

Lista 2 - Medidas de Centralidade

Redes Complexas para Ciência da Computação

Luben Miguel, Rodrigo Lassance

```
1 # pacotes sendo usados
2 library(igraph)
3 library(tidyverse)
4 library(Matrix)
5 library(knitr)
6 library(scales)
```

Exercício 1

Primeiramente, importamos as seguintes redes:

```
1 # pre-processamento e importacao dos dados
2
3 # dados direcionados
4 # lendo os dados dos tweets sobre israel
5 israel <- read.table("dados/rt_israel.edges", sep="," , header = FALSE) |>
6   select(-3) |>
7   as.matrix()
8
9 # lendo os dados da rede de aeroportos
10 airports <- read.table("dados/inf-openflights.edges", sep=" " , header = FALSE) |>
11   select(-3) |>
12   as.matrix()
13
14 # grafo das paginas web
15 web_hiperlink <- read.table("dados/web-EPA.edges", sep=" " , header = FALSE) |>
16   as.matrix()
17
```

```

18 # grafo dos tweets de israel
19 israel_graph <- israel |>
20   graph_from_edgelist(directed=TRUE)
21
22 israel_adj <- israel_graph |>
23   as_adj()
24
25 airports_graph <- airports |>
26   graph_from_edgelist(directed=TRUE)
27
28 airports_adj <- airports_graph |>
29   as_adj()
30
31 web_graph <- web_hiperlink |>
32   graph_from_edgelist(directed=TRUE)
33
34 web_adj <- web_graph |>
35   as_adj()
36
37 # nao direcionado
38 # dados dos ratos sem processamento
39 ratos_dados <- read.table("dados/mammalia-voles-bhp-trapping.edges", sep=" ", header = FALSE)
40 # rede de rato do mato sem pesos
41 ratos_rede <- ratos_dados |>
42   select(-c(3, 4)) |>
43   group_by(V1, V2) |>
44   summarise(n = n()) |>
45   select(-3) |>
46   as.matrix()
47
48 # por causa do time stamp, algumas conexoes sao repetidas nos dados
49 # corrigindo isso na matriz de adjacencia
50 ratos_adj <- ratos_rede |>
51   graph_from_edgelist(directed=FALSE) |>
52   as_adj()
53
54 ratos_adj <- (ratos_adj >= 1) + 0
55
56 # corrigido isso
57 ratos_graph <- ratos_adj |> graph_from_adjacency_matrix(mode = "undirected")
58
59 # checando se sao simetricas ou nao simetricas

```

```
60 c(isSymmetric(web_adj), isSymmetric(israel_adj), isSymmetric(airports_adj), isSymmetric(ra
```

```
[1] FALSE FALSE FALSE TRUE
```

Para calcular e plotar os graus $P(k)$ e graus de entrada e saída $P(k_{in})$, $P(k_{out})$ elaboramos a seguinte função:

```
1  # escala em loglog2
2  loglog_trans <- trans_new(
3    name = 'log-log',
4    transform = function(x) log(log(x)),
5    inverse = function(x) exp(exp(x)),
6    breaks = extended_breaks(),
7    minor_breaks = regular_minor_breaks(),
8    format = format_format(),
9    domain = c(1, Inf)
10 )
11
12 plot_degree <- function(igraph_obj, rede_label, overall = TRUE){
13   directed <- is_directed(igraph_obj)
14
15   if(!directed){
16     dados_grau <- igraph_obj |>
17     degree() |>
18     as.data.frame() |>
19     rename(grau = "degree(igraph_obj)")
20
21     p1 <- dados_grau |>
22     ggplot(aes(x = grau))+
23     geom_histogram(aes(y = ..density..),
24     colour = "black", fill = "white",
25     alpha = 0.5)+
26     geom_density(colour = "dodgerblue3", size = 1)+
27     theme_minimal()+
28     scale_x_continuous(trans= "log2") +
29     labs(x = "Grau",
30     y = "Densidade")
31     show(p1)
32   }else{
33     if(!overall){
34       out_degree <- igraph_obj |>
```

```

35 degree(mode = "out") |>
36 as.data.frame() |>
37 rename(grau = "degree(igraph_obj, mode = \"out\")") |>
38 mutate(mode = "Out-degree")
39
40   in_degree <- igraph_obj |>
41 degree(mode = "in") |>
42 as.data.frame() |>
43 rename(grau = "degree(igraph_obj, mode = \"in\")") |>
44 mutate(mode = "In-degree")
45
46   dados_grau <- bind_rows(out_degree, in_degree)
47
48 p1 <- dados_grau |>
49 ggplot(aes(x = grau)) +
50   geom_histogram(aes(y = ..density..),
51     colour = "black", fill = "white",
52     alpha = 0.5) +
53   geom_density(colour = "dodgerblue3", size = 1) +
54   facet_wrap(~mode, nrow = 2) +
55   theme_minimal() +
56   scale_x_continuous(trans="log2") +
57   labs(x = "Grau",
58     y = "Densidade",
59     title = rede_label)
60   show(p1)
61   }else{
62     dados_grau <- igraph_obj |>
63     degree(mode = "all") |>
64     as.data.frame() |>
65     rename(grau = "degree(igraph_obj, mode = \"all\")")
66
67     p1 <- dados_grau |>
68     ggplot(aes(x = grau)) +
69       geom_histogram(aes(y = ..density..),
70         colour = "black", fill = "white",
71         alpha = 0.5) +
72       geom_density(colour = "dodgerblue3", size = 1) +
73       theme_minimal() +
74       scale_x_continuous(trans="log2") +
75       labs(x = "Grau",

