.frame.color = nodecol,
  vertex.label.cex = 0.5,
  vertex.label.color = 'white',
  vertex.size = 10,
  edge.arrow.size = 0.3,
  edge.curved = F,
  vertex.color = nodecol,
  layout = layout.random,
  sub = 'Darker colors indicate better sampling nodes'
  )
title(main = 'Random graph: higher degree nodes less likely starting nodes (deterministic links)',
      cex.main = 0.9)
```

**Random graph: higher degree nodes less likely starting nodes (deterministic links)**

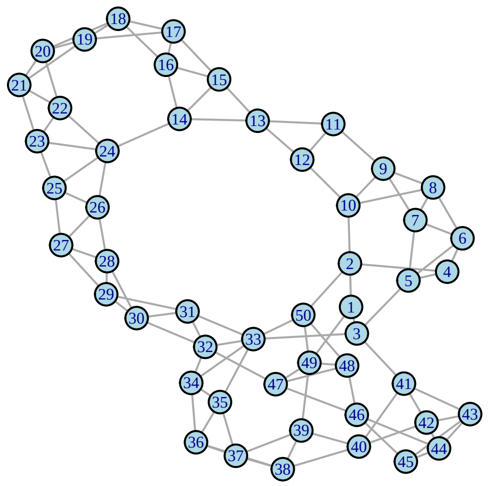


Darker colors indicate better sampling nodes

## Smalll world network

**Small world network: nodes equally likely to be the starting node**

```
# generate a small world network with 50 nodes
small.eq <- sample_smallworld(
  dim = 1,
  size = 50,
  nei = 2,
  p = 0.05,
  loops = F
  )

layout.small <- layout_with_kk(small.eq)

plot(
  small.eq,
  edge.arrow.size = 0.3,
  vertex.size = 10,
  vertex.label.cex = 0.5,
  layout = layout.small,
  vertex.color = 'lightblue'
  )
```

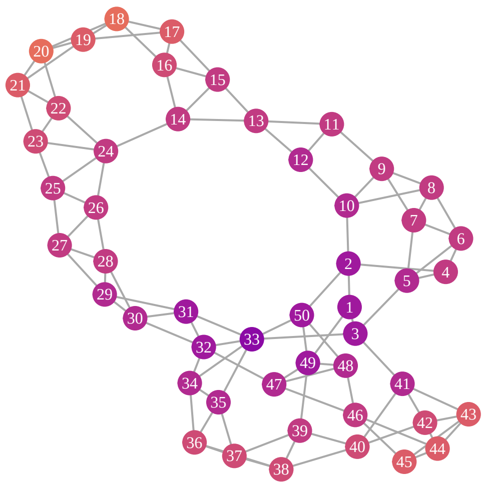


```r
# find the value of nodes for sampling based on a deterministic model

temp1 <- as_adjacency_matrix(small.eq, sparse = F)
diag(temp1) <- 1 # nodes invaded stay invaded
temp2 <- multistart(adjmat = temp1, stoch = F)

numinvfree <- colMeans(temp2) # trait for coloring nodes
numinvfree
```

```
##  [1] 26.78 28.18 28.94 21.46 24.82 20.46 22.36 20.98 22.82 25.54 21.54 23.34
## [13] 22.72 21.98 19.98 17.10 15.62 12.68 13.28 13.04 14.26 17.10 18.24 22.20
## [25] 20.54 21.62 21.46 22.56 23.10 24.52 27.08 28.24 30.86 25.30 23.08 18.44
## [37] 17.28 16.66 20.28 17.84 23.28 16.90 16.04 15.44 14.60 20.92 25.58 25.00
## [49] 26.44 29.26
```

```r
nnodes <- dim(temp1)[1] # the total number of nodes

pal1 <- viridis_pal(option = "C",
                    direction = -1,
                    end = .8)

