Análise Textual - Notas Taquigráficas da CPI da Pandemia

Autor: Alisson Soares

26 de fevereiro de 2022

1 Introdução

O texto a seguir se trata de alguns exemplos do que é possível fazer com técnicas de análise textual no R. Não se trata, portanto, de análise empírica dos dados: no caso, da CPI da Pandemia, a partir das notas taquigráficas. O intuito aqui é fornecer ideias de uso de algumas das ferramentas disponíveis para análise textual, para análises posteriores mais apuradas por quem se interessar pelo tema. Utilizamos aqui a base de dados feita a partir das notas taquigráficas das reuniões da CPI da pandemia, ocorrida no ano de 2021. Se quiser, confira o readme (https://github.com/SoaresAlisson/NotasTaquigraficas).

A primeira coisa a se fazer é colocar como opção global de nosso projeto que strings não sejam consideradas como fatores.

2 Ajustes iniciais

```
# Opções globais
options(stringsAsFactors = FALSE)
# carregando os pacotes necessários inicialmente
library(dplyr) # para manipular os dados
library(magrittr) # para usar o pipe "%>%"
library(ggplot2) # Para gerar gráficos
library(patchwork) # para gerar gráficos múltiplos
library(quanteda) # para análise textual
library(quanteda.textplots) # para gráficos textuais
library(widyr)
library(rvest) # para web scraping
```

Vamos importar os dados. Previamente eu fiz a raspagem de dados das Notas Taquigráficas na página da CPI da Pandemia no site do Senado e estruturei em um dataframe no formato .rds do R: Todas as notas da CPI da Pandemia (https://github.com/SoaresAlisson/NotasTaquigraficas/raw/master /rds/NT_todas_normalizado.rds).

Opção 1: baixando a base de dados direto do site

```
NotasTaq <- readRDS(url("https://github.com/SoaresAlisson/NotasTaquigraficas/raw/ma
ster/rds/NT_todas_normalizado.rds"))
```

Opção 2: Caso já tenha baixado a base de dados (https://github.com/SoaresAlisson/NotasTaquigraficas /raw/master/rds/NT_todas_normalizado.rds) em seu computador, vamos carregar dali usando, por exemplo, o seguinte comando.

```
NotasTaq <- readRDS("~/Documentos/Programação/R/NotasTaquigraficas/NT_todas_normali
zado.rds")
```

Vamos para as análises iniciais:

```
# vamos transformar nosso dataframe em tibble caso ainda não o seja
NotasTaq <- as_tibble(NotasTaq)
# conferindo se é tibble
class(NotasTaq)
## [1] "tbl_df"     "tbl"        "data.frame"
# "tbl_df" indica que é tibble

# observando a estrutura
str(NotasTaq)
## tibble [94,272 × 9] (S3: tbl_df/tbl/data.frame)
##  $ reuniao        : num [1:94272] 1 1 1 1 1 1 1 1 1 ...
##  $ data           : Date[1:94272], format: "2021-04-27" "2021-04-27" ...
##  $ nome           : chr [1:94272] "Otto Alencar" "Ciro Nogueira" "Otto Alencar"
"Ciro Nogueira" ...
##  $ funcao_blocoPar: chr [1:94272] "PRESIDENTE" "(Bloco Parlamentar Unidos pelo B
rasil/PP - PI. Para questão de ordem.)" "PRESIDENTE" "(Bloco Parlamentar Unidos pel
o Brasil/PP - PI)" ...
##  $ BlocoParl      : chr [1:94272] "(Otto Alencar. PSD - BA. Fala da Presidênci
a.)" "Bloco Parlamentar Unidos pelo Brasil" "(Otto Alencar. PSD - BA)" "Bloco Parla
mentar Unidos pelo Brasil" ...
##  $ partido        : Named chr [1:94272] "PSD" "PP" "PSD" "PP" ...
##   ..- attr(*, "names")= chr [1:94272] "PSD" "PP" "PSD" "PP" ...
##  $ estado         : chr [1:94272] "BA" "PI" "BA" "PI" ...
##  $ complemento    : chr [1:94272] "Fala da Presidência" "Para questão de ordem"
"" "" ...
##  $ fala           : chr [1:94272] "Invocando a proteção de Deus, declaro aberta
a sessão para eleição, já que temos quórum suficiente para a abert"| __truncated__
"Senhor Presidente, Senhoras e Senhores Senadores, eu achava que nós deveríamos sus
pender a atual sessão até que"| __truncated__ "Senador Ciro Nogueira, esta é uma Co
missão Parlamentar de Inquérito, Vossa Excelência sabe que não é temática. "| __tru
ncated__ "Senhor Presidente, não é o caso de indeferir ou não. Isso aqui... " ...
```

Se quisermos dar uma olhada na tabela completa em uma nova janela, usamos o comando:

```
View(NotasTaq)
```

Caso queira buscar em toda base de dados:

3 Explorando os dados

Vamos observar os nomes das pessoas que falaram na CPI:

```r
# Obtendo os nomes de todos que participaram
participantes <- NotasTaq$nome |> unique()

# contando os total de participantes que falaram na CPI
length(participantes)
## [1] 154

# vendo os nomes
participantes
##   [1] "Otto Alencar"
##   [2] "Ciro Nogueira"
##   [3] "Jorginho Mello"
##   [4] "Izalci Lucas"
##   [5] "Alessandro Vieira"
##   [6] "Eduardo Braga"
##   [7] "Eduardo Girão"
##   [8] "Marcos Rogério"
##   [9] "Omar Aziz"
##  [10] "Humberto Costa"
##  [11] "Rogério Carvalho"
##  [12] "Weverton"
##  [13] "Eliziane Gama"
##  [14] "Randolfe Rodrigues"
##  [15] "Paulo Rocha"
##  [16] "Flávio Bolsonaro"
##  [17] "Renan Calheiros"
##  [18] "Fernando Bezerra Coelho"
##  [19] "Luis Carlos Heinze"
##  [20] "Angelo Coronel"
##  [21] "Eduardo Pazuello"
##  [22] "Marcos do Val"
##  [23] "Simone Tebet"
##  [24] "Leila Barros"
##  [25] "Tasso Jereissati"
##  [26] "Zenaide Maia"
##  [27] "Fabiano Contarato"
##  [28] "Gen. Eduardo Pazuello"
##  [29] "Vanderlan Cardoso"
##  [30] "Telmário Mota"
##  [31] "Soraya Thronicke"
##  [32] "Jean Paul Prates"
##  [33] "Mara Gabrilli"
##  [34] "Mayra Pinheiro"
##  [35] "Dimas Tadeu Covas"
##  [36] "Nise Hitomi Yamaguchi"
##  [37] "Luana Araújo"
##  [38] "Reguffe"
##  [39] "Marcelo Antônio Cartaxo Queiroga Lopes"
##  [40] "Mecias de Jesus"
##  [41] "Roberto Rocha"
##  [42] "Antônio Elcio Franco Filho"
##  [43] "Natalia Pasternak"
##  [44] "Cláudio Maierovitch"
##  [45] "Kátia Abreu"
```

```
##  [46] "Daniella Ribeiro"
##  [47] "Jorge Kajuru"
##  [48] "Marcellus José Barroso Campêlo"
##  [49] "Marcellus Campelo"
##  [50] "Wilson Witzel"
##  [51] "Carlos Portinho"
##  [52] "Ricardo Ariel Zimerman"
##  [53] "Francisco Eduardo Cardoso Alves"
##  [54] "Styvenson Valentim"
##  [55] "Nelsinho Trad"
##  [56] "Giordano"
##  [57] "Osmar Terra"
##  [58] "Jurema Werneck"
##  [59] "Pedro Hallal"
##  [60] "Luis Miranda"
##  [61] "Luis Ricardo Fernandes Miranda"
##  [62] "Fausto Vieira dos Santos Junior"
##  [63] "Wagner Lima da Costa"
##  [64] "Gina Moraes de Almeida"
##  [65] "Carlos Roberto Wizard Martins"
##  [66] "Alberto Zacharias Toron"
##  [67] "Guilherme Cremonesi Caurin"
##  [68] "Luiz Henrique Mandetta"
##  [69] "Rodrigo Cunha"
##  [70] "Luiz Paulo Dominguetti Pereira"
##  [71] "Flavio Correa de Moraes"
##  [72] "Regina Célia Silva Oliveira"
##  [73] "Pedro Henrique Medeiros de Araújo"
##  [74] "Roberto Ferreira dias"
##  [75] "Maria Jamile José"
##  [76] "Francieli Fontana Sutile Tardetti Fantinato"
##  [77] "Francieli Fontana Sutile Fantinato"
##  [78] "Thiago Leônidas"
##  [79] "William Amorim Santana"
##  [80] "Eliana Maria dias Santiago"
##  [81] "Emanuela Batista de Souza Medrades"
##  [82] "Ticiano Figueiredo de Oliveira"
##  [83] "Pedro Ivo Velloso"
##  [84] "Cristiano Alberto Hossri Carvalho"
##  [85] "Fábio Henrique Ming Martini"
##  [86] "Amilton Gomes de Paula"
##  [87] "Otávio de Queiroga"
##  [88] "Daniel Sampaio"
##  [89] "Eliane Nogueira"
##  [90] "Marcelo Blanco da Costa"
##  [91] "Reinhold Stephanes Junior"
##  [92] "Marcelo Blanco"
##  [93] "Eric Furtado Ferreira Borges"
##  [94] "Nelson Luiz Sperle Teich"
##  [95] "Airton Antonio Soligo"
##  [96] "Emerson Paxá Pinto Oliveira"
##  [97] "Helcio Bruno de Almeida"
##  [98] "João Carlos Gonçalves Krakauer Maia"
##  [99] "Jailton Batista"
## [100] "Ricardo Barros"
```

```
## [101] "Alexandre Figueiredo Costa Silva Marques"
## [102] "Savio de Faria Caram Zuquim"
## [103] "Eduardo de Vilhena Toledo"
## [104] "Túlio Silveira"
## [105] "Francisco Emerson Maximiano"
## [106] "Ticiano Figueiredo"
## [107] "Emanuel Ramalho Catori"
## [108] "Michel Saliba Oliveira"
## [109] "Roberto Pereira Ramos Júnior"
## [110] "Alexandre Queiroz"
## [111] "José Ricardo Santana"
## [112] "Marcelo Queiroga"
## [113] "Ivanildo Gonçalves da Silva"
## [114] "Alan Diniz Moreira Guedes de Ornelas"
## [115] "Francisco Araújo Filho"
## [116] "Cleber Lopes de Oliveira"
## [117] "Marcos Tolentino da Silva"
## [118] "Luciano Duarte Peres"
## [119] "Marconny Nunes Ribeiro Albernaz de Faria"
## [120] "Wagner de Campos Rosário"
## [121] "Felipe Dantas de Araujo"
## [122] "Pedro Benedito Batista Júnior"
## [123] "Aristides Zacarelli"
## [124] "Maria José Ferreira Pessoa"
## [125] "Vinicius Luiz Ferreira"
## [126] "Danilo Berndt Trento"
## [127] "Bruna Mendes dos Santos Morato"
## [128] "Antonio Barra Torres"
## [129] "Rose de Freitas"
## [130] "Beno Brandão"
## [131] "Luciano Hang"
## [132] "Daniel Freitas"
## [133] "Bia Kicis"
## [134] "Otávio Oscar Fakhoury"
## [135] "Antonio Manssur"
## [136] "Milena Ramos Câmara"
## [137] "Raimundo Nonato Brasil"
## [138] "Andreia da Silva Lima"
## [139] "Walter José Faiad de Moura"
## [140] "Paulo Roberto Vanderlei Rebello Filho"
## [141] "Walter Correa de Souza Neto"
## [142] "Tadeu Frederico de Andrade"
## [143] "Priscila Pamela Cesario dos Santos"
## [144] "Katia Shirlene Castilho dos Santos"
## [145] "Arquivaldo Bites Leão Leite"
## [146] "Rosane Maria dos Santos Brandão"
## [147] "Mayra Pires Lima"
## [148] "Antonio Carlos Alves de Sá Costa"
## [149] "Giovanna Gomes Mendes da Silva"
## [150] "Márcio Antonio do Nascimento Silva"
## [151] "Elton da Silva Chaves"
## [152] "Fabio Wajngarten"
## [153] "Carlos Murillo"
## [154] "Ernesto Araújo"
```