```

```

76     y = "Densidade",
77     title = rede_label)
78     show(p1)
79   }
80 }
81 return(dados_grau)
82 }

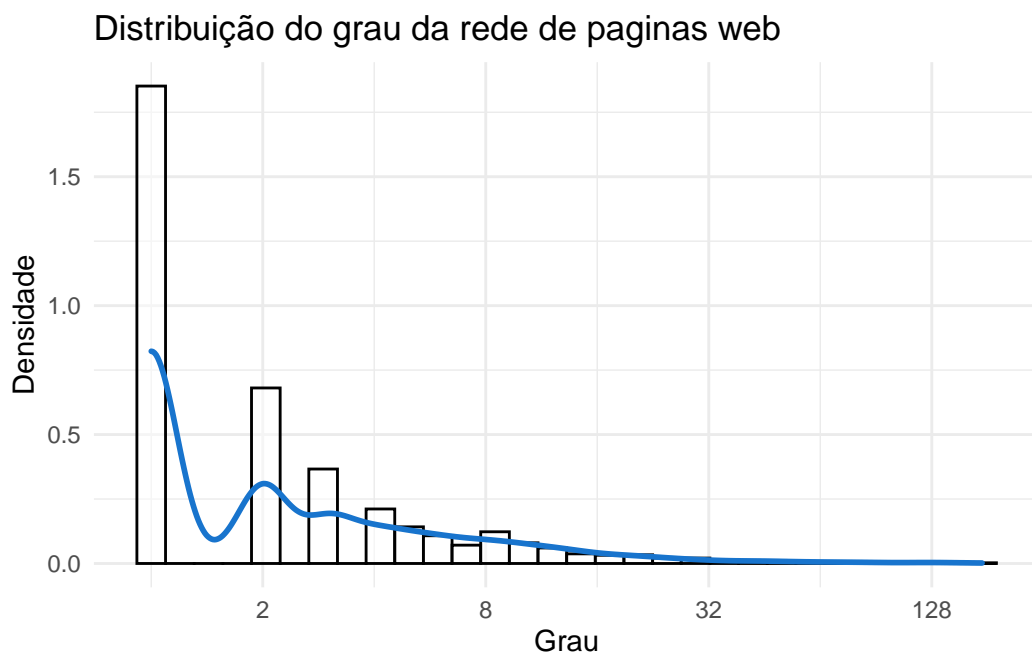
```

Plotamos o grau para todas as redes a seguir: * Rede de páginas web:

```

1 grau_web <- plot_degree(web_graph,
2     rede_label = "Distribuição do grau da rede de paginas web")