# create palette that incorporates the possible extremes in terms of number of
# nodes free from invasion (0 and nnodes-1) to make comparison across figures
# easier
nodecol = pal1(15)[as.numeric(cut(c(0, numinvfree, (nnodes - 1)), breaks = 15))]
nodecol <- nodecol[2:(nnodes + 1)]

plot(
  small.eq,
  vertex.frame.color = nodecol,
  vertex.label.cex = 0.5,
  vertex.label.color = 'white',
  vertex.size = 10,
  edge.arrow.size = 0.3,
  edge.curved = F,
  vertex.color = nodecol,
  layout = layout.small,
  sub = 'Darker colors indicate better sampling nodes',
  main = 'Small world: all starting nodes equally likely (deterministic links)'
  )
```

## Small world: all starting nodes equally likely (deterministic links)



Darker colors indicate better sampling nodes

Note that this small world network is not directed, and movement through the network can be more rapid than in the random graph example. There is a path between any two nodes.

**Small world network: likelihood a node is the starting node is proportional to node degree**

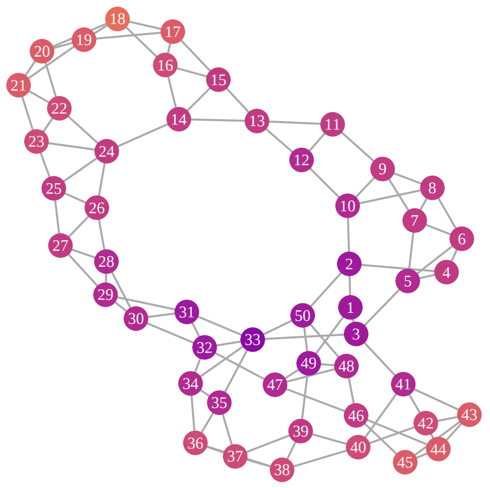
```
# find the degree for each node and contruct a proportional weight from it
wtintermed <- degree(small.eq, mode = "all", loops = F)
wtvec1 <- wtintermed / sum(wtintermed)

temp1 <- as_adjacency_matrix(small.eq, sparse = F)
diag(temp1) <- 1 # nodes invaded stay invaded
msf.outex <- smartsurv(adjmat = temp1,
                       stoch = F,
                       nrealz = 1)

temp <-
  smartsurv.weight(ss.out = msf.outex,
  adjmat = temp1,
  wtvec = wtvec1)

# Note that startwt gives output that includes number of nodes invasion free
```

```r
numinvfree <- temp$tsampfree$sampfree # trait for coloring nodes
numinvfree
```

```
##  [1] 26.610 27.920 28.920 21.085 24.585 20.085 22.040 20.545 22.420 25.150
## [11] 21.145 22.920 22.375 21.775 19.710 16.910 15.380 12.575 13.185 13.105
## [21] 14.365 17.150 18.330 22.190 20.680 21.805 21.725 22.875 23.565 25.025
## [31] 27.545 28.745 31.160 25.735 23.460 18.865 17.645 16.860 20.325 17.865
## [41] 23.155 16.840 15.920 15.455 14.600 20.995 25.860 25.000 26.445 29.270
```

```r
nnodes <- dim(temp1)[1] # the total number of nodes

pal1 <- viridis_pal(option = "C",
                    direction = -1,
                    end = .8)

# create palette that incorporates the possible extremes in terms of number of
# nodes free from invasion (0 and nnodes-1) to make comparison across figures
# easier
nodecol = pal1(15)[as.numeric(cut(c(0, numinvfree, (nnodes - 1)), breaks = 15))]
nodecol <- nodecol[2:(nnodes + 1)]

