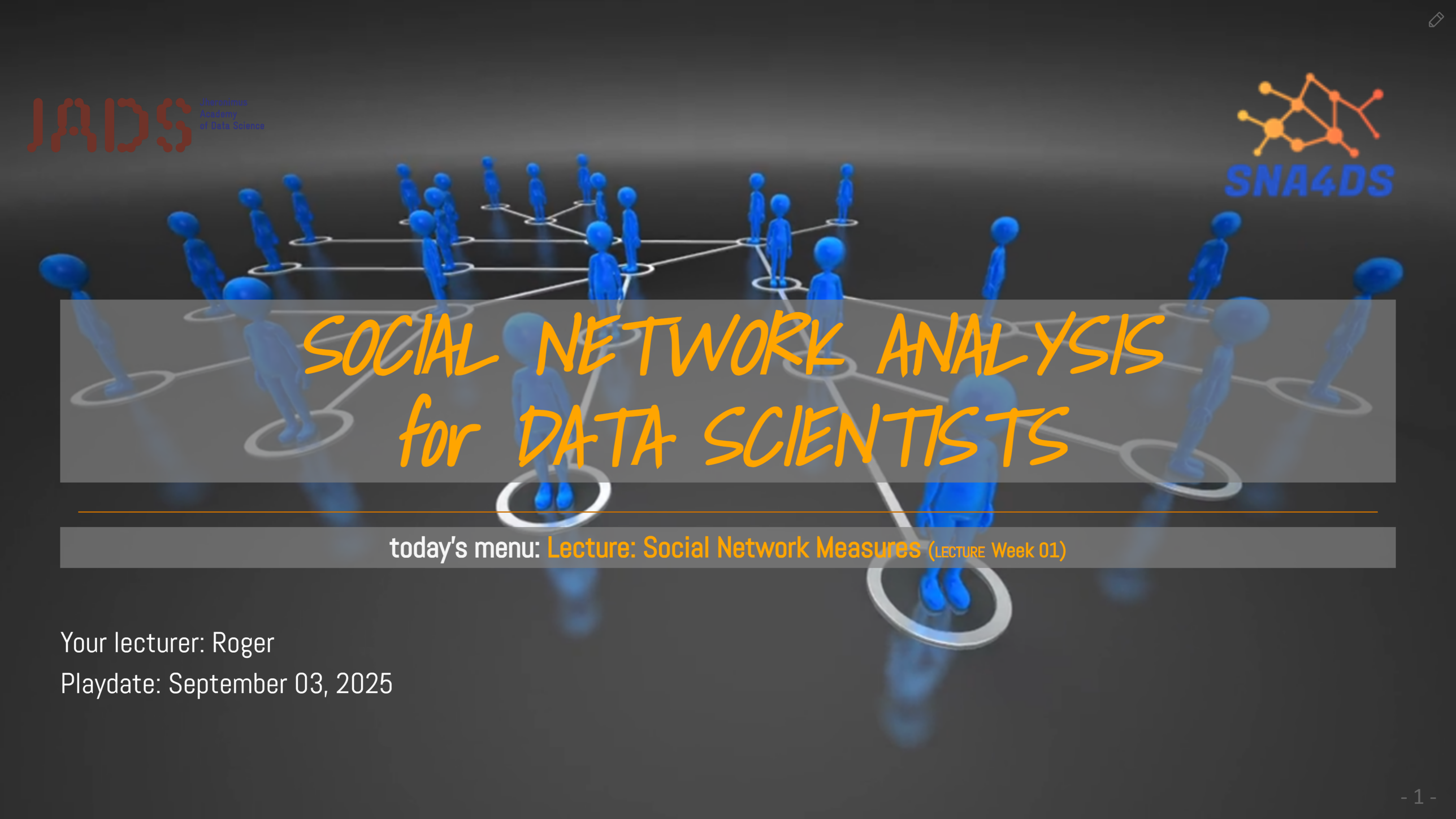


SOCIAL NETWORK ANALYSIS for DATA SCIENTISTS

today's menu: Lecture: Social Network Measures (LECTURE Week 01)

Your lecturer: Roger

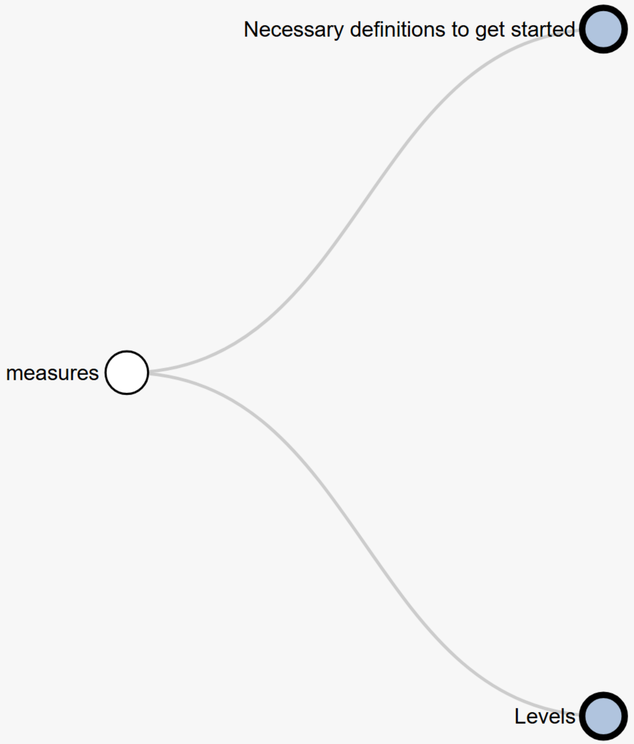
Playdate: September 03, 2025

You already learnt these:

(previous tutorial, lab, Easly & Kleinberg chapters)

- directed / undirected graph
- weighted /unweighted graph
- bipartite / unipartite graph
- dependency graph
- connectivity
- (connected) components
- giant component
- triadic closure
- clustering coefficient
- path
- cycle
- length of a path
- shortest path
- dyadic distance
- embeddedness
- neighborhood
- bridge, local bridge, span of a bridge
- tie strength (weak / strong ties)

Main network measures in this course



Why all of these definitions and measures?

1. to give you a good understanding of the graph you plan to run analyses on (***always*** do your due diligence ***before*** running advanced models!)
2. you need them as part of your analysis of the graph itself
3. because you will need these building blocks extensively for the advanced models later in this semester

Geodesic

The *length of a path* is the number of edges it goes through.

A *geodesic* from *i* to *j* is the **shortest possible** path from *i* to *j*.

(So, a geodesic and a shorest path are different terms for the same thing. We will use both terms interchangeably.)

The *geodesic length* is the minimum number of edges needed to move from *i* to *j*. This is also called the *distance* from *i* to *j*.

The *diameter* of the network is the longest distance across all pairs of nodes.

In a **weighted graph**, the shortest path from *i* to *j* is the path that has the smallest possible sum of weights among all possible paths from *i* to *j*.

Weighted networks

- In a *weighted* network, the edges are not binary, but have a numeric value.
  - the number of times a webpage links to another specific webpage
  - the average number of times two people talk to each other during the week
  - the $ of economic trade between two countries
  - et cetera

Multiplex networks

- In a *multiplex* (or *multilevel*) network, there are multiple kinds of edges possible between nodes
  - two people being connected by friendship, trust, coworkership, advice-sharing
  - two countries engaging in economic trade, cultural trade, collaboration in innovation
  - et cetera

Vertex-level indices

Quantifying node position

There are MANY MANY MANY ways to quantify the position of a node in a network.

Of course, which to choose depends on the research objective.

Let's look at some of the prominent measures.

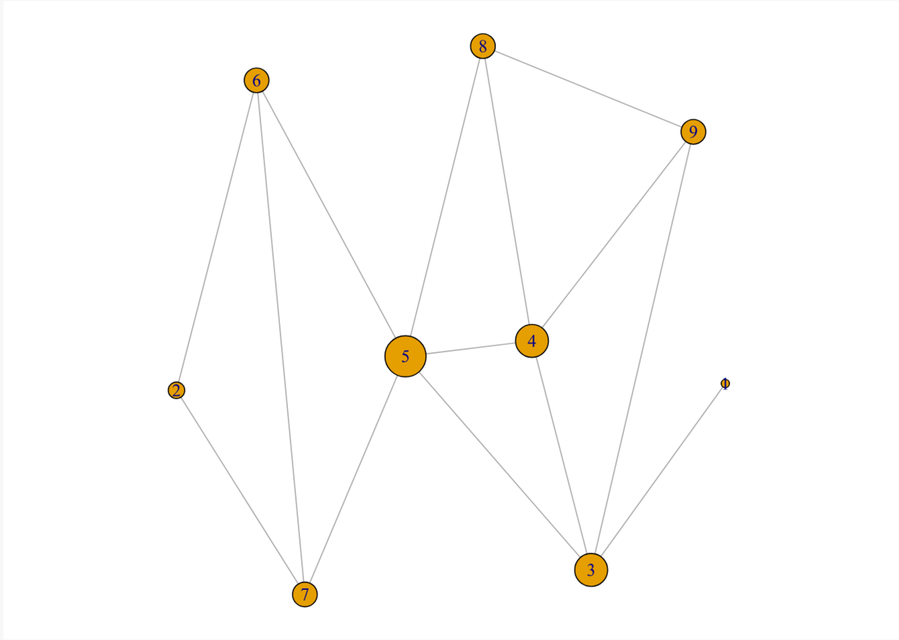
Degree

Degree measures a node's *extraversion/outgoingness* ("outdegree"), *popularity* ("indegree"), or *involvement* ("total degree").

outdegree = number of outgoing edges

indegree = number of incoming edges

total degree = total number of neighbors



Node size is proportional to its degree

```
snafun::v_degree(graph, vids = NULL, mode = c("all", "out", "in"), loops = FALSE, rescaled = FALSE)
```

Closeness

Closeness measures how much effort it takes to reach all other nodes in the network. Sum the distances from *i* to all other vertices, this is *i*'s *farness*. Then, invert this sum.