```



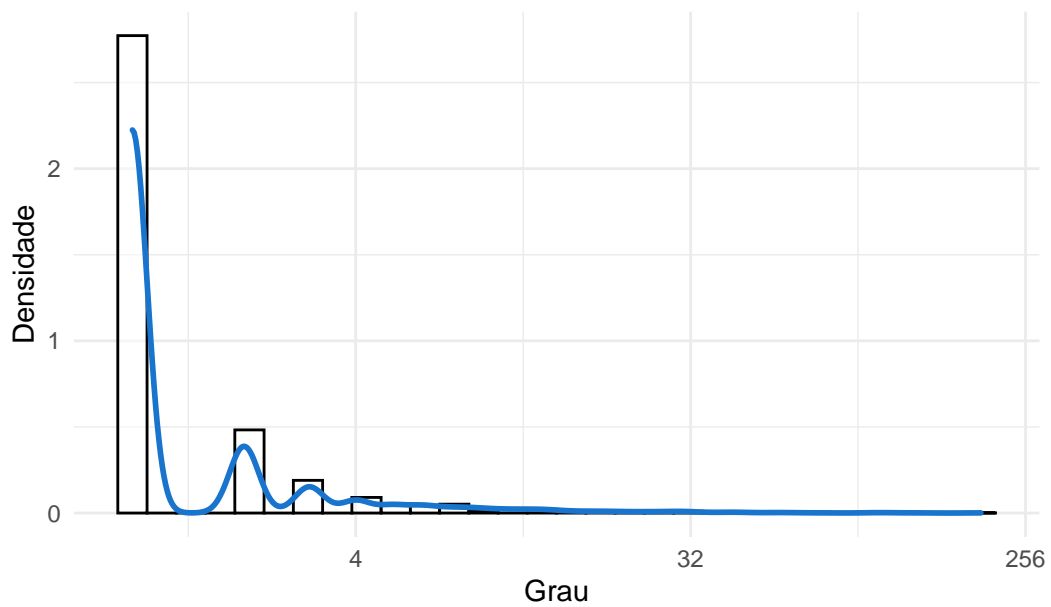
- Rede de retweets sobre Israel

```

1 grau_israel <- plot_degree(israel_graph,
2     rede_label = "Distribuição do grau da rede de Israel")

```

Distribuição do grau da rede de Israel



- Rede de aeroportos

```
1 grau_airports <- plot_degree(airports_graph,  
2 rede_label = "Distribuição do grau da rede de aeroportos")
```

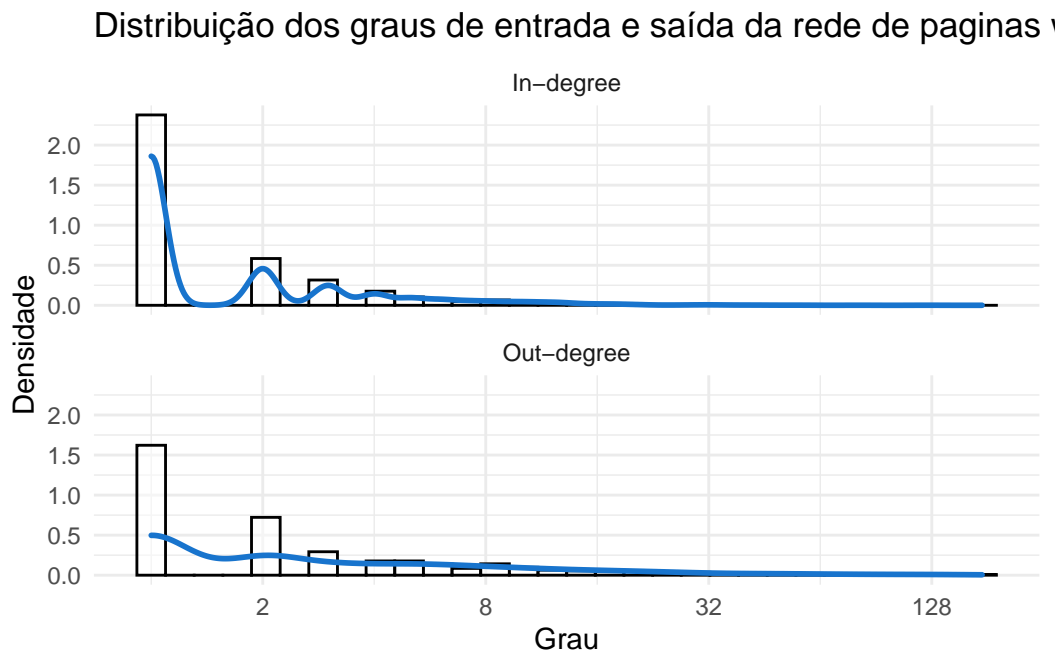
- ```
ratos_grau <- plot_degree(ratos_graph, rede_label = "Distribuição do grau da rede de ratos")
```