plot(
  small.eq,
  vertex.frame.color = nodecol,
  vertex.label.cex = 0.5,
  vertex.label.color = 'white',
  vertex.size = 10,
  edge.arrow.size = 0.3,
  edge.curved = F,
  vertex.color = nodecol,
  layout = layout.small,
  sub = 'Darker colors indicate better sampling nodes'
  )
title(main = 'Small world: higher degree nodes more likely starting nodes (deterministic links)',
      cex.main = 0.9)
```

## Small world: higher degree nodes more likely starting nodes (deterministic links)



Darker colors indicate better sampling nodes

This has little effect for the small world model because the degree is so similar across all nodes.

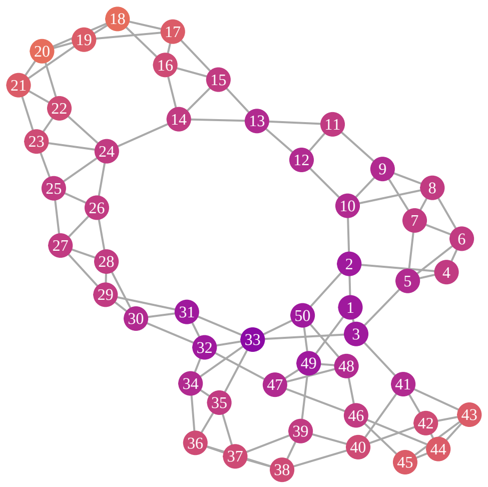
### Small world network: likelihood a node is the starting node is inversely proportional to node degree

```
# find the degree for each node and contruct a weight inversely proportional to it
wtintermed <- degree(small.eq, mode = "all", loops = F)
hilo <- sum(range(wtintermed))
wtintermed <- hilo - wtintermed

wtvec1 <- wtintermed / sum(wtintermed)

temp1 <- as_adjacency_matrix(small.eq, sparse = F)
diag(temp1) <- 1 # nodes invaded stay invaded
msf.outex <- smartsurv(adjmat = temp1,
                       stoch = F,
                       nrealz = 1)

temp <-
  smartsurv.weight(ss.out = msf.outex,
  adjmat = temp1,
  wtvec = wtvec1)

# Note that startwt gives output that includes number of nodes invasion free
numinvfree <- temp$tsampfree$sampfree # trait for coloring nodes
numinvfree
```

```
##  [1] 26.916 28.388 28.956 21.760 25.008 20.760 22.616 21.328 23.140 25.852
## [11] 21.856 23.676 22.996 22.144 20.196 17.252 15.812 12.764 13.356 12.988
## [21] 14.176 17.060 18.168 22.208 20.428 21.472 21.248 22.308 22.728 24.116
## [31] 26.708 27.836 30.620 24.952 22.776 18.100 16.988 16.500 20.244 17.820
## [41] 23.380 16.948 16.136 15.428 14.600 20.860 25.356 25.000 26.436 29.252
```

```
nnodes <- dim(temp1)[1] # the total number of nodes

pal1 <- viridis_pal(option = "C",
                    direction = -1,
                    end = .8)

# create palette that incorporates the possible extremes in terms of number of
# nodes free from invasion (0 and nnodes-1) to make comparison across figures
# easier
nodecol = pal1(15)[as.numeric(cut(c(0, numinvfree, (nnodes - 1)), breaks = 15))]
nodecol <- nodecol[2:(nnodes + 1)]

plot(
  small.eq,
  vertex.frame.color = nodecol,
  vertex.label.cex = 0.5,
  vertex.label.color = 'white',
  vertex.size = 10,
  edge.arrow.size = 0.3,
  edge.curved = F,
  vertex.color = nodecol,
  layout = layout.small,
  sub = 'Darker colors indicate better sampling nodes'
  )
title(main = 'Small world: higher degree nodes less likely starting nodes (deterministic links)',
      cex.main = 0.9)
```

**Small world: higher degree nodes less likely starting nodes (deterministic links)**

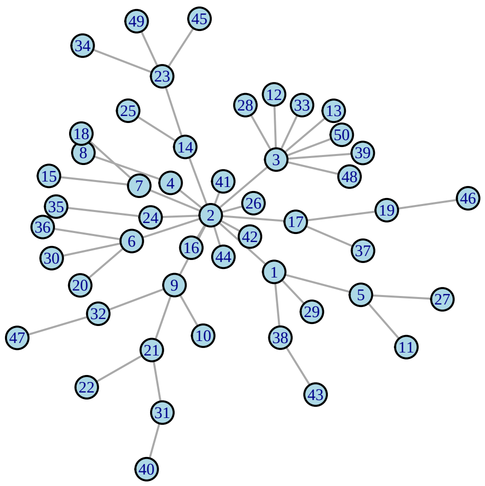


Darker colors indicate better sampling nodes

## Scale-free network

### Scale-free network: nodes equally likely to be the starting node