Formally:

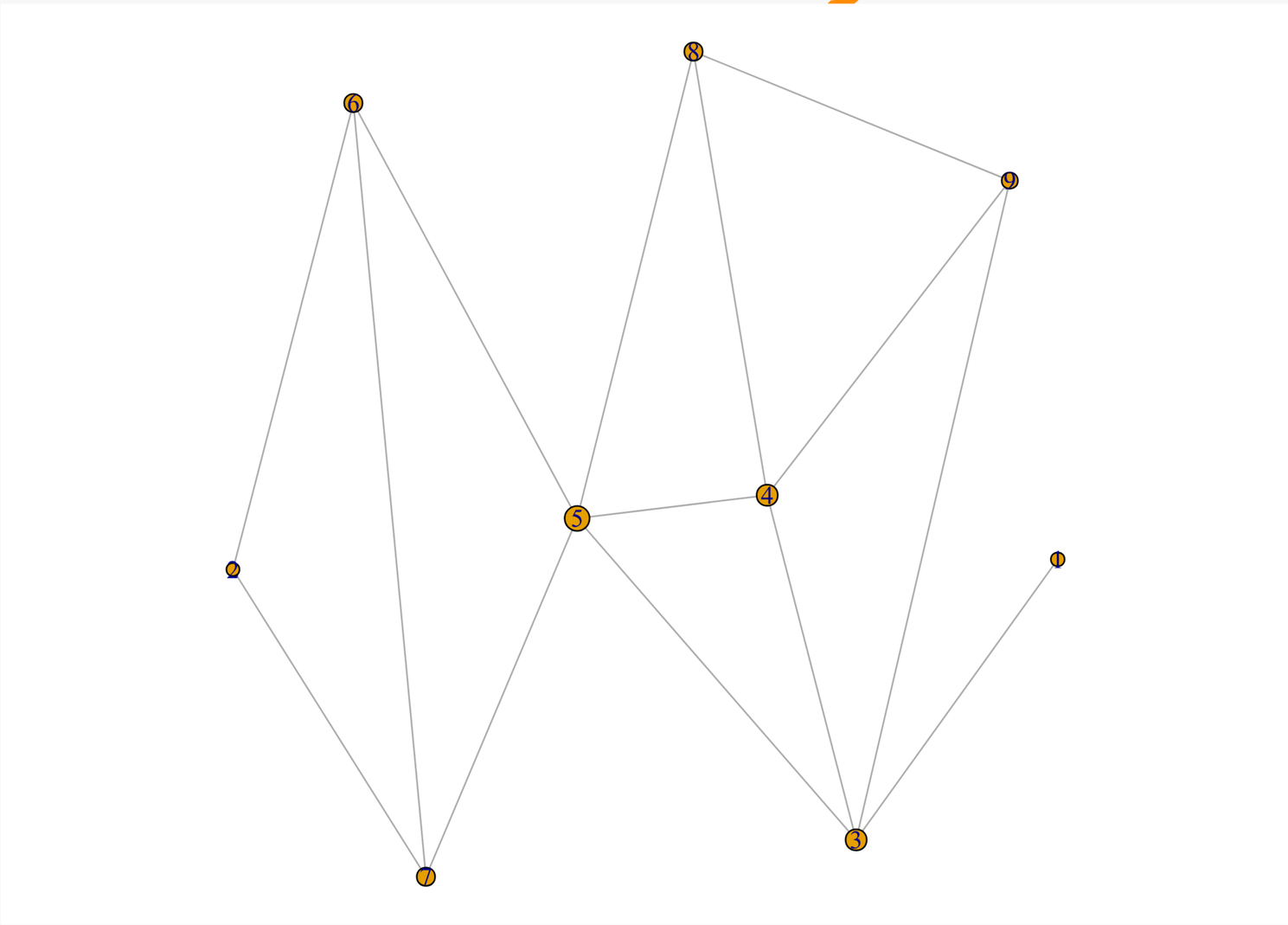
$$Closeness(i) = \frac{1}{\sum_{i \neq v} d(v, i)}$$

with *d(v, i)* equal to the path length between *i* and *v*.

Closeness is ill-defined when the network is not fully connected (b/c there are no paths possible for each dyad). Different implementations solve this is different ways and, hence, give different results for such networks.

```
snafun::v_closeness(graph, vids = NULL, mode = c("all", "out", "in"), rescaled = FALSE)
```

Closeness in the example graph



Node size is proportional to its closeness centrality

Stress centrality

Stress centrality measures the amount of 'work' or 'stress' a vertex has to sustain in the network. A vertex is more central as more shortest paths run through it.

So: determine all shortest paths between every pair of vertices in the network and calculate the number that go through *i*.