Agora, mostramos a distribuição dos graus de entrada e de saída para as redes direcionadas:

\* Rede de páginas web:

```
1 in_out_grau_web <- plot_degree(web_graph,
2 rede_label = "Distribuição dos graus de entrada e saída da rede de páginas web",
3 overall = FALSE)
```

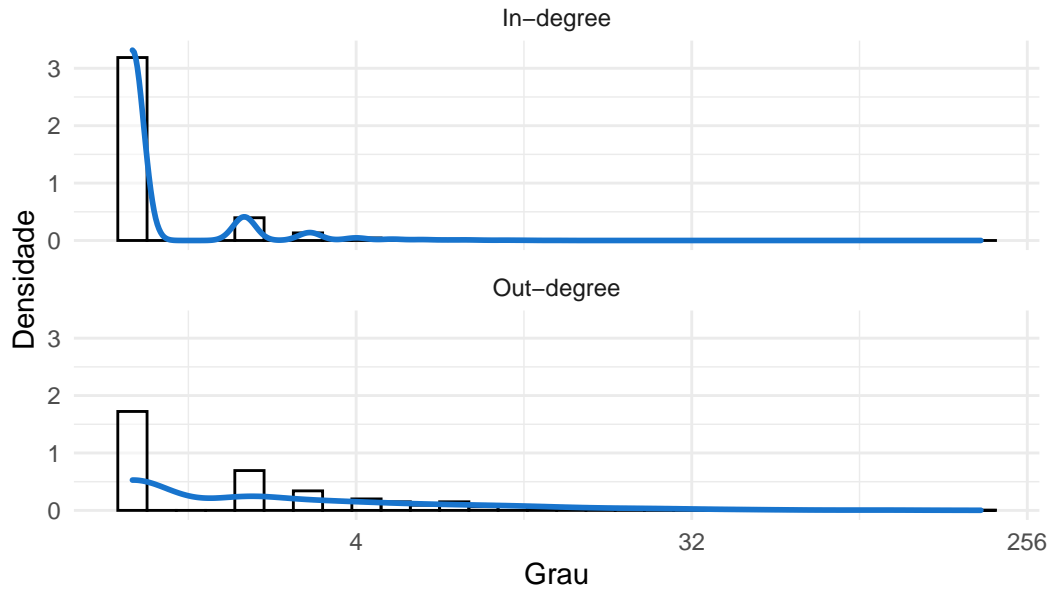


- Rede de retweets sobre Israel

```
1 in_out_grau_israel <- plot_degree(israel_graph,
2 rede_label = "Distribuição dos graus de entrada e saída da rede de Israel",
3 overall = FALSE)
```



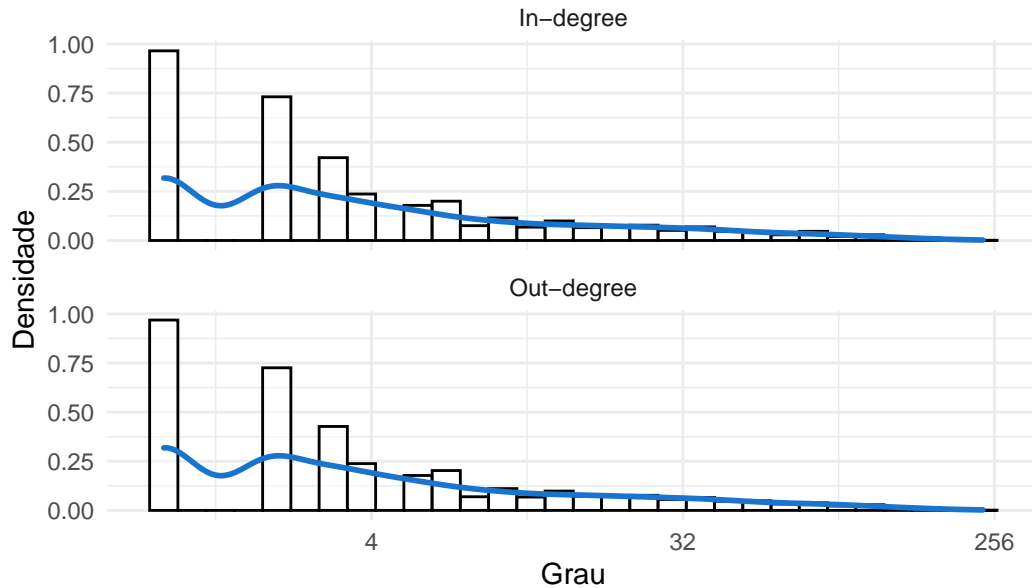
## Distribuição dos graus de entrada e saída da rede de Israel



