Lista 2 - Medidas de Centralidade

Redes Complexas para Ciência da Computação

Luben Miguel, Rodrigo Lassance

```r
# pacotes sendo usados
library(igraph)
library(tidyverse)
library(Matrix)
library(knitr)
library(scales)
```

Para os exercícios 1-3, precisamos importar 3 redes direcionadas e 1 não-direcionada. Entre as redes direcionadas, foram escolhidas a de retweets sobre Israel e a de destinos de aeroportos (rt-israel e inf-openflights, já previamente descritos na lista de exercícios anterior), além de uma rede de direcionamentos de páginas web (wep-EPA). Já a rede não-direcionada foi a de interações entre ratos-do-mato (mammalia-voles-bhp-trapping, com interação sendo definida como existente caso dois ratos tenham sido pegos em uma mesma armadilha pelo menos uma vez). Visto que essa última rede apresenta pesos nas arestas, tais pesos foram inicialmente igualados a 1.

```r
# pre-processamento e importacao dos dados

# dados direcionados
# lendo os dados dos tweets sobre israel
israel <- read.table("dados/rt_israel.edges", sep=",", header = FALSE) |>
  select(-3) |>
  as.matrix()

# lendo os dados da rede de aeroportos
airports <- read.table("dados/inf-openflights.edges", sep=" ", header = FALSE) |>
  select(-3) |>
  as.matrix()

# grafo das paginas web
```

```r
web_hiperlink <- read.table("dados/web-EPA.edges", sep=" ", header = FALSE) |>
  as.matrix()

# grafo dos tweets de israel
israel_graph <- israel |>
  graph_from_edgelist(directed=TRUE)

israel_adj <- israel_graph |>
  as_adj()

airports_graph <- airports |>
  graph_from_edgelist(directed=TRUE)

airports_adj <- airports_graph |>
  as_adj()

web_graph <- web_hiperlink |>
  graph_from_edgelist(directed=TRUE)

web_adj <- web_graph |>
  as_adj()

# nao direcionado
# dados dos ratos sem processamento
ratos_dados <- read.table("dados/mammalia-voles-bhp-trapping.edges",
                          sep=" ", header = FALSE)
# rede de rato do mato sem pesos
ratos_rede <-  ratos_dados |>
  select(-c(3, 4)) |>
  group_by(V1, V2) |>
  summarise(n  = n()) |>
  select(-3) |>
  as.matrix()

# por causa do time stamp, algumas conexoes sao repetidas nos dados
# corrigindo isso na matriz de adjacencias
ratos_adj <- ratos_rede |>
  graph_from_edgelist(directed=FALSE) |>
  as_adj()

ratos_adj <- (ratos_adj >= 1) + 0
```

```r

# corrigido isso
ratos_graph <- ratos_adj |> graph_from_adjacency_matrix(mode = "undirected")

# checando se sao simetricas ou nao simetricas
c(isSymmetric(web_adj), isSymmetric(israel_adj),
  isSymmetric(airports_adj), isSymmetric(ratos_adj))
```

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

Exercício 1

No enunciado, foi pedido para se gerar a distribuição de probabilidade dos graus em escala log-log. Como o eixo Y trata de probabilidades, percebeu-se que tomar a sua escala no logaritmo gerava resultados pouco intuitivos. Desse modo, primando pela facilitação da visualização das informações, optou-se por plotar as frequências absolutas no lugar das relativas. Assim, para calcular e plotar os graus $\mathrm{P}(k)$ e graus de entrada e saída $\mathrm{P}(k_{in})$, $\mathrm{P}(k_{out})$ elaboramos a seguinte função:

```r
plot_degree <- function(igraph_obj, rede_label, overall = TRUE){
  directed <- is_directed(igraph_obj)

  if(!directed){
  dados_grau <- igraph_obj |>
  degree() |>
  as.data.frame() |>
  rename(grau = "degree(igraph_obj)")

  p1 <- dados_grau |>
  ggplot(aes(x = grau))+
  geom_histogram(
  colour = "black", fill = "white",
  alpha = 0.5)+
  theme_minimal()+
  scale_x_continuous(trans= "log2") +
  scale_y_continuous(trans= "log2") +
  labs(x = expression("Grau"),
  y = expression("Frequência"))
  show(p1)
  }else{
```