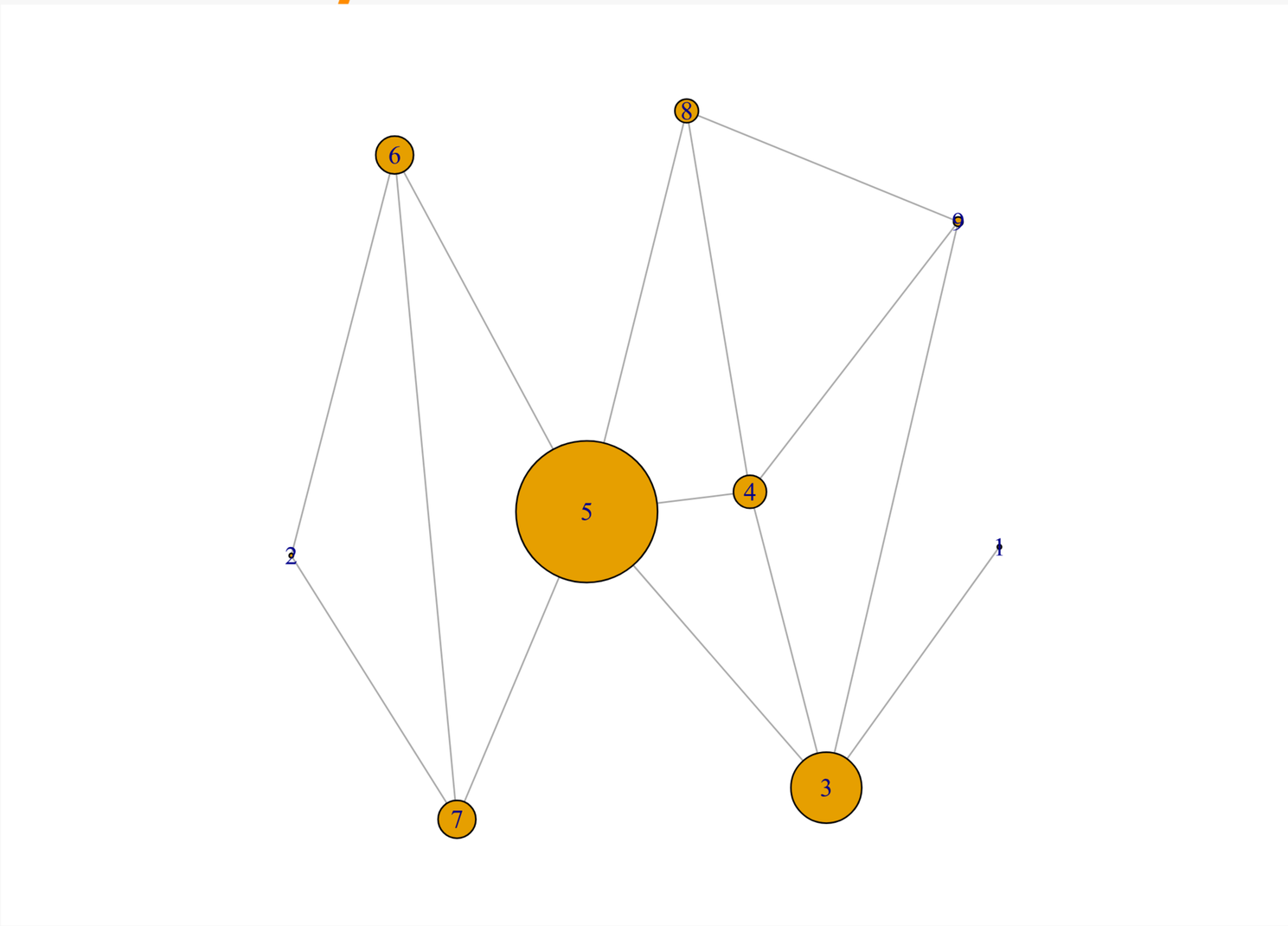
$$Stress(i) = \sum_{\substack{j \neq i \\ h \neq i}} \sigma_{hj}(i)$$

where $\sigma_{hj}(i)$ denotes the number of shortest paths between *h* and *j* that pass through *i*.

In words: *i*'s stress centrality measures how many shortest possible routes go through *i*.

```
snafun::v_stress(g, vids = NULL, directed = TRUE, rescaled = FALSE)
```

Stress centrality



Node size is proportional to its stress centrality

Betweenness centrality

Betweenness centrality makes stress centrality relative, by taking into account how many geodesics exist in the network anyway.

Formally:

$$Betweenness(i) = \sum_{\substack{i,j \\ i \neq j \\ h \neq i \\ h \neq j}} \frac{\sigma_{hj}(i)}{\sigma_{hj}}$$

where:

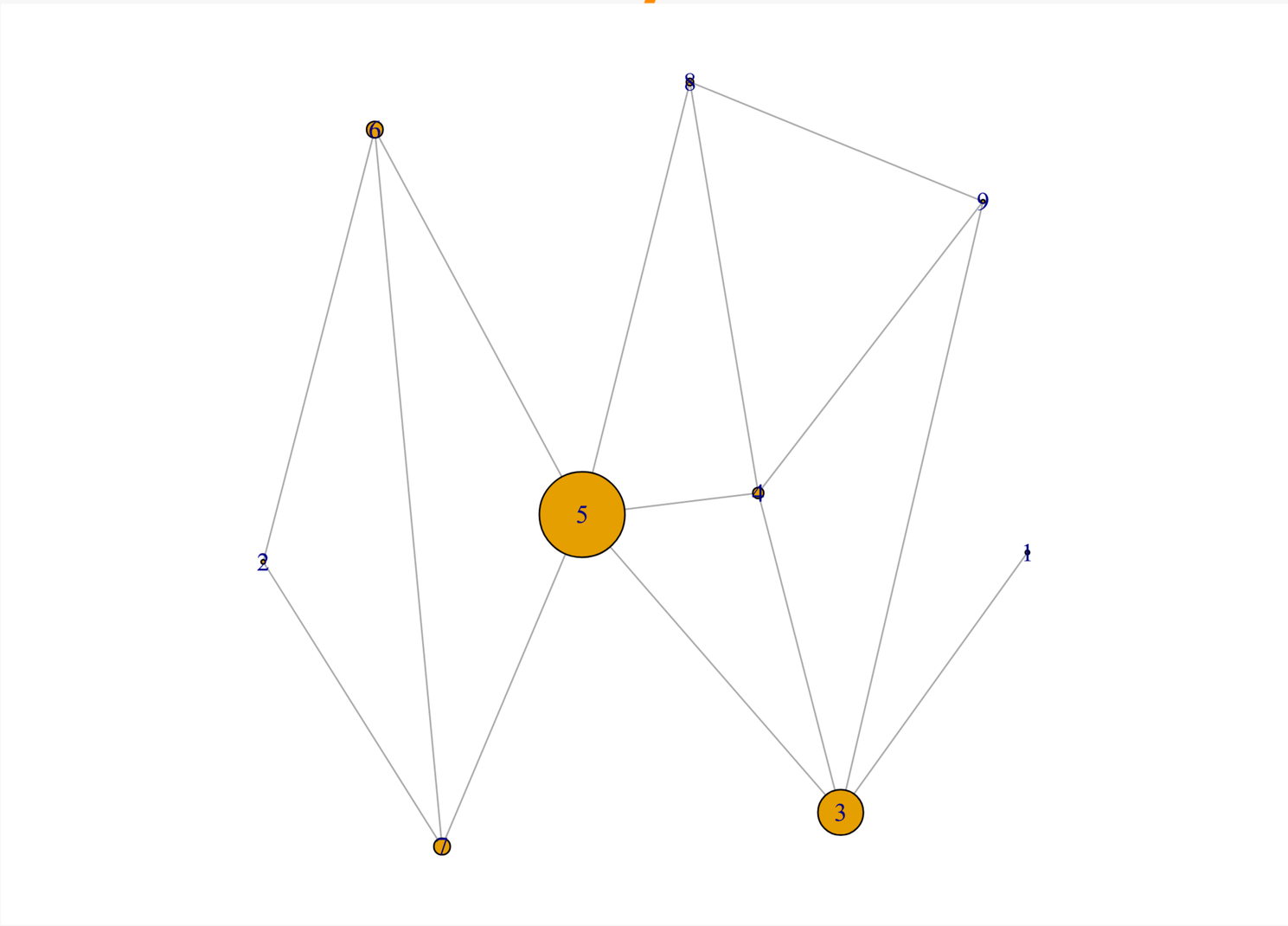
σ_{hj} is the number of geodesics from h to j

$\sigma_{hj}(i)$ is the number of geodesics from h to j that run through i

In words: the betweenness centrality of *i* is the proportion of all shortest paths between actors other than *i* in the graph that pass through *i*. Betweenness shows which nodes have information access advantage and are important to the network's efficiency. It also shows the relative stress on nodes.

```
snafun::v_betweenness(x, vids = NULL, directed = TRUE, rescaled = FALSE)
```