- Rede de aeroportos

```
1 in_out_grau_airports <- plot_degree(airports_graph,
2 rede_label = "Distribuição dos graus de entrada e saída da rede",
3 overall = FALSE)
```

## Distribuição dos graus de entrada e saída da rede de aeroportos



## Exercício 2

A seguir, calculamos a centralidade de autovetor para cada rede:

```
1 get_eigen_centrality <- function(igraph_obj){
2 if(is_directed(igraph_obj)){
3 eigen_centrality(igraph_obj, directed = TRUE)$vector
4 }else{
5 eigen_centrality(igraph_obj)$vector
6 }
7 }
8
9 # grafo de israel
10 israel_eigen <- israel_graph |> get_eigen_centrality()
11
12 # grafo da web
13 web_eigen <- web_graph |> get_eigen_centrality()
14
15 # grafo dos aeroportos
16 airport_eigen <- airports_graph |> get_eigen_centrality()
17
```

```

18 # grafo dos ratos
19 ratos_eigen <- ratos_graph |> get_eigen_centrality()

```

Com essa centralidade calculada para todas as redes, obtemos a seguir a tabela com as correlações de pearson entre a medida de centralidade de autovetor e o grau:

```

1 eigen_list <- list("israel" = israel_eigen,
2 "web" = web_eigen,
3 "airport" = airport_eigen,
4 "ratos" = ratos_eigen)
5
6 grau_list <- list("israel" = grau_israel$grau,
7 "web" = grau_web$grau,
8 "airport" = grau_airports$grau,
9 "ratos" = ratos_grau$grau)
10
11 map2_dfr(eigen_list, grau_list, function(x, y){
12 cor(x, y)
13 }) |> pivot_longer(1:4, values_to = "valor") |>
14 mutate(valor = round(valor, 3)) |>
15 knitr::kable(format = "latex",
16 booktabs = TRUE,
17 escape = FALSE,
18 col.names = c("Rede",
19 "Valor da correlação"),
20 caption = "Correlação de pearson entre centralidade de grau e autovetor para",
21 kableExtra::kable_styling(full_width = FALSE,
22 latex_options = "hold_position")

```

Tabela 1: Correlação de pearson entre centralidade de grau e autovetor para cada rede

| Rede    | Valor da correlação |
|---------|---------------------|
| israel  | 0.041               |
| web     | 0.084               |
| airport | 0.931               |
| ratos   | 0.469               |

```

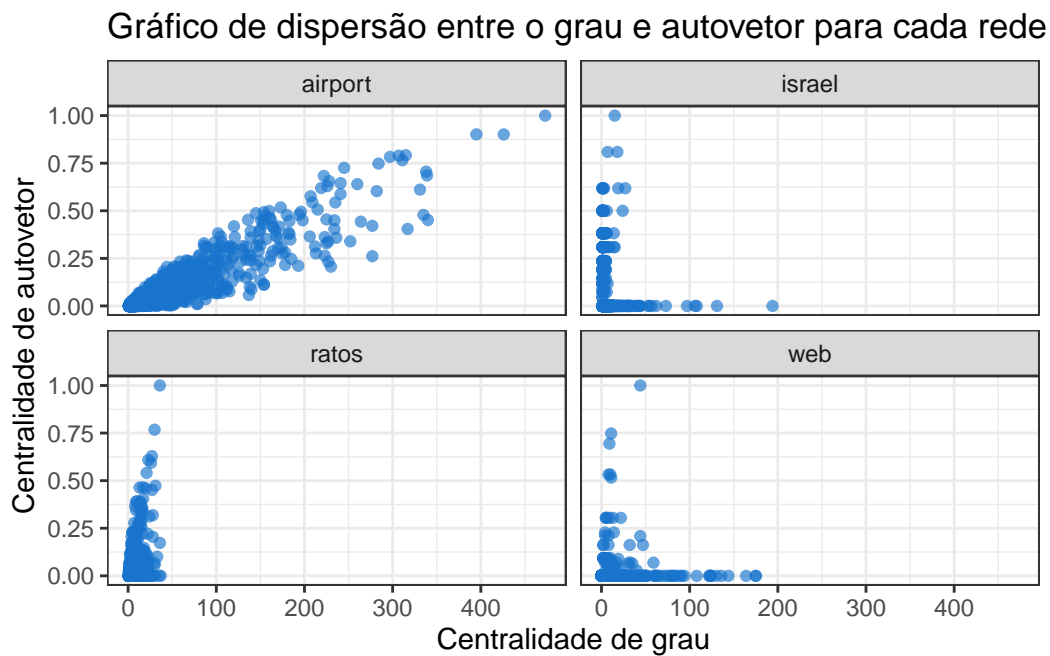
1 eigen_grau_df <- data.frame(name_rede = c(rep("israel", length(israel_eigen)),
2 rep("airport", length(airport_eigen)),
3 rep("ratos", length(ratos_eigen)),