Aparecem 154 nomes diferentes, mas repare que alguns destes são uma mesma pessoa, porém com mais de uma grafia, como o ex-ministro da saúde Eduardo Pazuello, que aparece uma com e sem “gen.”, bem como Francieli, o deputado Luis Miranda e Marcelo Blanco, também aparecem com nomes grafado de duas formas distintas. Vamos ajustar estes nomes.

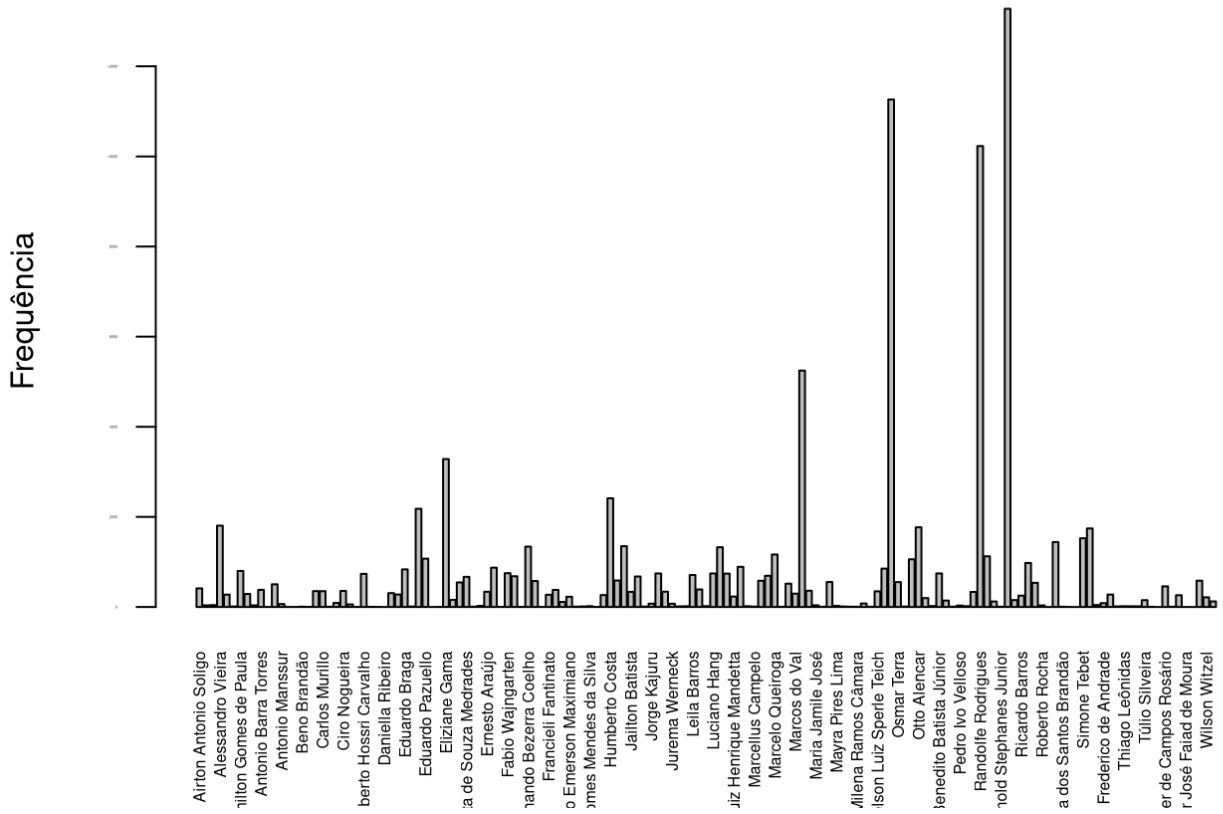
```
NotasTaq <- NotasTaq %>%
        mutate(nome = gsub("Marcelo Antônio Cartaxo Queiroga Lopes", "Marcelo Queir
oga", nome) %>%
                    gsub("Gen. Eduardo Pazuello", "Eduardo Pazuello", .) %>%
                    gsub("Francieli Fontana Sutile Tardetti Fantinato|Francieli Fon
tana Sutile Fantinato", "Francieli Fantinato", .) %>%
                    gsub("Luis Ricardo Fernandes Miranda", "Luis Miranda" , .) %>%
                    gsub("Marcelo Blanco da Costa", "Marcelo Blanco", .)
            )
```

Conferindo se deu certo. O número de participantes deve diminuir após as substituições:

```
NotasTaq$nome |> unique() |> length()
## [1] 149
```

Numa primeira observação, veremos a quantidade de intervenções de cada pessoa, isto é, quantas vezes que uma pessoa iniciou uma fala, independente da quantidade de palavras ditas por esta. Para tal, vamos fazer um gráfico de barras com o barplot do pacote base, que é nativo do R:

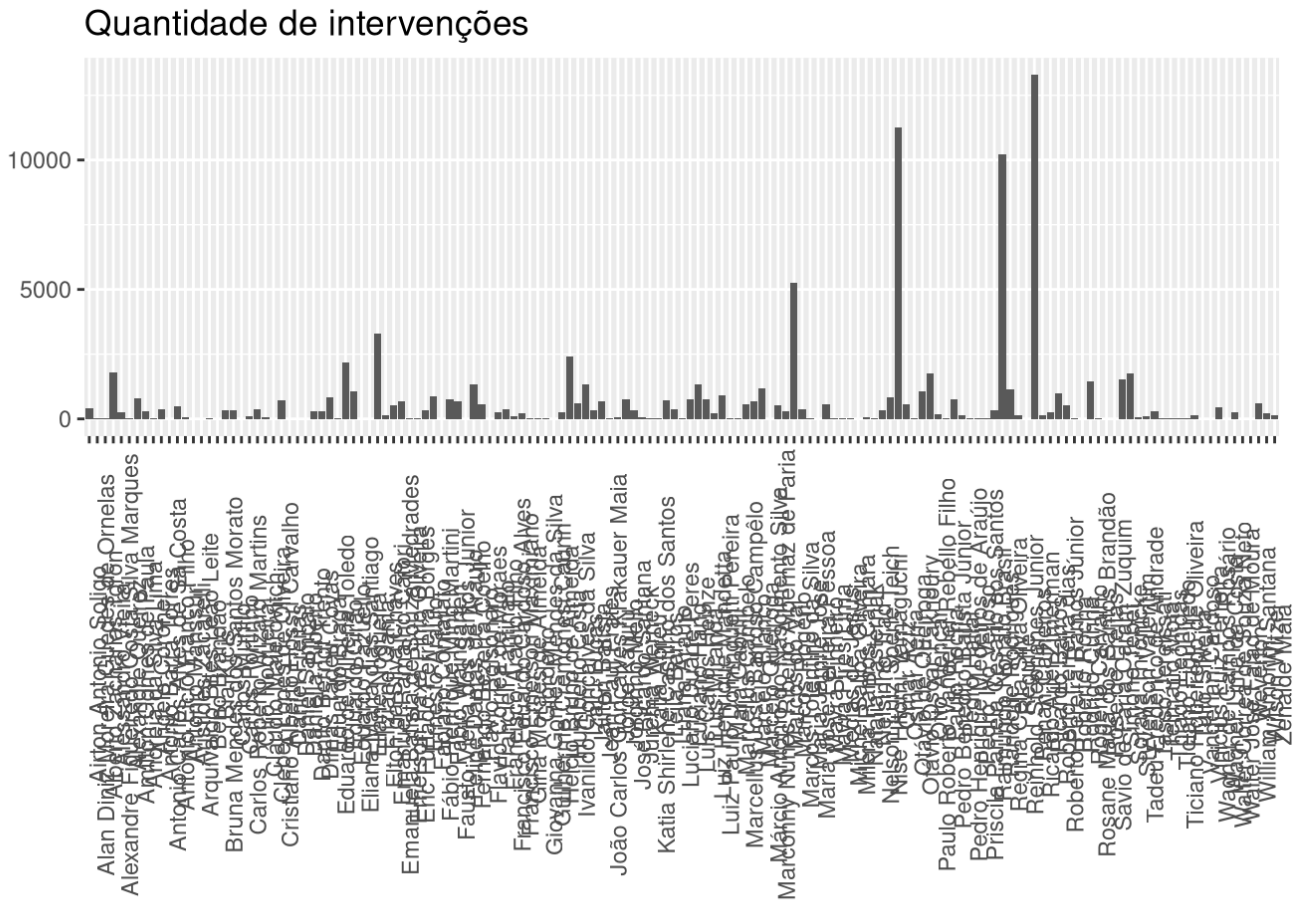
```
intervencoes <- NotasTaq$nome |> table()

intervencoes %>%
    barplot(.,
            main="Quantidade de Intervenções", # titulo
            ylab="Frequência", # eixo y
            cex.lab=1, # tamanho do label
            cex.axis=0.1, # tamanho do texto nos eixos
            cex.names=0.5, # tamanho dos nomes eixo x
            las=2 # rotacionando
    )
```