```
# generate a scale-free network with 50 nodes
sf.eq <- sample_pa(
  n = 50,
  power = 1,
  m = 1,
  directed = F
  )

layout.sf <- layout_with_kk(sf.eq)

plot(
  sf.eq,
  edge.arrow.size = 0.3,
  vertex.size = 10,
  vertex.label.cex = 0.5,
  layout = layout.sf,
  vertex.color = 'lightblue'
  )
```



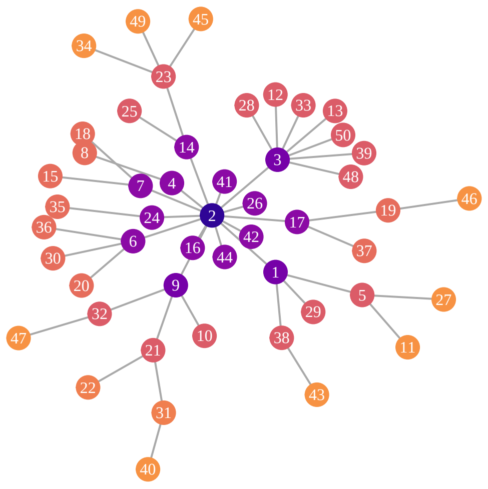
```
# find the value of nodes for sampling based on a deterministic model

temp1 <- as_adjacency_matrix(sf.eq, sparse = F)
diag(temp1) <- 1 # nodes invaded stay invaded
temp2 <- multistart(adjmat = temp1, stoch = F)

numinvfree <- colMeans(temp2) # trait for coloring nodes
numinvfree
```

```
##  [1] 33.24 44.44 33.94 31.30 14.84 32.02 31.64 10.98 33.62 15.38  5.84 14.46
## [13] 14.46 32.66 11.66 31.00 31.98 11.66 12.48 12.30 15.76  6.86 14.24 31.30
## [25] 13.88 31.00  5.84 14.46 14.60 12.30  6.94 15.48 14.46  5.60 10.98 12.30
## [37] 12.40 14.70 14.46  4.16 31.00 31.00  5.52 31.00  5.60  3.96  6.16 14.46
## [49]  5.60 14.46
```

```
nnodes <- dim(temp1)[1] # the total number of nodes

pal1 <- viridis_pal(option = "C",
                    direction = -1,
                    end = .8)

# create palette that incorporates the possible extremes in terms of number of
# nodes free from invasion (0 and nnodes-1) to make comparison across figures
# easier
nodecol = pal1(15)[as.numeric(cut(c(0, numinvfree, (nnodes - 1)), breaks = 15))]
nodecol <- nodecol[2:(nnodes + 1)]

plot(
  sf.eq,
  vertex.frame.color = nodecol,
  vertex.label.cex = 0.5,
  vertex.label.color = 'white',
  vertex.size = 10,
  edge.arrow.size = 0.3,
  edge.curved = F,
  vertex.color = nodecol,
  layout = layout.sf,
```

```r
  sub = 'Darker colors indicate better sampling nodes',
  main = 'Scale-free: all starting nodes equally likely (deterministic links)'
  )
```

**Scale−free: all starting nodes equally likely (deterministic links)**



Darker colors indicate better sampling nodes

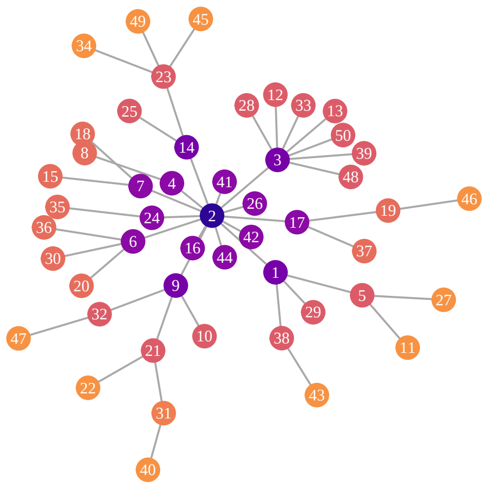
**Scale-free network: likelihood a node is the starting node is proportional to node degree**