```

```

4 rep("web", length(web_eigen))),
5 eigen = c(israel_eigen, airport_eigen, ratos_eigen,
6 web_eigen),
7 grau = c(grau_israel$grau, grau_airports$grau, ratos_grau$grau,
8 grau_web$grau))
9
10 eigen_grau_df |>
11 ggplot(aes(x = grau, y = eigen))+
12 geom_point(colour = "dodgerblue3", alpha = 0.65)+
13 facet_wrap(~name_rede, nrow = 2, ncol = 2)+
14 theme_bw()+
15 labs(x = "Centralidade de grau",
16 y = "Centralidade de autovetor",
17 title = "Gráfico de dispersão entre o grau e autovetor para cada rede")

```



Vemos principalmente nos scatterplots das redes de ratos, paginas web e retweets de israel muitos valores com centralidade de grau relativamente grande porém uma centralidade de autovetor igual a zero, tendo ambas centralidades uma correlação de pearson baixa para essas redes.

## Exercício 3

Vemos pelo Exercício 2 que para valores elevados de grau se observam valores nulos de centralidade de autovalor. Verificaremos agora seguir quantos vértices tem centralidade de autovetor nula e graus de entrada não nulos para cada rede:

```
1 in_grau_airports <- in_out_grau_airports |> filter(mode == "In-degree") |> pull(grau)
2 in_grau_israel <- in_out_grau_israel |> filter(mode == "In-degree") |> pull(grau)
3 in_grau_web <- in_out_grau_web |> filter(mode == "In-degree") |> pull(grau)
4
5 eigen_grau_df <- data.frame(name_rede = c(rep("israel", length(israel_eigen)),
6 rep("airport", length(airport_eigen)),
7 rep("web", length(web_eigen))),
8 eigen = c(israel_eigen, airport_eigen,
9 web_eigen),
10 grau = c(in_grau_israel, in_grau_airports,
11 in_grau_web))
12
13 # funcao para detectar se existe pelo menos um vertice com centralidade nula
14 detecta_nula <- function(eigen_grau_df, rede){
15 eigen_grau_df |>
16 filter(name_rede == rede) |>
17 filter(eigen == 0 & grau > 0) |>
18 nrow()
19 }
20
21 c(detecta_nula(eigen_grau_df,"israel"),
22 detecta_nula(eigen_grau_df,"airport"),
23 detecta_nula(eigen_grau_df,"web"))
```

```
[1] 2742 18 288
```

Ou seja, vemos que todas as redes direcionadas tem pelo menos um vértice com centralidade de autovetor nula mesmo tendo uma ou mais conexões de entrada.

## Exercício 4

A rede de interação entre ratos do mato possui um peso ignorado que nos diz quantas vezes ambos os ratos conectados caíram na mesma armadilha em alguma timestamp específico. Assim, somaremos todos os pesos associados a cada timestamp e as relações de ida e volta:

```

1 ratos_weighted_graph <- ratos_dados |>
2 group_by(V1, V2) |>
3 select(-4) |>
4 summarise(V3 = sum(V3)) |>
5 rename(weight = "V3") |>
6 graph_from_data_frame(directed = FALSE)

```

`summarise()` has grouped output by 'V1'. You can override using the `.groups` argument.