Ou usando ggplot2 para gerar o gráfico.

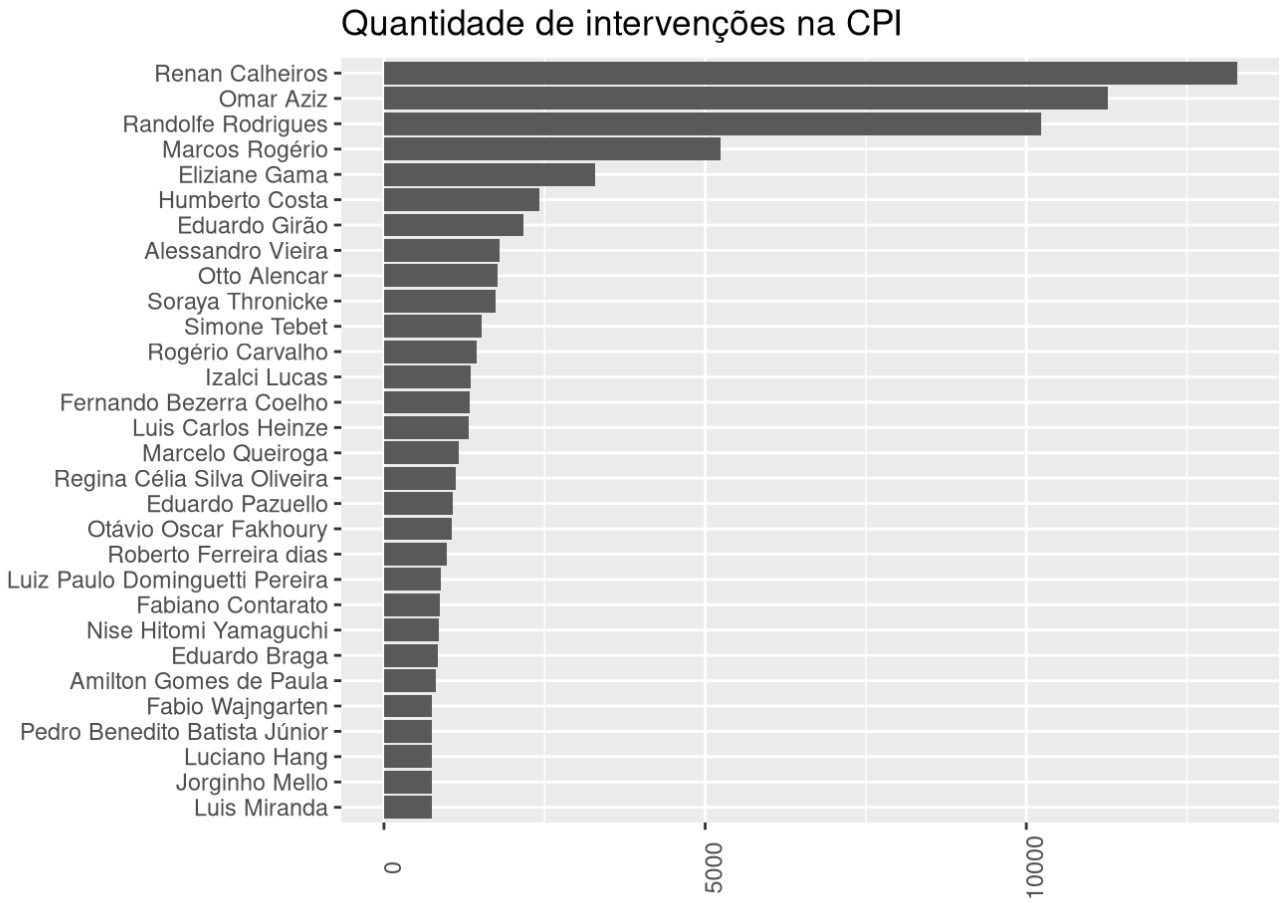
Se olharmos a frequência de mais de 150 pessoas, teremos um gráfico não muito compreensível:

```
ggplot(data = NotasTaq,
       aes(x = nome, y = )) +
        geom_bar() +
        labs(title = "Quantidade de intervenções",
             x = "",y = "") +
        theme(axis.text.x = element_text(angle = 90))
```



Melhoramos este gráfico se restringirmos apenas aos casos mais frequentes e rotacionando seu eixo.

```
# criando um novo dataframe com nomes e contagem
  intervencoes <- NotasTaq$nome %>% plyr::count()
# renomeando as colunas
  colnames(intervencoes) <- c("nome","freq")

  # Ordenando pela coluna "freq" de modo decrescente.
  dplyr::arrange(intervencoes, desc(freq)) %>%
    # restringindo aos 30 mais frequêntes
    head(30) %>%
    ggplot( aes(x = reorder(nome, freq), y = freq)) +
    geom_col() +
    labs(title = "Quantidade de intervenções na CPI", x = "",y = "") +
    theme(axis.text.x = element_text(angle = 90)) +
    # girando o gráfico
    coord_flip()
```



Alguns comentários sobre este gráfico acima:

- Nenhuma grande surpresa na distribuição de intervenções. Os nomes mais frequentes são o presidente e o relator.
- Para contar as intervenções pode-se usar o `table` - como usamos para gerar o barplot anterior - mas uma opção mais prática é usar o `plyr::count()`, que nos retorna um dataframe, o que torna mais fácil lidar com dados gerados.
- Apesar de termos ordenado nossos dados com base na frequência, o ggplot organiza os dados com base na ordem alfabética dos nomes. Mas se quisermos organizar na ordem das intervenções, usamos `aes(x = reorder(eixoX, eixoY), y = eixoY))` ou no caso `aes(x = reorder(nome, freq), y = freq))`.

Para examinarmos alguns termos e algumas palavras em seu contexto, isto é, com algumas palavras ao seu redor, podemos utilizar o seguinte código.

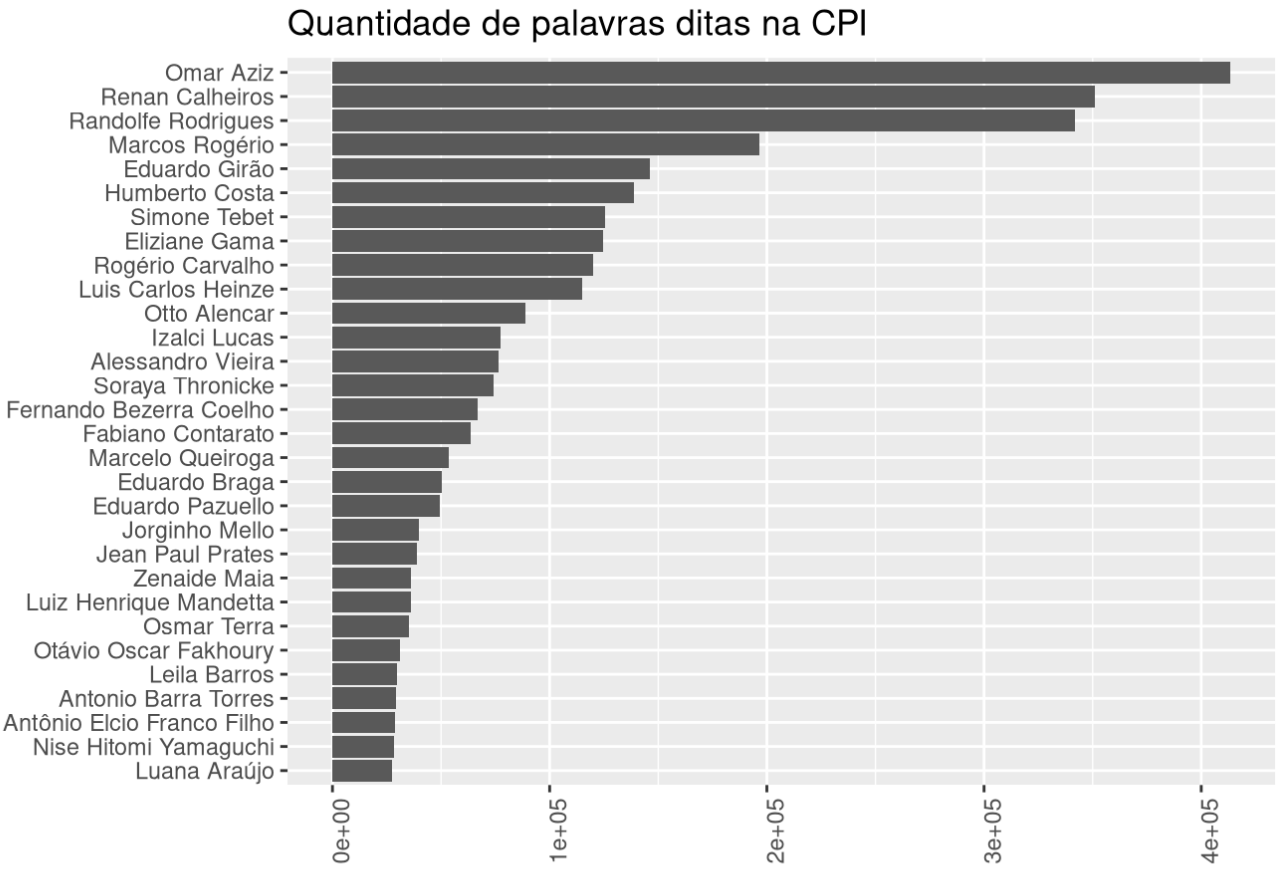
```
# Termo a ser buscado
termo="MP"
stringr::str_extract_all(NotasTaq$fala, paste("(\\w+){2}", termo, "(\\w+\\W{1,
2}){3}") ) |> unlist() |> plyr::count() |> arrange(-freq) |> as_tibble()
## # A tibble: 23 × 2
##    x                               freq
##    <chr>                          <int>
##  1 "do MP de São Paulo, "             2
##  2 "Essa MP deveria ser editada "     2
##  3 "assina MP 1.003, que "            1
##  4 "assina MP que libera mais "       1
##  5 "da MP 1.003, e "                  1
##  6 "da MP 1.003, um "                 1
##  7 "da MP 1.026 dizia "               1
##  8 "da MP 1.026; ou "                 1
##  9 "da MP da crise energética; "      1
## 10 "da MP e do fato "                 1
## # … with 13 more rows
```