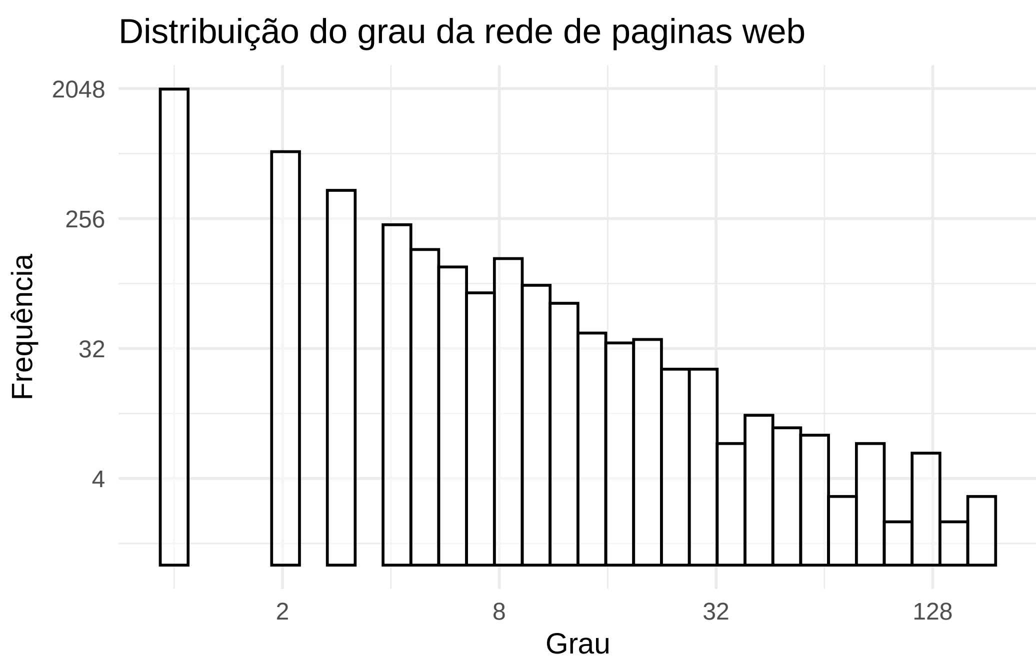
```r
  if(!overall){
  out_degree <- igraph_obj |>
degree(mode = "out") |>
as.data.frame() |>
rename(grau = "degree(igraph_obj, mode = \"out\")") |>
mutate(mode = "Out-degree")

  in_degree <- igraph_obj |>
degree(mode = "in") |>
as.data.frame() |>
rename(grau = "degree(igraph_obj, mode = \"in\")") |>
mutate(mode = "In-degree")

  dados_grau <- bind_rows(out_degree, in_degree)

p1 <- dados_grau |>
ggplot(aes(x = grau))+
geom_histogram(
colour = "black", fill = "white",
alpha = 0.5)+
facet_wrap(~mode, nrow = 2)+
theme_minimal()+
scale_x_continuous(trans= "log2") +
scale_y_continuous(trans= "log2") +
labs(x = expression("Grau"),
y = expression("Frequência"),
title = rede_label)
show(p1)
  }else{
    dados_grau <- igraph_obj |>
degree(mode = "all") |>
as.data.frame() |>
rename(grau = "degree(igraph_obj, mode = \"all\")")

p1 <- dados_grau |>
ggplot(aes(x = grau))+
geom_histogram(
colour = "black", fill = "white",
alpha = 0.5)+
theme_minimal()+
scale_x_continuous(trans= "log2") +
```

```
    scale_y_continuous(trans= "log2") +
    labs(x = expression("Grau"),
    y = expression("Frequência"),
    title = rede_label)
    show(p1)
      }
    }
    return(dados_grau)
}
```

Plotamos o grau para todas as redes a seguir:

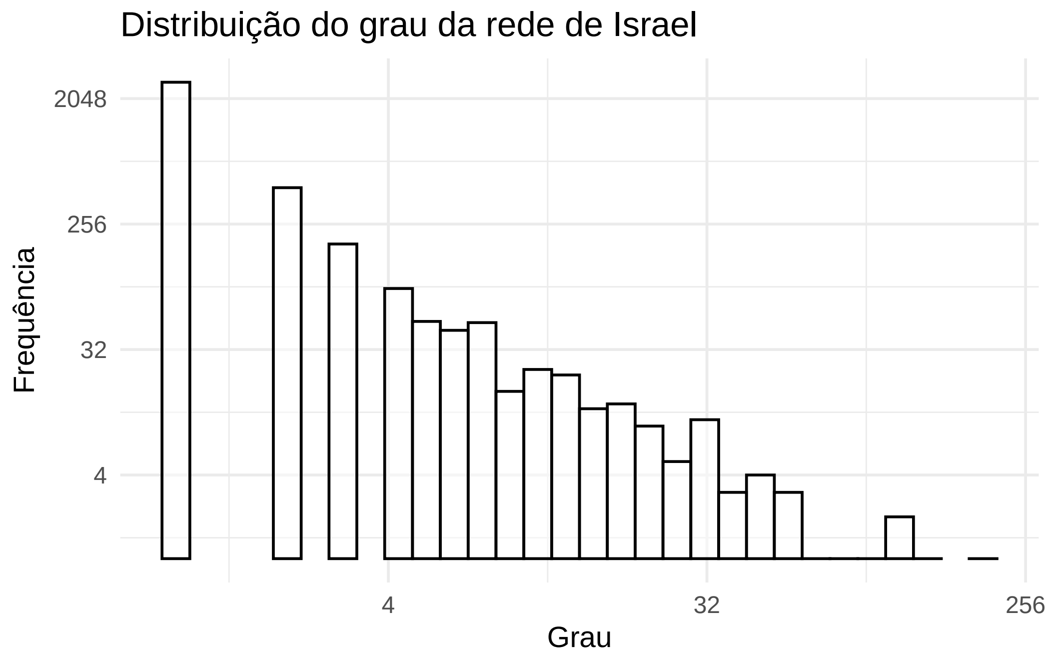
- Rede de páginas web:

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



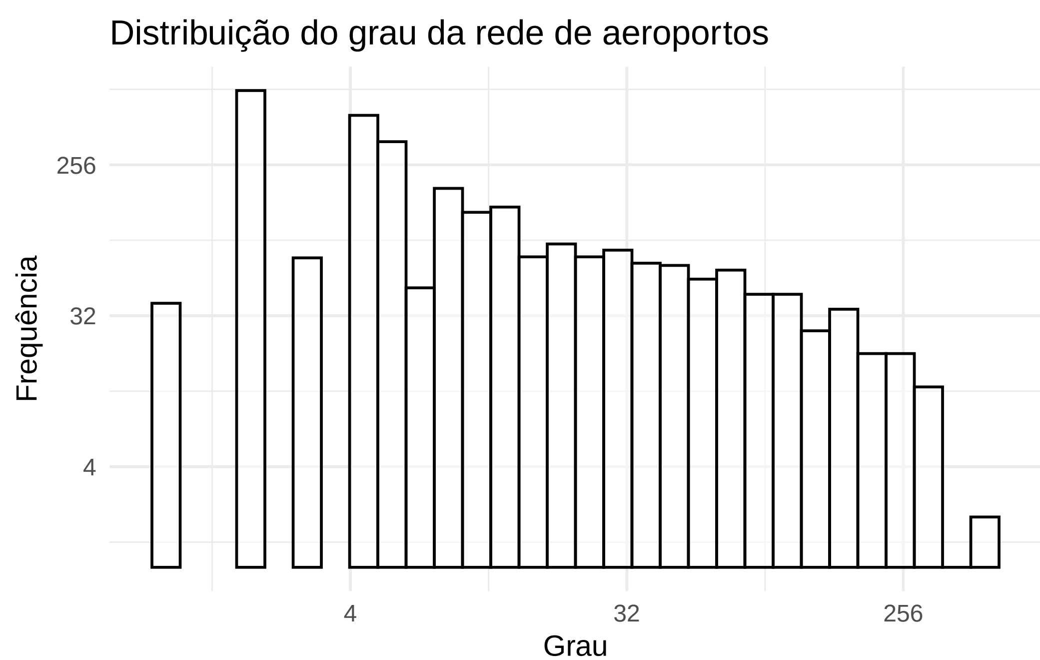
- Rede de retweets sobre Israel

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



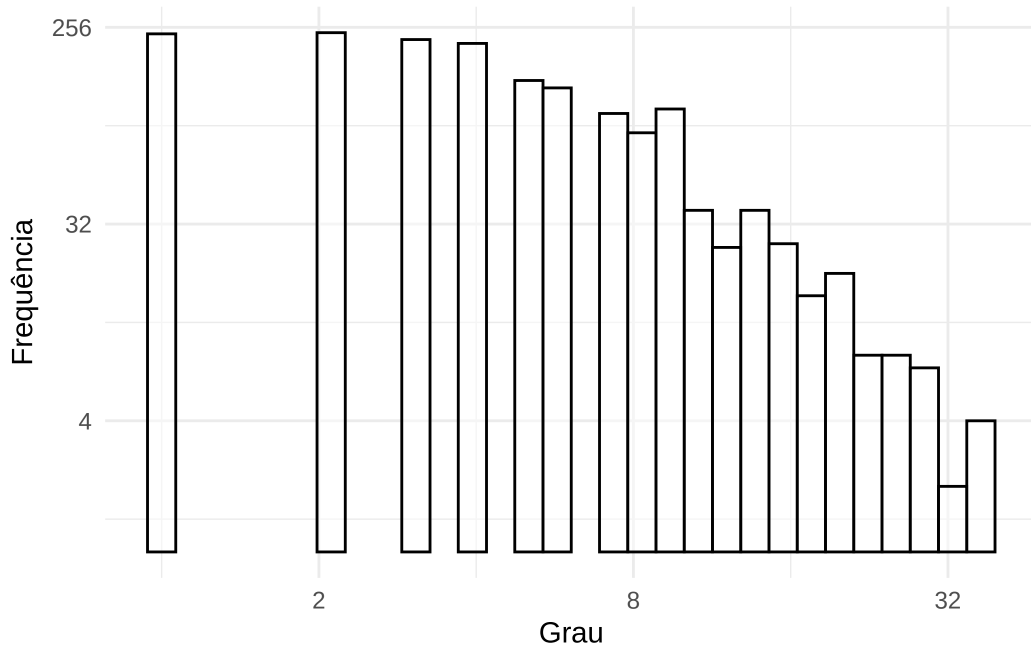
- Rede de aeroportos

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



- Rede de ratos

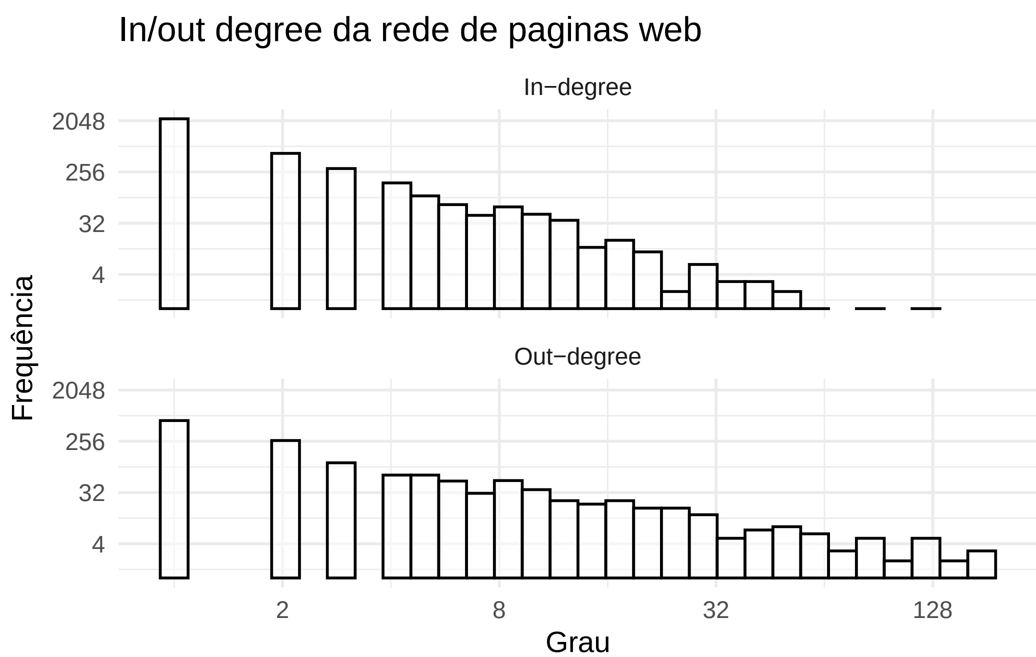
```
ratos_grau <- plot_degree(ratos_graph,
                          rede_label = "Distribuição do grau da rede de ratos")
```