Para obter a acessibilidade, podemos acessar o peso no grafo e calcular a entropia nos pesos a partir da seguinte função:

```

1 accessibility <- function(igraph_obj){
2 vtx_list <- V(igraph_obj)
3 access <- numeric(length(vtx_list))
4 for(i in 1:length(vtx_list)){
5 pesos <- incident_edges(igraph_obj, vtx_list[i])[[1]]$weight
6 pesos <- pesos/sum(pesos)
7 access[i] <- exp(-sum(pesos*log(pesos)))
8 }
9 access <- setNames(access, vtx_list$name)
10 return(access)
11 }

```

Tendo o vetor de acessibilidade:

```

1 acesso_ratos <- accessibility(ratos_weighted_graph)

```

Desta maneira, podemos obter o gráfico de dispersão entre grau e acessibilidade como a seguir:

```

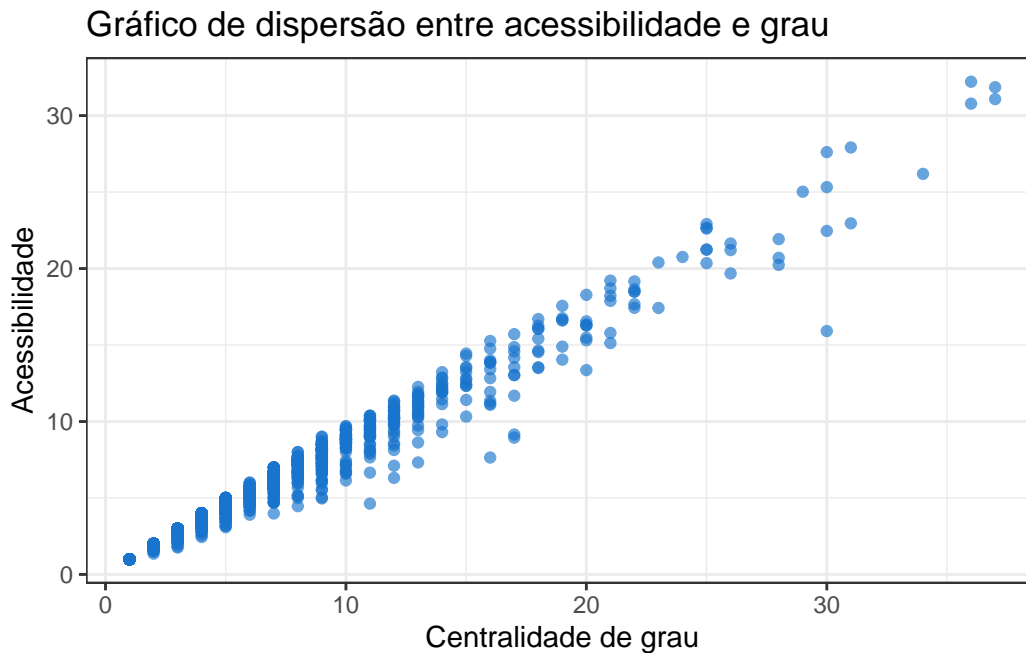
1 grau_acess_data_frame <- data.frame(grau = degree(ratos_weighted_graph),
2 acessib = acesso_ratos) |>
3 rownames_to_column(var = "vertice")
4
5 grau_acess_data_frame |>
6 ggplot(aes(x = grau, y = acessib))+
7 geom_point(colour = "dodgerblue3", alpha = 0.65)+
8 theme_bw()+

```

```

9 labs(x = "Centralidade de grau",
10 y = "Acessibilidade",
11 title = "Gráfico de dispersão entre acessibilidade e grau")

```



Vemos uma tendência linear nesse gráfico, com a acessibilidade tendo em geral valores menores que o grau. Há casos porém, que vemos uma acessibilidade consideravelmente menor que o grau, tendo por exemplo acessibilidades menor que 10 em casos de grau maior que 10.

## Exercício 5

Podemos ver os vértices que tem acessibilidade muito menor que 10 para casos de graus maiores que 10, como os vértices com acessibilidade menor que 8 e grau maior que 10:

```

1 grau_access_data_frame |>
2 filter(grau > 10 & acessib < 8) |>
3 mutate(acessib = round(acessib, 3)) |>
4 knitr::kable(format = "latex",
5 booktabs = TRUE,
6 escape = FALSE,
7 col.names = c("Vértice",

```

```

8 "Grau",
9 "Acessibilidade"),
10 caption = "Grau comparado a acessibilidade para diferentes vértices") |>
11 kableExtra::kable_styling(full_width = FALSE,
12 latex_options = "hold_position")

```

Tabela 2: Grau comparado a acessibilidade para diferentes vértices

| Vértice | Grau | Acessibilidade |
|---------|------|----------------|
| 689     | 11   | 6.657          |
| 812     | 16   | 7.645          |
| 829     | 11   | 4.630          |
| 1277    | 11   | 7.651          |
| 1489    | 12   | 7.107          |
| 1494    | 13   | 7.322          |
| 1575    | 12   | 6.314          |
| 1657    | 11   | 7.887          |

Ou seja, vemos alguns exemplos de vértice que a acessibilidade é bem menor que o grau, destacando principalmente o vértice 829, com acessibilidade 4.630 e grau 11.