Vamos substituir abreviaturas para evitar duplicações, como “Sr.” e “Senhor” não serem considerados termos distintos, mas sim um mesmo termo. É bem provável que outras abreviaturas importantes não estejam nesta lista.

```
# criando uma dataframe de abreviaturas
D.subs <- read.table(header=TRUE, sep=":", text=
'abr:subs
V.? Sa.:Vossa Senhoria
S. Paulo:São Paulo
S. Exa.:Sua Excelência
S. Exas.:Suas Excelências
V. Exa.:Vossa Excelência
V. Exas.:Vossas Excelências
art.:artigo
Cel.:coronel
Exma.|Ex.ma.:excelentíssima
Exmo.|Ex.mo.:excelentíssimo
STF|Supremo Tribunal Federal:Supremo_Tribunal_Federal
MPF:Ministério_Público_Federal
[a|A] MP:a Medida_Provisória
[o|O] MP:o Ministério_Público
PF:Polícia_federal
Gen.:general
Jr.:Júnior
Mr.:mister
Sgt.:sargento
Dra.:Doutora
Dr.:Doutor
Drs.:doutores
Sr.:Senhor
Srs.:Senhores
Sras.:Senhoras
Sra.|sr.a.:Senhora
Srta.|sr.a.:Senhorita
V.:Vossa
Exa.:Excelência')

# Preparando o df com substituições para que . seja entendido literalmente, não com
o regex
D.subs <- D.subs |> mutate(abr = stringr::str_replace_all(abr, c("\\." = "\\\\.")
))
# strings a serem substituídas..
subsVec <- tibble::deframe(D.subs)
# realizando as substituições
NotasTaq2  <- NotasTaq |> mutate(fala= stringr::str_replace_all(fala, subsVec))
```

Vamos ver agora o ranking por quantidade de palavras ditas, utilizando números mais gerais dos parlamentares, referentes à todas as sessões. Para tal, vamos agregar as falas de diferentes dias em uma mesma linha, por nome, através dos comandos (ou “verbos”) do dplyr group_by e summarize. Como o que queremos é que junte todas as falas em uma só célula, vamos usar de `paste()` com o parâmetro `collapse = " "`, que indica que entre uma fala e outra, que serão condensadas em uma só célula, entre cada elemento será inserido um espaço vazio, para evitar que uma palavra final de uma célula fique colada à palavra inicial da célula seguinte.

```
NotasTaq_falas.agrupadas <- NotasTaq2 %>%
      group_by(nome) %>%
      summarize(falas = paste(fala, collapse = " "))

# se quisermos observar a estrutura de nosso dataframe
    str(NotasTaq_falas.agrupadas)
## tibble [149 × 2] (S3: tbl_df/tbl/data.frame)
##  $ nome : chr [1:149] "Airton Antonio Soligo" "Alan Diniz Moreira Guedes de Orne
las" "Alberto Zacharias Toron" "Alessandro Vieira" ...
##  $ falas: chr [1:149] "Senhor Presidente, acredito que, em função da decisão, eu
não sou obrigado, mas estou aqui para dizer a verdade"| __truncated__ "Presidente,
só uma questão de ordem então?  Para deixar o registro, então, de que, com a limina
r do Supremo_Tri"| __truncated__ "Perfeitamente, Excelência, Senhor Presidente em e
xercício.  Senhor Presidente, o despacho, se Vossa Excelência "| __truncated__ "Pel
a ordem, Presidente Otto Alencar.  Senhor Presidente...  Obrigado, Senhor President
e. Apenas quero contradit"| __truncated__ ...

# contando as palavras
    NT_falasJuntasCount <-  NotasTaq_falas.agrupadas |>
                mutate(N_palavras = stringr::str_count(falas, "\\W"), .after = 1) |
>
                # reordenar pelo número de palavras (arrange) dos maiores valores a
os menores (desc)
                arrange(desc(N_palavras))

# plotando o gráfico
NT_falasJuntasCount %>%
      # restringindo aos primeiros resultados
      head(30) %>%
      # reorder para ordenar o gráfico, não pela ordem alfabética dos nomes
      ggplot( aes(x = reorder(nome, N_palavras), y = N_palavras)) +
      geom_col() +
      labs(title = "Quantidade de palavras ditas na CPI", x = "",y = "",
           caption = "Elaboração: Alisson Soares") +
      theme(axis.text.x = element_text(angle = 90)) +
      # girando o gráfico
      coord_flip()
```



Elaboração: Alisson Soares

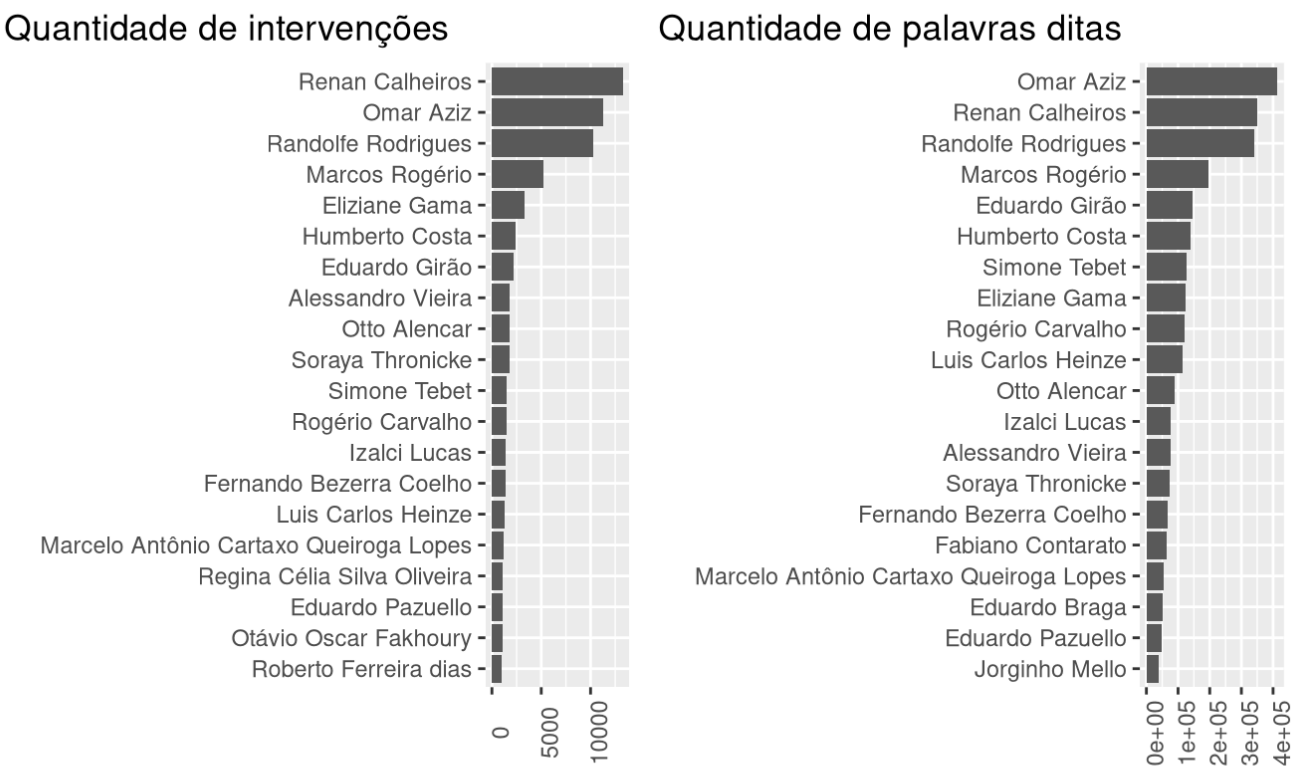
Vamos colocar os gráficos lado a lado para facilitar a comparação. Para tal, vamos utilizar o pacote patchwork (https://cran.r-project.org/web/packages/patchwork/index.html) que torna bem fácil colocar múltiplos gráficos juntos, nas mais diferentes configurações de layout. Para instalar, podemos usar o comando `install.packages('patchwork')`.

```
# carregando o pacote
library(patchwork)
```