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

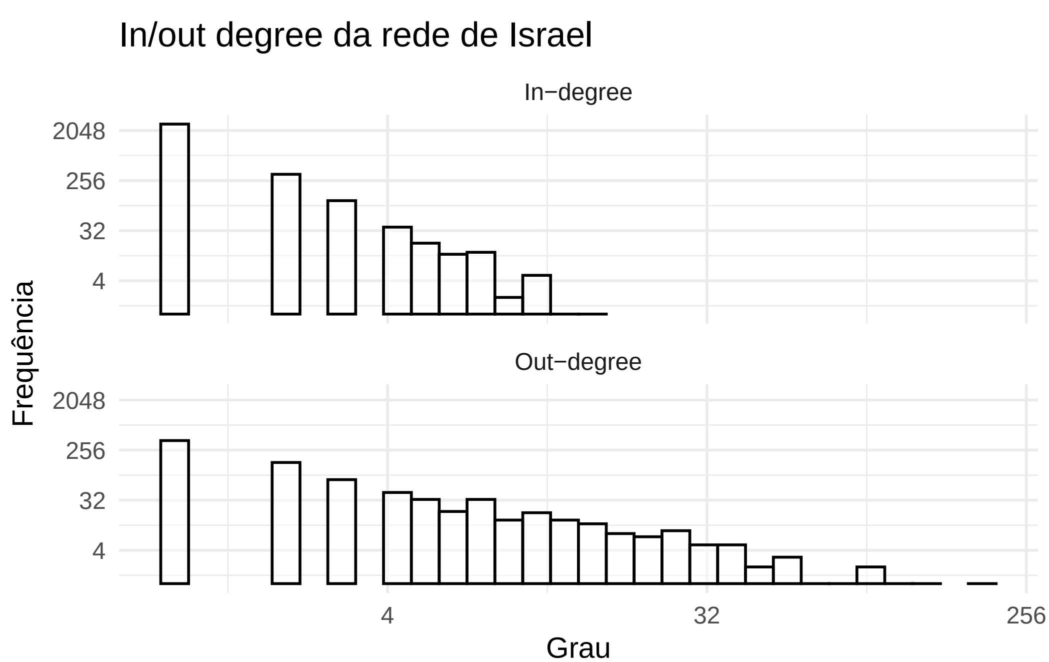
- Rede de páginas web:

```
in_out_grau_web <- plot_degree(web_graph,
                               rede_label = "In/out degree da rede de paginas web",
                               overall = FALSE)
```



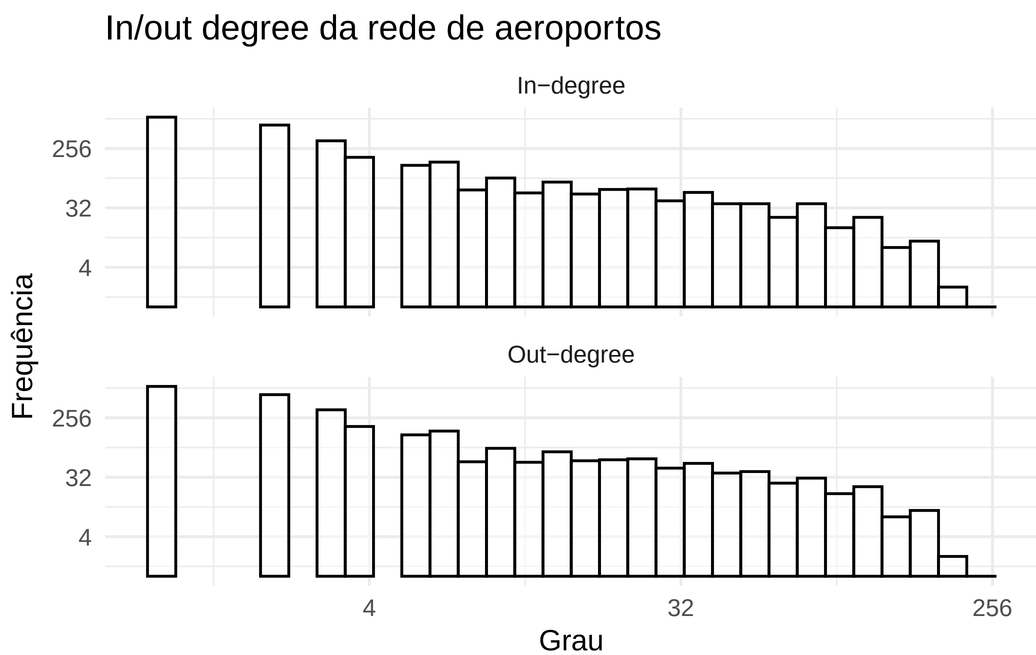
- Rede de retweets sobre Israel

```r
in_out_grau_israel <- plot_degree(israel_graph,
                                  rede_label = "In/out degree da rede de Israel",
                                  overall = FALSE)
```



- Rede de aeroportos

```r
in_out_grau_airports <- plot_degree(airports_graph,
                                    rede_label = "In/out degree da rede de aeroportos",
                                    overall = FALSE)
```