```r
# find the degree for each node and contruct a proportional weight from it
wtintermed <- degree(sf.eq, mode = "all", loops = F)
wtvec1 <- wtintermed / sum(wtintermed)

temp1 <- as_adjacency_matrix(sf.eq, sparse = F)
diag(temp1) <- 1 # nodes invaded stay invaded
msf.outex <- smartsurv(adjmat = temp1,
                       stoch = F,
                       nrealz = 1)

temp <-smartsurv.weight(ss.out = msf.outex,
                   adjmat = temp1,
                   wtvec = wtvec1)

# Note that startwt gives output that includes number of nodes invasion free
numinvfree <- temp$tsampfree$sampfree # trait for coloring nodes
numinvfree
```

```
##  [1] 33.418367 44.846939 34.132653 31.438776 14.642857 32.173469 31.785714
##  [8] 10.704082 33.806122 15.193878  5.459184 14.255102 14.255102 32.826531
## [15] 11.397959 31.132653 32.132653 11.397959 12.234694 12.051020 15.581633
## [22]  6.500000 14.030612 31.438776 13.663265 31.132653  5.459184 14.255102
## [29] 14.397959 12.051020  6.581633 15.295918 14.255102  5.214286 10.704082
## [36] 12.051020 12.153061 14.500000 14.255102  3.744898 31.132653 31.132653
## [43]  5.132653 31.132653  5.214286  3.540816  5.785714 14.255102  5.214286
## [50] 14.255102
```

```r
nnodes <- dim(temp1)[1] # the total number of nodes

pal1 <- viridis_pal(option = "C",
                    direction = -1,
                    end = .8)

# create palette that incorporates the possible extremes in terms of number of
# nodes free from invasion (0 and nnodes-1) to make comparison across figures
# easier
nodecol = pal1(15)[as.numeric(cut(c(0, numinvfree, (nnodes - 1)), breaks = 15))]
nodecol <- nodecol[2:(nnodes + 1)]

plot(
  sf.eq,
  vertex.frame.color = nodecol,
  vertex.label.cex = 0.5,
  vertex.label.color = 'white',
  vertex.size = 10,
  edge.arrow.size = 0.3,
  edge.curved = F,
  vertex.color = nodecol,
  layout = layout.sf,
  sub = 'Darker colors indicate better sampling nodes'
  )
title(main = 'Scale-free: higher degree nodes more likely starting nodes (deterministic links)',
      cex.main = 0.9)
```

**Scale−free: higher degree nodes more likely starting nodes (deterministic links)**



Darker colors indicate better sampling nodes

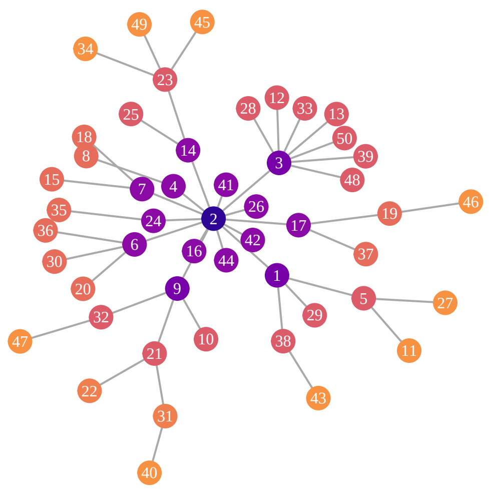
**Scale-free network: likelihood a node is the starting node is inversely proportional to node degree**