Para usá-lo, vamos primeiro salvar os gráficos como objetos R e depois vamos organizá-los no patchwork. Os gráficos são os mesmos que usamos mais acima.

```
# Número de corte do máximo de itens a aparecer no gráfico
n_max <- 20
graf1 <- dplyr::arrange(intervencoes, desc(freq)) %>%
# restringindo aos 30 mais frequentes
        head(n_max) %>%
        ggplot( aes(x = reorder(nome, freq), y = freq)) +
        geom_col() +
        labs(title = "Quantidade de intervenções", x = "",y = "") +
        theme(axis.text.x = element_text(angle = 90),
              # Deslocando o título do gráfico para ficar mais visível
              plot.title.position = "plot") +
        # girando o gráfico
        coord_flip()

graf2 <- NT_falasJuntasCount %>%
# restringindo aos primeiros resultados
        top_n(20, N_palavras) %>%
        ggplot( aes(x = reorder(nome, N_palavras), y = N_palavras)) +
        geom_col() +
      labs(title = "Quantidade de palavras ditas", x = "",y = "") +
        #labs(title = "palavras ditas", x = "",y = "") +
        theme(axis.text.x = element_text(angle = 90), plot.title.position = "plot")
+
        # girando o gráfico
        coord_flip()

graf1 + graf2 +
# adicionando titulo ao gráfico
plot_annotation(title = 'CPI da Pandemia: falas quantificadas de todas as reuniões
',
                caption = '*Apenas os 20 mais frequentes aparecem nos gráficos\nEla
boração: Alisson Soares')
```

CPI da Pandemia: falas quantificadas de todas as reuniões



*Apenas os 20 mais frequentes aparecem nos gráficos
Elaboração: Alisson Soares

Vamos criar um dataframe separando apenas os senadores

Vamos restringir nossa base de dados para poder explorá-la melhor. Vamos restringir a somente as falas dos senadores.

3.1 Opção 1: pegando apenas os senadores

Como os senadores que participaram possuem campo com partido e bloco parlamentar preenchido, selecionaremos os senadores excluindo as linhas cuja célula da coluna “BlocoParl” esteja vazia. <<

- O campo “funcao_blocoPar” possui, além da função e bloco parlamentar, possui também questões de ordem
- O campo “part” melhora a situação, mas podendo haver parlamentar sem partido, não resolve nosso problema
- o campo estado pode resolver nosso problema.

```
siglas <- c("AC", "AL", "AP", "AM", "BA", "CE", "DF", "ES", "GO", "MA", "MT", "MS",
"MG", "PA", "PB", "PR", "PE", "PI", "RJ", "RN", "RS", "RO", "RR", "SC", "SP", "SE",
"TO")
DFsenadores <- NotasTaq2 %>%
      #filter(siglas %in% estado)
      filter(estado %in% siglas)
    DFsenadores
## # A tibble: 67,220 × 9
##    reuniao data       nome        funcao_blocoPar BlocoParl partido estado
##      <dbl> <date>     <chr>       <chr>           <chr>     <chr>   <chr>
##  1       1 2021-04-27 Otto Alenc… PRESIDENTE      (Otto Al… PSD     BA
##  2       1 2021-04-27 Ciro Nogue… (Bloco Parlame… Bloco Pa… PP      PI
##  3       1 2021-04-27 Otto Alenc… PRESIDENTE      (Otto Al… PSD     BA
##  4       1 2021-04-27 Ciro Nogue… (Bloco Parlame… Bloco Pa… PP      PI
##  5       1 2021-04-27 Otto Alenc… PRESIDENTE      (Otto Al… PSD     BA
##  6       1 2021-04-27 Ciro Nogue… (Bloco Parlame… Bloco Pa… PP      PI
##  7       1 2021-04-27 Otto Alenc… PRESIDENTE      (Otto Al… PSD     BA
##  8       1 2021-04-27 Ciro Nogue… (Bloco Parlame… Bloco Pa… PP      PI
##  9       1 2021-04-27 Otto Alenc… PRESIDENTE      (Otto Al… PSD     BA
## 10       1 2021-04-27 Ciro Nogue… (Bloco Parlame… Bloco Pa… PP      PI
## # … with 67,210 more rows, and 2 more variables: complemento <chr>,
## #   fala <chr>
    unique(DFsenadores$nome)
##  [1] "Otto Alencar"              "Ciro Nogueira"
##  [3] "Jorginho Mello"            "Izalci Lucas"
##  [5] "Alessandro Vieira"         "Eduardo Braga"
##  [7] "Eduardo Girão"             "Marcos Rogério"
##  [9] "Omar Aziz"                 "Humberto Costa"
## [11] "Rogério Carvalho"          "Weverton"
## [13] "Eliziane Gama"             "Randolfe Rodrigues"
## [15] "Paulo Rocha"               "Flávio Bolsonaro"
## [17] "Renan Calheiros"           "Fernando Bezerra Coelho"
## [19] "Luis Carlos Heinze"        "Angelo Coronel"
## [21] "Marcos do Val"             "Simone Tebet"
## [23] "Leila Barros"              "Tasso Jereissati"
## [25] "Zenaide Maia"              "Fabiano Contarato"
## [27] "Vanderlan Cardoso"         "Telmário Mota"
## [29] "Soraya Thronicke"          "Jean Paul Prates"
## [31] "Mara Gabrilli"             "Reguffe"
## [33] "Mecias de Jesus"           "Roberto Rocha"
## [35] "Kátia Abreu"               "Daniella Ribeiro"
## [37] "Jorge Kajuru"              "Carlos Portinho"
## [39] "Styvenson Valentim"        "Nelsinho Trad"
## [41] "Giordano"                  "Osmar Terra"
## [43] "Luis Miranda"              "Rodrigo Cunha"
## [45] "Eliane Nogueira"           "Reinhold Stephanes Junior"
## [47] "Ricardo Barros"            "Rose de Freitas"
## [49] "Daniel Freitas"            "Bia Kicis"
```

No entanto, esta opção captou também os deputados que lá falaram, como Luis Miranda, Ricardo Barros e Bia Kicis. Teremos de utilizar uma outra abordagem.

3.2 Opção 2: Somente senadores, a partir do site

do Senado

Vamos pegar os nomes dos senadores no site do senado fazendo raspagem dos dados no site do senado. A depender de quando você rodar o código abaixo, os nomes terão mudado. Por isso colo os nomes mais à frente.

```
library(rvest)

# Pegando os senadores em exercício
pagina_senadores <- rvest::read_html(url("https://www25.senado.leg.br/web/senadores/em-exercicio/"))
```

Vamos pegar apenas os nomes dos senadores

```
senadores <- pagina_senadores %>%
        rvest::html_elements("#senadoresemexercicio-tabela-senadores a") %>%
        rvest::html_text()

senadores[1:15]
##  [1] ""                "Mailza Gomes"    "Marcio Bittar"
##  [4] "Sérgio Petecão"  ""                "Fernando Collor"
##  [7] "Renan Calheiros" "Rodrigo Cunha"   ""
## [10] "Eduardo Braga"   "Omar Aziz"       "Plínio Valério"
## [13] ""                "Davi Alcolumbre" "Lucas Barreto"
```

Vimos que há muitos elementos vazios no vetor. Vamos retirar os elementos vazios com os parâmetros `!=` (diferente de) e `""` indicando os elementos vazios:

```
senadores <- senadores[senadores!= ""]
senadores
##  [1] "Mailza Gomes"            "Marcio Bittar"
##  [3] "Sérgio Petecão"          "Fernando Collor"
##  [5] "Renan Calheiros"         "Rodrigo Cunha"
##  [7] "Eduardo Braga"           "Omar Aziz"
##  [9] "Plínio Valério"          "Davi Alcolumbre"
## [11] "Lucas Barreto"           "Randolfe Rodrigues"
## [13] "Angelo Coronel"          "Jaques Wagner"
## [15] "Otto Alencar"            "Chiquinho Feitosa"
## [17] "Cid Gomes"               "Eduardo Girão"
## [19] "Izalci Lucas"            "Leila Barros"
## [21] "Reguffe"                 "Fabiano Contarato"
## [23] "Marcos do Val"           "Rose de Freitas"
## [25] "Jorge Kajuru"            "Luiz do Carmo"
## [27] "Vanderlan Cardoso"       "Eliziane Gama"
## [29] "Roberto Rocha"           "Weverton"
## [31] "Alexandre Silveira"      "Carlos Viana"
## [33] "Rodrigo Pacheco"         "Nelsinho Trad"
## [35] "Simone Tebet"            "Soraya Thronicke"
## [37] "Carlos Fávaro"           "Jayme Campos"
## [39] "Wellington Fagundes"     "Jader Barbalho"
## [41] "Paulo Rocha"             "Zequinha Marinho"
## [43] "Daniella Ribeiro"        "Nilda Gondim"
## [45] "Veneziano Vital do Rêgo" "Fernando Bezerra Coelho"
## [47] "Humberto Costa"          "Jarbas Vasconcelos"
## [49] "Eliane Nogueira"         "Elmano Férrer"
## [51] "Marcelo Castro"          "Alvaro Dias"
## [53] "Flávio Arns"             "Oriovisto Guimarães"
## [55] "Carlos Portinho"         "Flávio Bolsonaro"
## [57] "Romário"                 "Jean Paul Prates"
## [59] "Styvenson Valentim"      "Zenaide Maia"
## [61] "Acir Gurgacz"            "Confúcio Moura"
## [63] "Marcos Rogério"          "Chico Rodrigues"
## [65] "Mecias de Jesus"         "Telmário Mota"
## [67] "Lasier Martins"          "Luis Carlos Heinze"
## [69] "Paulo Paim"              "Dário Berger"
## [71] "Esperidião Amin"         "Jorginho Mello"
## [73] "Alessandro Vieira"       "Maria do Carmo Alves"
## [75] "Rogério Carvalho"        "Giordano"
## [77] "José Serra"              "Mara Gabrilli"
## [79] "Eduardo Gomes"           "Irajá"
## [81] "Kátia Abreu"
```

De posse dos nomes dos senadores, vamos ver quais nomes no nosso dataframe da CPI tem intersecção com a lista de senadores. Lembrando, esta lista foi gerada pouco tempo depois da CPI, assim, dependendo de quanto tempo você for tentar reproduzir o exemplo, o link pode ter expirado ou a lista de senadores pode já ter mudado.