Exercício 2

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

```r
get_eigen_centrality <- function(igraph_obj){
  if(is_directed(igraph_obj)){
    eigen_centrality(igraph_obj, directed = TRUE)$vector
  }else{
    eigen_centrality(igraph_obj)$vector
  }
}

# grafo de israel
israel_eigen <- israel_graph |> get_eigen_centrality()

```

```r
# grafo da web
web_eigen <- web_graph |> get_eigen_centrality()

# grafo dos aeroportos
airport_eigen <- airports_graph |> get_eigen_centrality()

# grafo dos ratos
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:

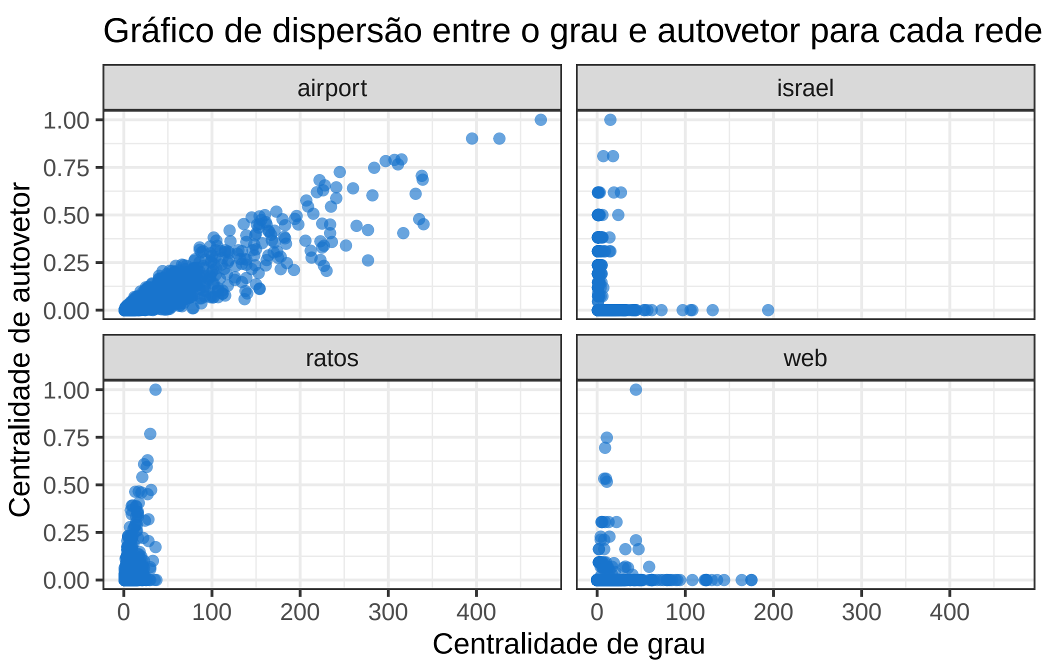
```r
eigen_list <- list("israel" = israel_eigen,
                   "web" = web_eigen,
                   "airport" = airport_eigen,
                   "ratos" = ratos_eigen)

grau_list <- list("israel" = grau_israel$grau,
                  "web" = grau_web$grau,
                  "airport" = grau_airports$grau,
                  "ratos" = ratos_grau$grau)

map2_dfr(eigen_list, grau_list, function(x, y){
  cor(x, y)
}) |> pivot_longer(1:4, values_to = "valor") |>
  mutate(valor = round(valor, 3)) |>
  knitr::kable(format = "latex",
               booktabs = TRUE,
               escape = FALSE,
               col.names = c("Rede",
                             "Valor da correlação"),
               caption = "Correlação de Pearson entre centralidade de grau e autovetor") |
  kableExtra::kable_styling(full_width = FALSE,
                            latex_options = "hold_position")
```

```r
eigen_grau_df <- data.frame(name_rede = c(rep("israel", length(israel_eigen)),
                                          rep("airport", length(airport_eigen)),
                                          rep("ratos", length(ratos_eigen)),
                                         rep("web", length(web_eigen))),
                            eigen = c(israel_eigen, airport_eigen, ratos_eigen,
                                      web_eigen),
```

Tabela 1: Correlação de Pearson entre centralidade de grau e autovetor

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

```
                         grau = c(grau_israel$grau, grau_airports$grau, ratos_grau$grau,
                                  grau_web$grau))

eigen_grau_df |>
  ggplot(aes(x = grau, y = eigen))+
  geom_point(colour = "dodgerblue3", alpha = 0.65)+
  facet_wrap(~name_rede, nrow = 2, ncol = 2)+
  theme_bw()+
  labs(x = "Centralidade de grau",
       y = "Centralidade de autovetor",
       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. Isso se dá pelo fato de a correlação de Pearson ser uma medida que avalia a relação linear entre duas variáveis. Para os casos das redes de israel e web, há evidências de uma relação fortemente não-linear entre elas.