## Exercício 6

Tomando novamente a rede de interação entre ratos e armadilhas sem peso, podemos obter a centralidade de Katz, fixando  $\alpha < \frac{1}{k_1}$ . Nesse caso, o maior autovalor  $k_1$  é dado por:

```

1 k_1 <- eigen(ratos_adj) |> pluck("values") |> max()
2 k_1

```

```
[1] 12.12621
```

Ou seja, devemos tomar  $\alpha$  menor que:

```
1 1/k_1
```

```
[1] 0.08246599
```



Assim, tomando  $\alpha = 0.05$ , obtemos a centralidade de Katz através da função do pacote *centiserve*:

```
1 katz <- ratos_graph |> centiserve::katzcent(alpha = 0.05)
```

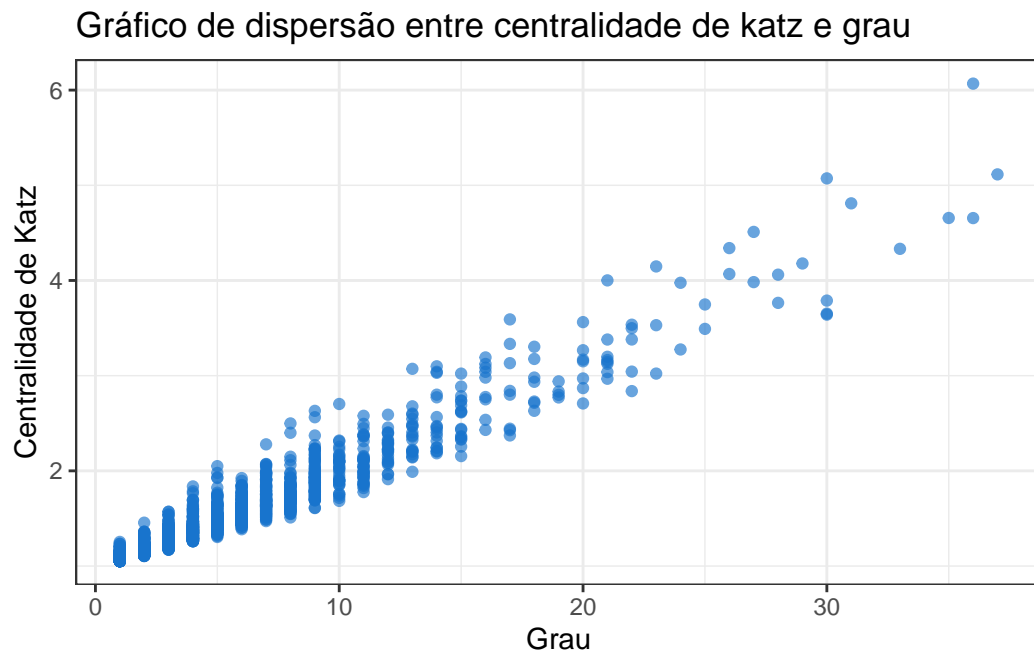
Assim, obtemos agora a correlação de spearman entre a centralidade de katz e o grau:

```
1 cor(katz, ratos_grau$grau, method = "spearman")
```

```
[1] 0.9606899
```

Tendo o gráfico de dispersão entre as centralidades:

```
1 grau_katz_data_frame <- data.frame(grau = ratos_grau$grau,
2 katz = katz) |>
3 rownames_to_column(var = "vertice")
4
5 grau_katz_data_frame |>
6 ggplot(aes(x = grau, y = katz))+
7 geom_point(colour = "dodgerblue3", alpha = 0.65)+
8 theme_bw()+
9 labs(x = "Grau",
10 y = "Centralidade de Katz",
11 title = "Gráfico de dispersão entre centralidade de katz e grau")
```



Ou seja, a centralidade de Katz tem uma relação mais positiva que o grau do que a centralidade de autovetor comum.