Betweenness centrality



Node size is proportional to its betweenness centrality

Interpreting centrality scores

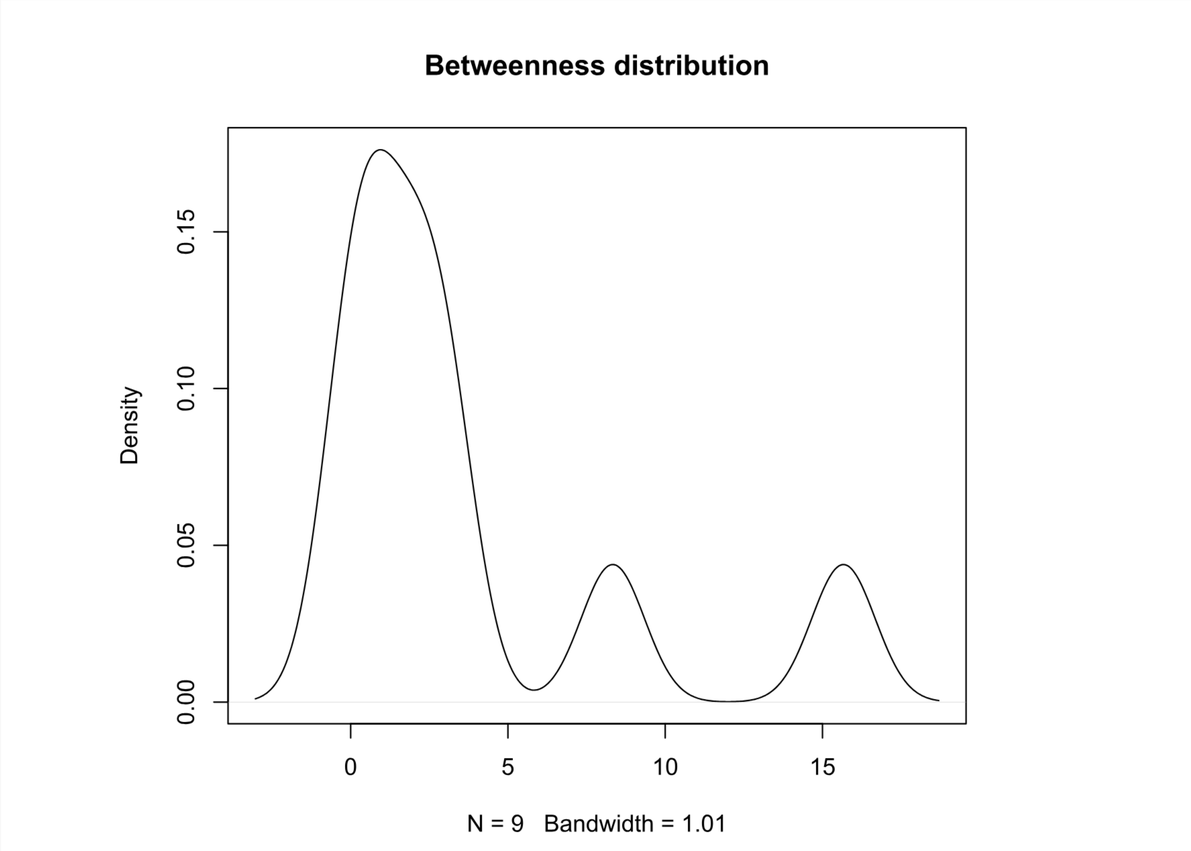
Centrality describes specific vertices (or edges), but says nothing about the whole network

- Centrality scores are not/barely comparable between networks, because they depend heavily on the specific graph (especially its size and density).
- It is possible to condition on this, but that is beyond today's lecture.
- Centrality scores are most relevant when comparing the scores of vertices within a specific network.

Therefore, we often consider *centrality distributions*.

A basic distribution of the betweenness centralities

```
plot(density(snafun::v_betweenness(gamenet)), main = "Betweenness distribution")
```