Já havíamos gerado um vetor com os nomes de todos que participaram da CPI, o “participantes”. Vamos cruzá-lo com a listagem de nome de senadores que geramos.

```
str(participantes)
##  chr [1:154] "Otto Alencar" "Ciro Nogueira" "Jorginho Mello" ...

senadoresNaCPI <- participantes[participantes %in% senadores]
senadoresNaCPI
##  [1] "Otto Alencar"            "Jorginho Mello"
##  [3] "Izalci Lucas"            "Alessandro Vieira"
##  [5] "Eduardo Braga"           "Eduardo Girão"
##  [7] "Marcos Rogério"          "Omar Aziz"
##  [9] "Humberto Costa"          "Rogério Carvalho"
## [11] "Weverton"                "Eliziane Gama"
## [13] "Randolfe Rodrigues"      "Paulo Rocha"
## [15] "Flávio Bolsonaro"        "Renan Calheiros"
## [17] "Fernando Bezerra Coelho" "Luis Carlos Heinze"
## [19] "Angelo Coronel"          "Marcos do Val"
## [21] "Simone Tebet"            "Leila Barros"
## [23] "Zenaide Maia"            "Fabiano Contarato"
## [25] "Vanderlan Cardoso"       "Telmário Mota"
## [27] "Soraya Thronicke"        "Jean Paul Prates"
## [29] "Mara Gabrilli"           "Reguffe"
## [31] "Mecias de Jesus"         "Roberto Rocha"
## [33] "Kátia Abreu"             "Daniella Ribeiro"
## [35] "Jorge Kajuru"            "Carlos Portinho"
## [37] "Styvenson Valentim"      "Nelsinho Trad"
## [39] "Giordano"                "Rodrigo Cunha"
## [41] "Eliane Nogueira"         "Rose de Freitas"
```

Tendo agora a lista dos senadores que participaram da CPI, vamos filtrar as falas somente destes.

```
senadoresdf <- NotasTaq2 %>%
        filter(nome %in% senadoresNaCPI )
str(senadoresdf)
## tibble [65,457 × 9] (S3: tbl_df/tbl/data.frame)
##  $ reuniao        : num [1:65457] 1 1 1 1 1 1 1 1 1 1 ...
##  $ data           : Date[1:65457], format: "2021-04-27" "2021-04-27" ...
##  $ nome           : chr [1:65457] "Otto Alencar" "Otto Alencar" "Otto Alencar" "
Otto Alencar" ...
##  $ funcao_blocoPar: chr [1:65457] "PRESIDENTE" "PRESIDENTE" "PRESIDENTE" "PRESID
ENTE" ...
##  $ BlocoParl      : chr [1:65457] "(Otto Alencar. PSD - BA. Fala da Presidênci
a.)" "(Otto Alencar. PSD - BA)" "(Otto Alencar. PSD - BA)" "(Otto Alencar. PSD - B
A)" ...
##  $ partido        : Named chr [1:65457] "PSD" "PSD" "PSD" "PSD" ...
##   ..- attr(*, "names")= chr [1:65457] "PSD" "PSD" "PSD" "PSD" ...
##  $ estado         : chr [1:65457] "BA" "BA" "BA" "BA" ...
##  $ complemento    : chr [1:65457] "Fala da Presidência" "" "" "" ...
##  $ fala           : chr [1:65457] "Invocando a proteção de Deus, declaro aberta
a sessão para eleição, já que temos quórum suficiente para a abert"| __truncated__
"Senador Ciro Nogueira, esta é uma Comissão Parlamentar de Inquérito, Vossa Excelên
cia sabe que não é temática. "| __truncated__ "Eu indeferi. Sou Presidente e posso
indeferir. " "Por que Vossa Excelência não questionou à época essa questão de orde
m? " ...
```

Criando um DF das falas de cada senador todas reunidas

```
senadores_falasJuntas <- NT_falasJuntasCount %>%
        filter(nome %in% senadoresNaCPI )
str(senadores_falasJuntas)
## tibble [42 × 3] (S3: tbl_df/tbl/data.frame)
##  $ nome      : chr [1:42] "Omar Aziz" "Renan Calheiros" "Randolfe Rodrigues" "Ma
rcos Rogério" ...
##  $ N_palavras: int [1:42] 413543 351349 341887 196716 146154 138842 125590 12462
6 120231 114925 ...
##  $ falas     : chr [1:42] "Como é que é? Peço só um minutinho, só um minutinho!
Senhor Presidente...  Eu acho que Vossa Excelência.. Nós "| __truncated__ "Quer diz
er que há outros impedimentos a serem...  Acredito não ser o caso de VossaExcelênci
a, mas o Estado de A"| __truncated__ "Presidente...  Presidente, qual a ordem?  Pre
sidente, só para declinar a ordem, quem são?  Agora é a Eliziane? "| __truncated__
"Senhor Presidente, Senhoras e Senhores Senadores, faço a presente questão de orde
m, Senhor Presidente, desde lo"| __truncated__ ...
```

E outro tibble apenas com quem não for senador

```
nao_senadores <- NotasTaq2 %>%
        filter(BlocoParl == "")
nao_senadores
## # A tibble: 24,296 × 9
##    reuniao data       nome        funcao_blocoPar BlocoParl partido estado
##      <dbl> <date>     <chr>       <chr>           <chr>     <chr>   <chr>
##  1      10 2021-05-20 Eduardo Pa… ""              ""        ""      ""
##  2      10 2021-05-20 Eduardo Pa… ""              ""        ""      ""
##  3      10 2021-05-20 Eduardo Pa… ""              ""        ""      ""
##  4      10 2021-05-20 Eduardo Pa… ""              ""        ""      ""
##  5      10 2021-05-20 Eduardo Pa… ""              ""        ""      ""
##  6      10 2021-05-20 Eduardo Pa… ""              ""        ""      ""
##  7      10 2021-05-20 Eduardo Pa… ""              ""        ""      ""
##  8      10 2021-05-20 Eduardo Pa… ""              ""        ""      ""
##  9      10 2021-05-20 Eduardo Pa… ""              ""        ""      ""
## 10      10 2021-05-20 Eduardo Pa… ""              ""        ""      ""
## # … with 24,286 more rows, and 2 more variables: complemento <chr>,
## #   fala <chr>
unique(nao_senadores$nome)
##  [1] "Eduardo Pazuello"
##  [2] "Mayra Pinheiro"
##  [3] "Dimas Tadeu Covas"
##  [4] "Nise Hitomi Yamaguchi"
##  [5] "Luana Araújo"
##  [6] "Marcelo Queiroga"
##  [7] "Antônio Elcio Franco Filho"
##  [8] "Natalia Pasternak"
##  [9] "Cláudio Maierovitch"
## [10] "Marcellus José Barroso Campêlo"
## [11] "Marcellus Campelo"
## [12] "Wilson Witzel"
## [13] "Francisco Eduardo Cardoso Alves"
## [14] "Ricardo Ariel Zimerman"
## [15] "Osmar Terra"
## [16] "Jurema Werneck"
## [17] "Pedro Hallal"
## [18] "Luis Miranda"
## [19] "Fausto Vieira dos Santos Junior"
## [20] "Wagner Lima da Costa"
## [21] "Gina Moraes de Almeida"
## [22] "Carlos Roberto Wizard Martins"
## [23] "Alberto Zacharias Toron"
## [24] "Guilherme Cremonesi Caurin"
## [25] "Luiz Henrique Mandetta"
## [26] "Luiz Paulo Dominguetti Pereira"
## [27] "Flavio Correa de Moraes"
## [28] "Regina Célia Silva Oliveira"
## [29] "Pedro Henrique Medeiros de Araújo"
## [30] "Roberto Ferreira dias"
## [31] "Maria Jamile José"
## [32] "Francieli Fantinato"
## [33] "Thiago Leônidas"
## [34] "William Amorim Santana"
## [35] "Emanuela Batista de Souza Medrades"
```

```
## [36] "Ticiano Figueiredo de Oliveira"
## [37] "Pedro Ivo Velloso"
## [38] "Cristiano Alberto Hossri Carvalho"
## [39] "Fábio Henrique Ming Martini"
## [40] "Amilton Gomes de Paula"
## [41] "Otávio de Queiroga"
## [42] "Marcelo Blanco"
## [43] "Eric Furtado Ferreira Borges"
## [44] "Nelson Luiz Sperle Teich"
## [45] "Airton Antonio Soligo"
## [46] "Emerson Paxá Pinto Oliveira"
## [47] "Helcio Bruno de Almeida"
## [48] "João Carlos Gonçalves Krakauer Maia"
## [49] "Jailton Batista"
## [50] "Ricardo Barros"
## [51] "Alexandre Figueiredo Costa Silva Marques"
## [52] "Eduardo de Vilhena Toledo"
## [53] "Túlio Silveira"
## [54] "Francisco Emerson Maximiano"
## [55] "Ticiano Figueiredo"
## [56] "Emanuel Ramalho Catori"
## [57] "Michel Saliba Oliveira"
## [58] "Roberto Pereira Ramos Júnior"
## [59] "Alexandre Queiroz"
## [60] "José Ricardo Santana"
## [61] "Alan Diniz Moreira Guedes de Ornelas"
## [62] "Ivanildo Gonçalves da Silva"
## [63] "Francisco Araújo Filho"
## [64] "Cleber Lopes de Oliveira"
## [65] "Marcos Tolentino da Silva"
## [66] "Luciano Duarte Peres"
## [67] "Marconny Nunes Ribeiro Albernaz de Faria"
## [68] "Wagner de Campos Rosário"
## [69] "Pedro Benedito Batista Júnior"
## [70] "Aristides Zacarelli"
## [71] "Maria José Ferreira Pessoa"
## [72] "Vinicius Luiz Ferreira"
## [73] "Danilo Berndt Trento"
## [74] "Bruna Mendes dos Santos Morato"
## [75] "Antonio Barra Torres"
## [76] "Luciano Hang"
## [77] "Beno Brandão"
## [78] "Otávio Oscar Fakhoury"
## [79] "Antonio Manssur"
## [80] "Milena Ramos Câmara"
## [81] "Raimundo Nonato Brasil"
## [82] "Andreia da Silva Lima"
## [83] "Walter José Faiad de Moura"
## [84] "Paulo Roberto Vanderlei Rebello Filho"
## [85] "Walter Correa de Souza Neto"
## [86] "Tadeu Frederico de Andrade"
## [87] "Priscila Pamela Cesario dos Santos"
## [88] "Rosane Maria dos Santos Brandão"
## [89] "Mayra Pires Lima"
## [90] "Antonio Carlos Alves de Sá Costa"
```

```
## [91] "Giovanna Gomes Mendes da Silva"
## [92] "Katia Shirlene Castilho dos Santos"
## [93] "Márcio Antonio do Nascimento Silva"
## [94] "Elton da Silva Chaves"
## [95] "Fabio Wajngarten"
## [96] "Carlos Murillo"
## [97] "Ernesto Araújo"
```