```
# find the degree for each node and contruct a weight inversely proportional to
# it
wtintermed <- degree(sf.eq, mode = "all", loops = F)
hilo <- sum(range(wtintermed))
wtintermed <- hilo - wtintermed

wtvec1 <- wtintermed / sum(wtintermed)

temp1 <- as_adjacency_matrix(sf.eq, sparse = F)
diag(temp1) <- 1 # nodes invaded stay invaded
msf.outex <- smartsurv(adjmat = temp1,
                       stoch = F,
                       nrealz = 1)

temp <-
  smartsurv.weight(ss.out = msf.outex,
  adjmat = temp1,
  wtvec = wtvec1)

# Note that startwt gives output that includes number of nodes invasion free
numinvfree <- temp$tsampfree$sampfree # trait for coloring nodes
numinvfree
```

```
##  [1] 33.213190 44.378834 33.911043 31.279141 14.869632 31.996933 31.618098
##  [8] 11.021472 33.592025 15.407975  5.897239 14.490798 14.490798 32.634969
## [15] 11.699387 30.980061 31.957055 11.699387 12.516871 12.337423 15.786810
## [22]  6.914110 14.271472 31.279141 13.912577 30.980061  5.897239 14.490798
## [29] 14.630368 12.337423  6.993865 15.507669 14.490798  5.657975 11.021472
## [36] 12.337423 12.437117 14.730061 14.490798  4.222393 30.980061 30.980061
## [43]  5.578221 30.980061  5.657975  4.023006  6.216258 14.490798  5.657975
## [50] 14.490798
```

```
nnodes <- dim(temp1)[1] # the total number of nodes

pal1 <- viridis_pal(option = "C",
                    direction = -1,
                    end = .8)

# create palette that incorporates the possible extremes in terms of number of
# nodes free from invasion (0 and nnodes-1) to make comparison across figures
# easier
nodecol = pal1(15)[as.numeric(cut(c(0, numinvfree, (nnodes - 1)), breaks = 15))]
nodecol <- nodecol[2:(nnodes + 1)]

plot(
  sf.eq,
  vertex.frame.color = nodecol,
  vertex.label.cex = 0.5,
  vertex.label.color = 'white',
  vertex.size = 10,
  edge.arrow.size = 0.3,
  edge.curved = F,
  vertex.color = nodecol,
  layout = layout.sf,
  sub = 'Darker colors indicate better sampling nodes'
  )
title(main = 'Scale-free: higher degree nodes less likely starting nodes (deterministic links)',
      cex.main = 0.9)
```

**Scale−free: higher degree nodes less likely starting nodes (deterministic links)**



Darker colors indicate better sampling nodes

Using weights based on the number of information sources, or the quality of information sources, is one way to integrate the socioeconomic network and the biophysical network, with an egocentric network focus. There are many other possibilities for linking the two networks, potentially drawing on more information about the network of information communication. Examples of this analysis applied to potential spread of disease through a seed system are available in the following two studies.

## References

More information and references are available at www.garrettlab.com/ina

Andersen et al. 2019 and Buddenhagen et al. 2017 have examples of the application of the algorithms in INA function smartsurv.

Andersen, K. F., C. E. Buddenhagen, P. Rachkara, R. Gibson, S. Kalule, D. Phillips, and K. A. Garrett. 2019. Modeling epidemics in seed systems and landscapes to guide management strategies: The case of sweetpotato in Northern Uganda. Phytopathology 109:1519-1532.

Buddenhagen, C. E., J. F. Hernandez Nopsa, K. F. Andersen, J. Andrade-Piedra, G. A. Forbes, P. Kromann, S. Thomas-Sharma, P. Useche, and K. A. Garrett. 2017. Epidemic network analysis for mitigation of invasive pathogens in seed systems: Potato in Ecuador. Phytopathology 107:1209-1218.

Csardi, G. and T. Nepusz. 2006. The igraph software package for complex network research. InterJournal, Complex Systems: 1695. http://igraph.org

Wickham, H., J. Hester, and W. Chang. devtools: Tools to Make Developing R Packages Easier. 2020. R package version 2.3.0. https://CRAN.R-project.org/package=devtools