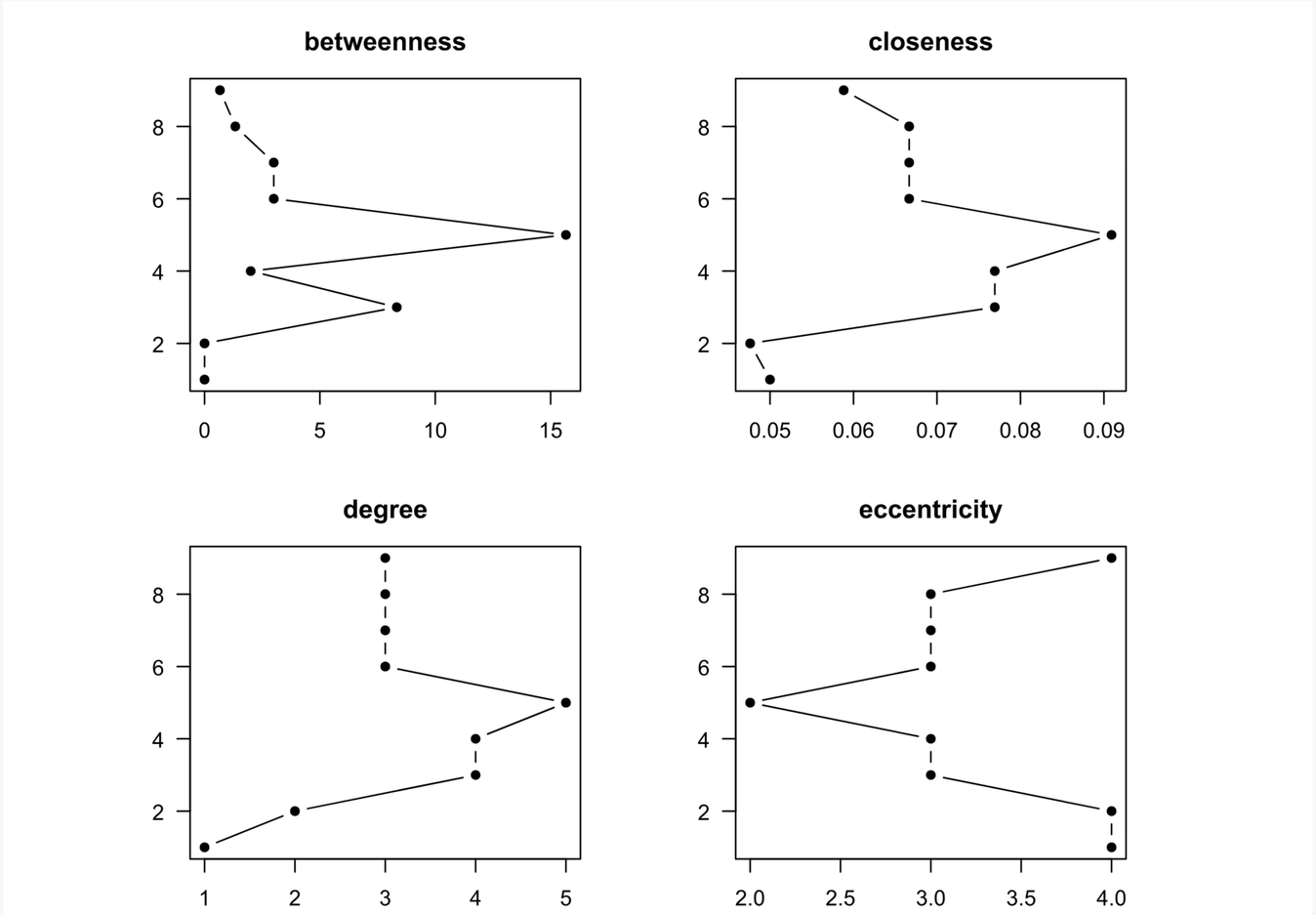
Important take-away

Many vertex (edge) measures are only meaningful when you compare the scores of the vertices (edges) within the same network with each other.

Compare across several measures

It is also informative to compare nodes across several measures at once.

```
snafun::plot_centralities(gamenet)
```



Important take-away

No matter how cool or sophisticated your network measure is, no single measure can single-handedly capture the essence of a node, an edge, a subgroup, or the network as a whole.

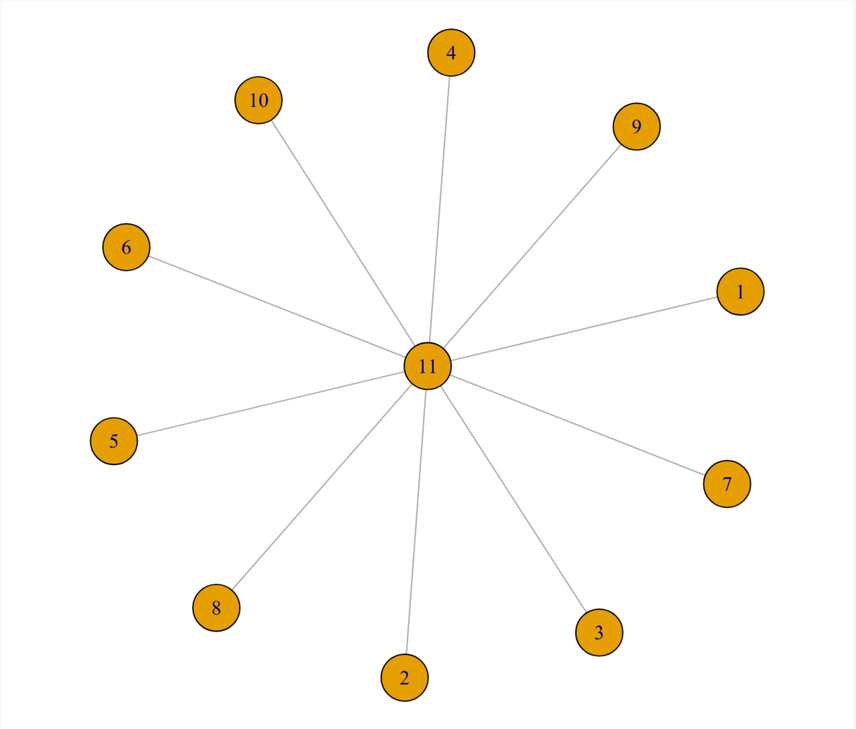
You always need multiple measures to get to the essence.

Centralization

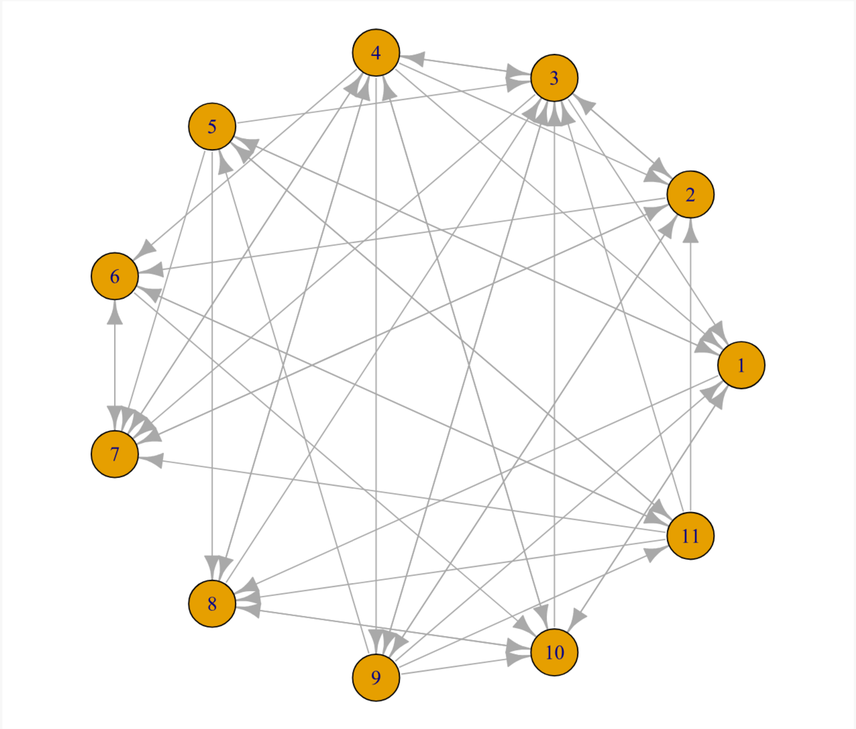
Centrality measures how "central an *individual* vertex or edge is within the network.