Criando um grande objeto com todas as palavras, um *bag-of-words*: Mas **cuidado**, não rode o objeto "tudo" diretamete. Pelo seu grande tamanho, pode travar o R. Ao invés disso, apenas confira sua estrutura para conferir se está ok.

```
tudo <- paste(NotasTaq_falas.agrupadas$falas, collapse = " ")
str(tudo)
##  chr "Senhor Presidente, acredito que, em função da decisão, eu não sou obrigad
o, mas estou aqui para dizer a verdade"| __truncated__
class(tudo)
## [1] "character"
typeof(tudo)
## [1] "character"
```

Vamos contar quantas palavras foram ditas no total na CPI

```
# Tokenizando a cada espaço em branco
totalpalavras <- sapply(strsplit(tudo, " "), length)
# convertendo para um formato mais facilmente legível por humanos
totalpalavras |> format(big.mark = ".")
## Warning in prettyNum(.Internal(format(x, trim, digits, nsmall, width,
## 3L, : 'big.mark' and 'decimal.mark' are both '.', which could be confusing
## [1] "3.250.073"
```

Convertendo para tibble e tokenizando em palavras tudo que foi dito na CPI.

```
tudo.df <- as_tibble(tudo, falas = tudo)
tudo.tokens <- tudo.df %>%
        tidytext::unnest_tokens(word, value)
```

4 Análise textual

Vamos observar os assuntos mais frequentes ali através da frequência de palavras. Vamos ver, de forma geral, as palavras mais frequentes.

4.1 Frequência e *wordcloud* (Nuvem de palavras)

A função wordcloud do pacote wordcloud permite enviar o texto diretamente para processamento, mas esta opção não é indicada, por não permitir pré-processamento e por ser muito, bastante, extremamente lenta com a quantidade de texto que temos aqui. (Eu descobri isto testando antes). Vamos contar os valores únicos no nosso vetor "tudo.tokens"

```
# contando termos repetidos na coluna "word"
# Isto é, gerar a tabela de frequência de termos.
tudo.freq <- tudo.tokens %>%  dplyr::count(word, sort = TRUE)
# observando um pedaço
head(tudo.freq, 25)
## # A tibble: 25 × 2
##    word       n
##    <chr>  <int>
##  1 que   114479
##  2 o     104603
##  3 de    103579
##  4 a     102691
##  5 não    69265
##  6 e      64094
##  7 é      51666
##  8 eu     50458
##  9 da     43084
## 10 do     42760
## # … with 15 more rows
```

O que nos dá uma lista das palavras mais frequentes, porém bem pouco informativas do assunto. Vamos retirar as palavras vazias ou stopwords. Eu testei previamente e acrescentei à lista algumas palavras frequentes, porém pouco informativas. Vamos criar dois objetos com a mesma lista de stopwords, um como vetor e outro como dataframe, pois vamos precisar de ambos formatos.

```
# montando nossa nova listagem de stopwords
# juntando a lista de `stopwords::stopwords('pt')` com a nossa
SW <- c(stopwords::stopwords('pt'), 'é', 'aqui', 'então', 'porque', 'pra')
SW.df <- tibble(words = SW)

# retirando as stopwords
quase.tudo <- tudo.tokens$word[!(tudo.tokens$word) %in% SW]

# observando as palavras mais frequentes atuais, sem algumas stopwords
head(quase.tudo, 40)
##  [1] "senhor"         "presidente"     "acredito"       "função"
##  [5] "decisão"        "obrigado"       "dizer"          "verdade"
##  [9] "gostaria"       "usá"            "los"            "senhor"
## [13] "presidente"     "primeiramente"  "bom"            "dia"
## [17] "todos"          "gostaria"       "iniciar"        "fala"
## [21] "cumprimentando" "todos"          "senadores"      "senhoras"
## [25] "senadoras"      "desta"          "comissão"       "pessoa"
## [29] "senhor"         "presidente"     "omar"           "aziz"
## [33] "eminente"       "senador"        "relator"        "renan"
## [37] "calheiros"      "senhoras"       "senadoras"      "nome"

# testando a retirada de stopwords: "tudo.tokens" é maior que "quase.tudo"?
length(tudo.tokens$word) > length(quase.tudo)
## [1] TRUE
length(tudo.tokens$word)
## [1] 3102313
length(quase.tudo)
## [1] 1620348

# Contando a frequência
wordCount_semSW <- quase.tudo %>%
        plyr::count() |>
        arrange(-freq) |>
        as_tibble()
wordCount_semSW
## # A tibble: 42,201 × 2
##    x           freq
##    <chr>      <int>
##  1 senhor     31611
##  2 senador    18665
##  3 presidente 18033
##  4 vossa      12073
##  5 saúde       9071
##  6 gente       8740
##  7 excelência  8731
##  8 ministério  7391
##  9 fazer       6420
## 10 cpi         6239
## # … with 42,191 more rows
```

E agora gerando nossa nuvem de palavras

```
# pegando apenas as palavras mais frequentes
pre.wc <- wordCount_semSW[1:150,]

wordcloud::wordcloud(pre.wc$x,
                     # se o input  para esta função contém as frequências de palavra
s, o item abaixo deve ser descomentado
                     pre.wc$freq,
                     # vetor com dois termos indicado o espectro de tamanho das palav
ras
                     scale=c(3,.6),
                     # cores, do menos frequente ao mais frequente
                     colors = c("royalblue","blue", "darkblue", "black"))
```

Vamos observar os ngramas - no caso bigramas e trigramas - que nos dão uma ideia melhor do sentido das discussões do que unigramas utilizados anteriormente.

```
multi.palavras <- tokenizers::tokenize_ngrams(tudo,
                                              # valores máximo e mínimo dos ngrams
                                              n=3, n_min = 2,
                                              stopwords = SW
                                              ) |>
                                  unlist()

tudo.freq <- multi.palavras  |>  plyr::count() |> arrange(-freq)
novas_sw <- c("senador", "vossa", "presidente")
# arrumando nossas stopwords para serem usadas com filter e grepl
# Ela será um único elemento com vários operadores "ous"
novas_sw <- paste(novas_sw, collapse = "|" )

# retirando as novas stop words
# opção 1
tudo.freq2 <- tudo.freq %>% filter(!grepl(novas_sw, .$x))

# opção 2
tudo.freq2 <- tudo.freq[!grepl(novas_sw, tudo.freq$x),]

# Observando nosso dataframe
tudo.freq2[1:35,]
##                                 x freq
## 3                ministério saúde 4703
## 6             comissão parlamentar 2128
## 7  comissão parlamentar inquérito 2103
## 8           parlamentar inquérito 2103
## 10               governo federal 1476
## 13                senhor relator 1210
## 14                     covid 19 1176
## 16          intervenção microfone 1162
## 17                marcos rogério 1104
## 18                 milhões doses  991
## 21            tratamento precoce  889
## 23                prevent senior  791
## 26                desta comissão  780
## 27                 questão ordem  744
## 30                      pode ser  724
## 31               renan calheiros  722
## 33            estados municípios  688
## 35               polícia federal  657
## 36                 todo respeito  637
## 37                estados unidos  628
## 38                     desta cpi  620
## 39                      fib bank  611
## 41                    todo mundo  604
## 42               naquele momento  602
## 43                 senhor senhor  600
## 44                neste momento  585
## 45               obrigado senhor  583
## 46                    quer dizer  579
## 47                  roberto dias  578
## 48                ricardo barros  571
## 51                   senhor pode  555
```

```
## 52                   alguma coisa  531
## 53                     sim senhor  523
## 54                 ministro saúde  522
## 55             ministério público  512
```