Exercício 3

Vemos pelo Exercício 2 que para valores elevados de grau se observam valores nulos de centralidade de autovalor. Verificaremos a seguir quantos vértices têm centralidade de autovetor nula e graus de entrada não nulos para cada rede. De modo a evitar erros de aproximação, verificamos quais valores do autovalor são inferiores ou iguais ao erro da máquina (`.Machine$double.eps`).

```r
in_grau_airports <- in_out_grau_airports |> filter(mode == "In-degree") |> pull(grau)
in_grau_israel <- in_out_grau_israel |> filter(mode == "In-degree") |> pull(grau)
in_grau_web <- in_out_grau_web |> filter(mode == "In-degree") |> pull(grau)

eigen_grau_df <- data.frame(name_rede = c(rep("israel", length(israel_eigen)),
                                      rep("airport", length(airport_eigen)),
                                     rep("web", length(web_eigen))),
                       eigen = c(israel_eigen, airport_eigen,
                                 web_eigen),
                       grau = c(in_grau_israel, in_grau_airports,
                                in_grau_web))

# funcao para detectar se existe pelo menos um vertice com centralidade nula
detecta_nula <- function(eigen_grau_df, rede
                         ){
  eigen_grau_df |>
    filter(name_rede == rede) |>
    filter(eigen <= .Machine$double.eps & grau >= 1) |>
    nrow()
}
c(detecta_nula(eigen_grau_df,"israel"),
detecta_nula(eigen_grau_df,"airport"),
detecta_nula(eigen_grau_df,"web"))
```

```
[1] 3039   34 3570
```

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 cairam na mesma armadilha em alguma timestamp específico. Assim, somaremos todos os pesos associados a cada timestamp e as relações de ida e volta:

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

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

```
accessibility <- function(igraph_obj){
  vtx_list <- V(igraph_obj)
  access <- numeric(length(vtx_list))
  for(i in 1:length(vtx_list)){
    pesos <- incident_edges(igraph_obj, vtx_list[i])[[1]]$weight
    pesos <- pesos/sum(pesos)
    access[i] <- exp(-sum(pesos*log(pesos)))
  }
  access <- setNames(access, vtx_list$name)
  return(access)
}
```

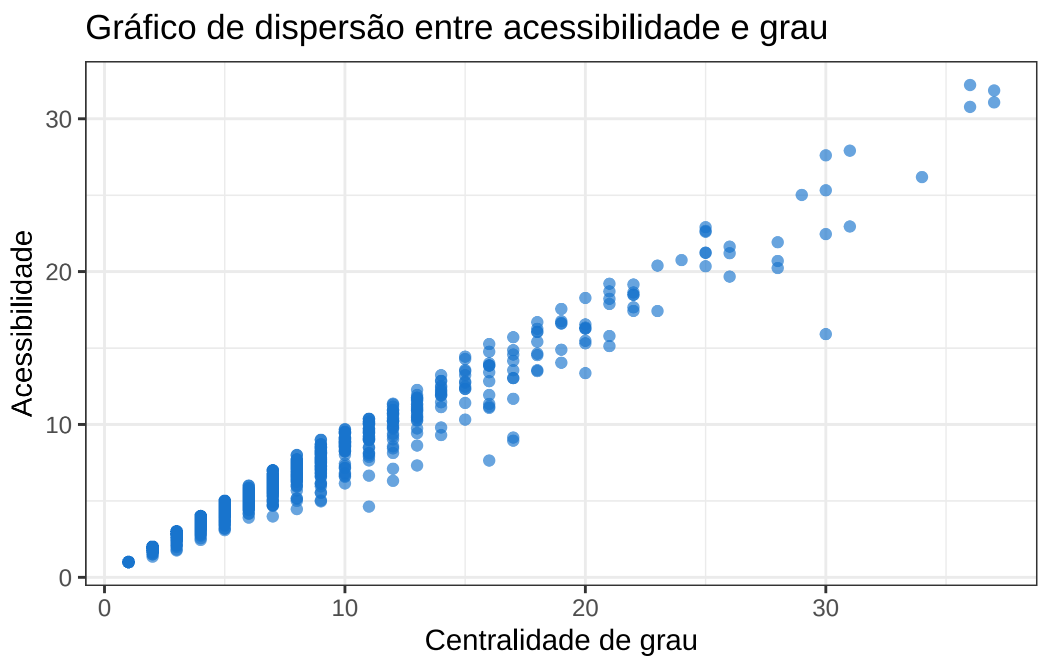
Tendo o vetor de acessibilidade:

```
acesso_ratos <- accessibility(ratos_weighted_graph)
```