Centralization measures how centralized the network is as a whole.

```
star <- igraph::make_star(11, center = 11,
snafun::plot(star)
```




```
randomnet <- snafun::create_random_graph(1
snafun::plot(randomnet, layout = igraph::l
```



centralization calculation

There are two common approaches:

$Centralization = \sum_i |max(c(w)) - c(i)|$

- sum of absolute differences between each centrality and the max centrality; this is known as the *freeman* method.

or

$Centralization = sd(c(i))$

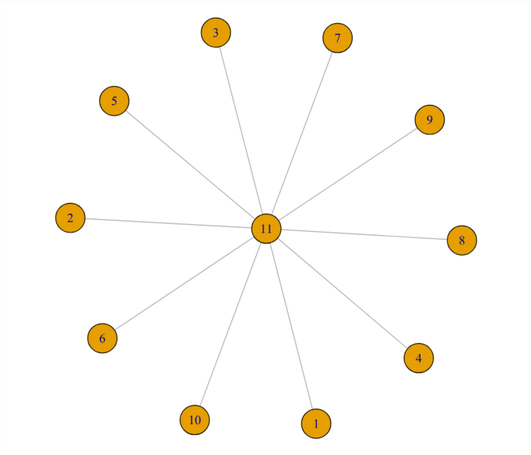
- the standard deviation of the centrality scores of the vertices in the graph.

- the *freeman* method is the default in `snafun`, `igraph`, and `sna`

Freeman centralization

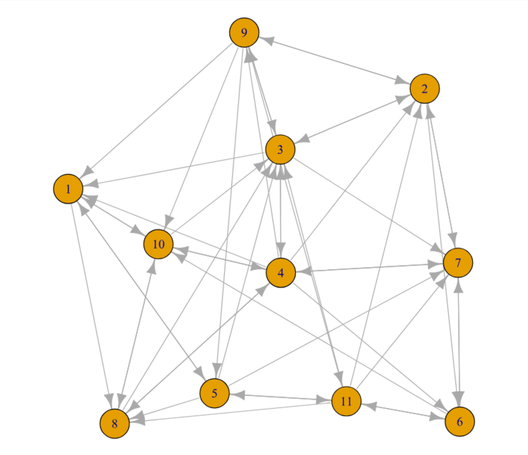
```
snafun::g_centralize(star, measure = "be
```

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




```
snafun::g_centralize(randomnet, measure =
```

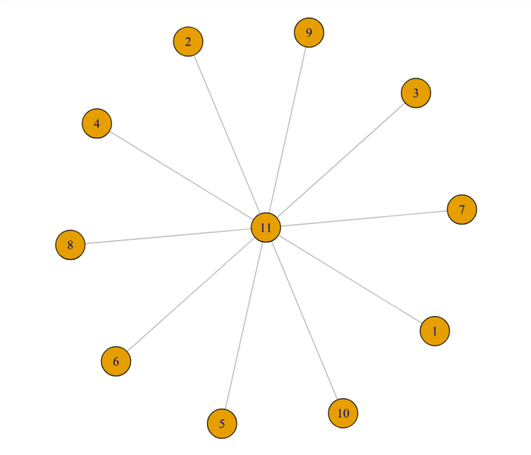
```
## [1] 0.092097
```



sd centralization

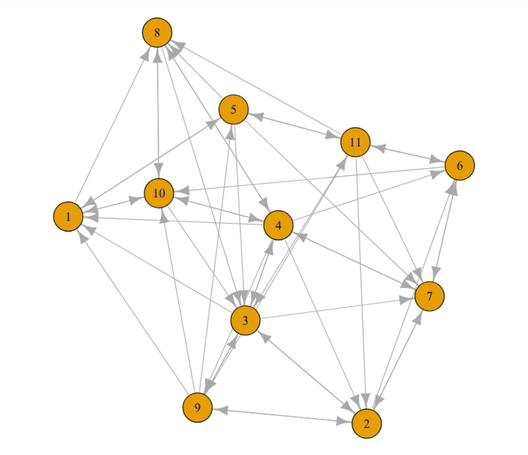
```
snafun::g_centralize(star, measure = "be
```

```
## [1] 13.56801
```




```
snafun::g_centralize(randomnet, measure =
```

```
## [1] 3.28676
```



What to do before the lab of Tuesday

- Finish Tutorials 01, 02, and 03
- Do the homeplay assignments
- Read the literature (See Canvas)