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

```
grau_acess_data_frame <- data.frame(grau = degree(ratos_weighted_graph),
                                    acessib = acesso_ratos) |>
```

```
  rownames_to_column(var = "vertice")

grau_acess_data_frame |>
  ggplot(aes(x = grau, y = acessib))+
  geom_point(colour = "dodgerblue3", alpha = 0.65)+
  theme_bw()+
  labs(x = "Centralidade de grau",
       y = "Acessibilidade",
       title = "Gráfico de dispersão entre acessibilidade e grau")
```



Vemos uma tendência linear nesse gŕafico, 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:

```r
grau_acess_data_frame |>
  filter(grau > 10 & acessib < 8) |>
  mutate(acessib = round(acessib, 3)) |>
  knitr::kable(format = "latex",
               booktabs = TRUE,
               escape = FALSE,
               col.names = c("Vértice",
                             "Grau",
                             "Acessibilidade"),
               caption = "Grau comparado a acessibilidade para diferentes vértices") |>
  kableExtra::kable_styling(full_width = FALSE,
                            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:

```r
k_1 <- eigen(ratos_adj) |> pluck("values") |> max()
k_1
```

```
[1] 12.12621
```

Ou seja, devemos tomar α menor que:

```r
1/k_1
```

```
[1] 0.08246599
```

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

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

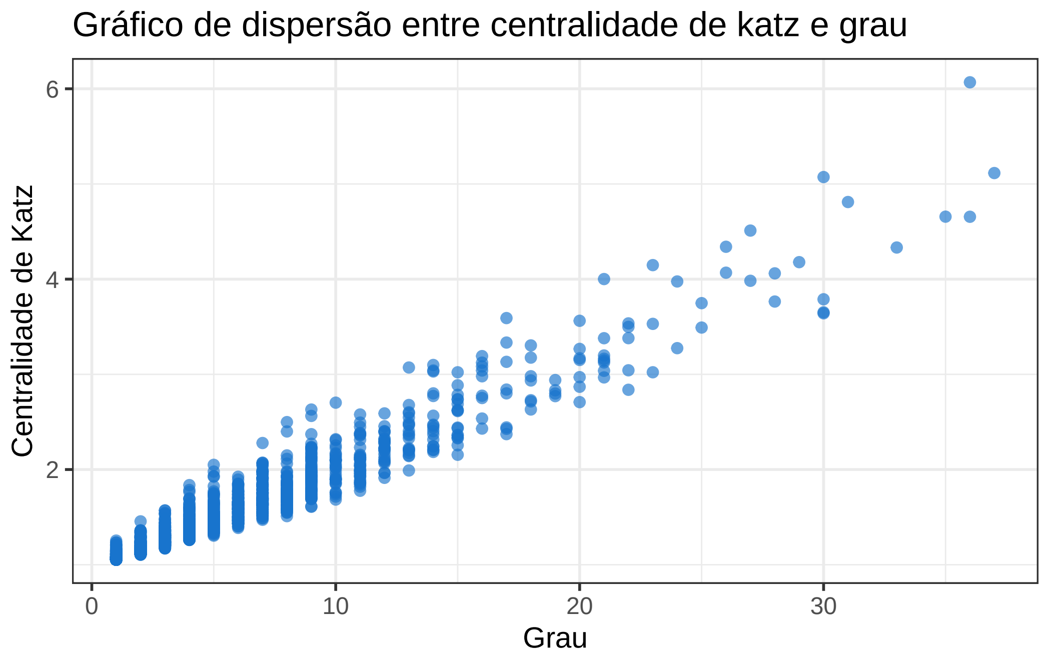
Assim, obtemos agora a correlação de spearman entre a centralidade de katz e o grau:

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

```
[1] 0.9606899
```

Tendo o gráfico de dispersão entre as centralidades:

```r
grau_katz_data_frame <- data.frame(grau = ratos_grau$grau,
                                   katz = katz) |>
  rownames_to_column(var = "vertice")

grau_katz_data_frame |>
  ggplot(aes(x = grau, y = katz))+
  geom_point(colour = "dodgerblue3", alpha = 0.65)+
  theme_bw()+
  labs(x = "Grau",
       y = "Centralidade de Katz",
       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.