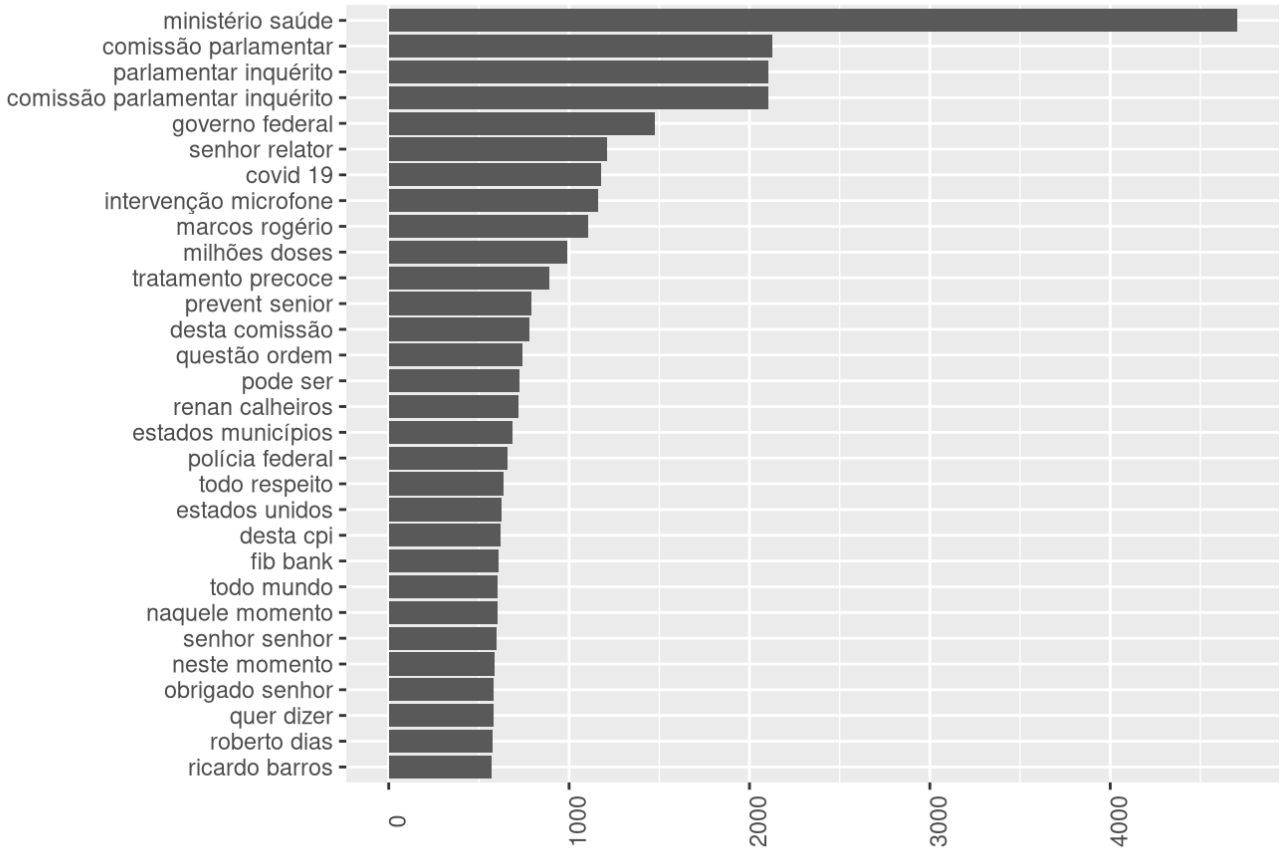
Gerando uma nuvem de palavras

```
# pegando apenas as palavras mais frequentes
pre.wc <- tudo.freq2[1:80,]

wordcloud::wordcloud(pre.wc$x,
                     # se o input  para esta função contém as frequências de palavra
s, o item abaixo deve ser descomentado
                     pre.wc$freq,
                     # vetor com dois termos indicado o espectro de tamanho das palav
ras
                     scale=c(3,.6),
                     # cores, do menos frequente ao mais frequente
                     colors = c("royalblue","blue", "darkblue", "black"))
```

Criando um ggplot com os ngrams mais frequentes

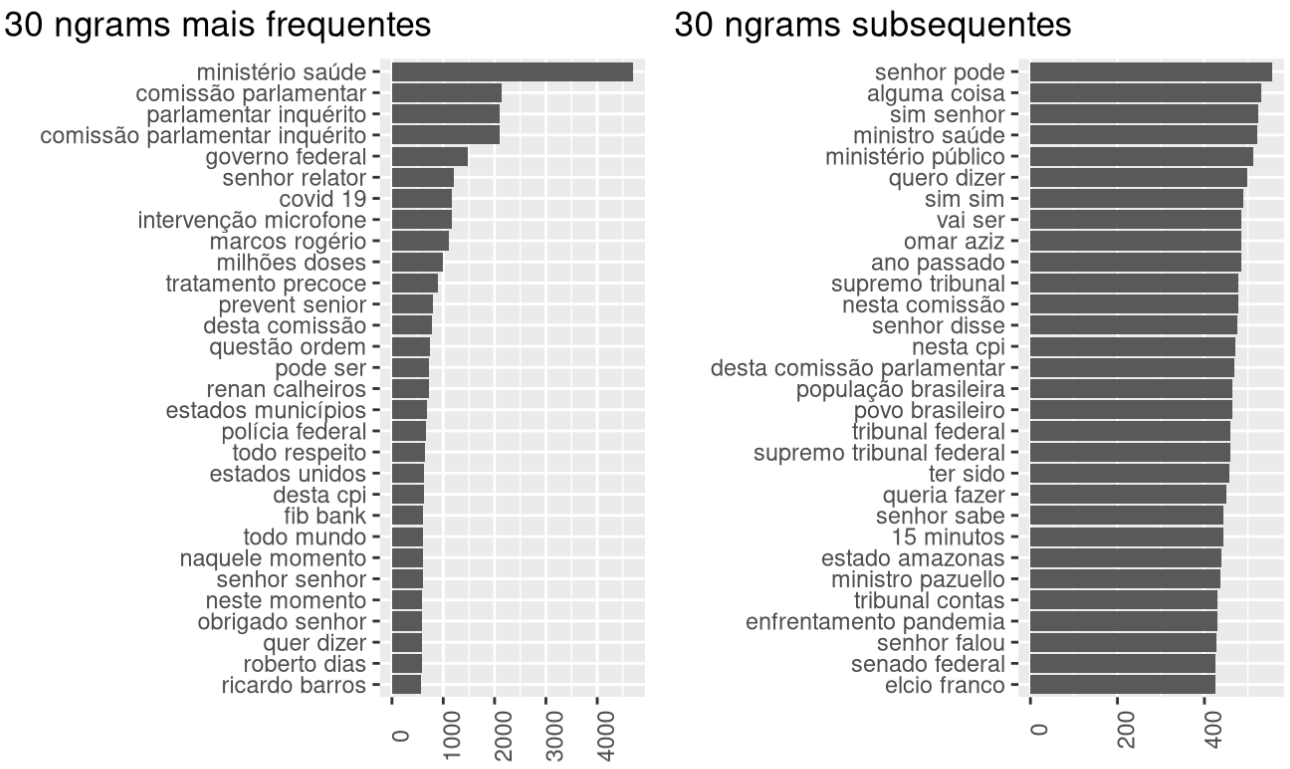
```
g.ngram.1 <- tudo.freq2[1:30,] %>%
        ggplot( aes(x = reorder(x, freq), y = freq)) +
        geom_col() +
        labs(title = "30 ngrams mais frequentes", x = "",y = "") +
        theme(axis.text.x = element_text(angle = 90),
              # Deslocando o título do gráfico para ficar mais visível
              plot.title.position = "plot") +
        # girando o gráfico
        coord_flip()
g.ngram.1
```

30 ngrams mais frequentes




```
g.ngram.2 <- tudo.freq2[31:60,] %>%
        ggplot( aes(x = reorder(x, freq), y = freq)) +
        geom_col() +
        labs(title = "30 ngrams subsequentes", x = "",y = "") +
        theme(axis.text.x = element_text(angle = 90),
              # Deslocando o título do gráfico para ficar mais visível
              plot.title.position = "plot") +
        # girando o gráfico
        coord_flip()

g.ngram.1 + g.ngram.2 +
        plot_annotation(title = '60 Bigramas e trigramas mais frequentes na CPI da
Pandemia',
                caption = '*Os gráficos estão em escalas diferentes\nElaboração: Al
isson Soares')
```

60 Bigramas e trigramas mais frequentes na CPI da Pandemia



*Os gráficos estão em escalas diferentes
Elaboração: Alisson Soares

Podemos ver que alguns termos como os trigramas “comissão parlamentar (de) inquérito” e “supremo tribunal federal” estão repetidos em bigramas. Mas bigramas e trigramas podem ser melhoradas, afim de manter termos mais significativos.

4.2 Extração de palavras chave Keywords - colocação (*collocation*)

O termo “colocação” refere-se a uma sequência de palavras que ocorrem juntas mais frequentemente que separadas ou pelo acaso. Usamos esta análise para encontrar termos

A função `keywords_collocation()` do pacote Udpipe faz este cálculo, aceitando como argumentos:

- input data frames, onde cada linha é um termo e estes estão na ordem que aparecem no texto
- `term` : indicação da coluna com termos
- `group` : indicação de id do documento

Além destes, há os argumentos opcionais:

- `ngram_max` : integral, indicando o tamanho máximo das colocações. O padrão é 2.
- `n_min` : integral indicando o número mínimo das colocações. O padrão 2.
- `sep` : separador das colocações. O padrão é ” “, isto é, espaço. Mas pode ser útil usar o símbolo”_” para tornar estes termos um único termo.

Esta função nos retorna um data frame com várias colunas, como veremos mais abaixo. Ela calcula:

• PMI (pointwise mutual information): log2(P(w1w2) / P(w1) P(w2)) • MD (mutual dependency): log2(P(w1w2)^2 / P(w1) P(w2)) • LFMD (log-frequency biased mutual dependency): MD + log2(P(w1w2))

```
PMI <- NotasTaq2 |> tidytext::unnest_tokens(word, fala,
                                            to_lower = FALSE) |>
           keywords_collocation(term = "word", group = "nome",
                                ngram_max = 4, sep = "_")
# vendo o df gerado
head(PMI, 20)
##               keyword ngram      left       right freq freq_left freq_right
pmi md      lfmd
## 1         Los_Angeles     2       Los     Angeles    3         3          3 19.9
7952  0 -19.97952
## 2          BCI_Balpex     2       BCI      Balpex    3         3          3 19.9
7952  0 -19.97952
## 3    Evelyn_Beatrice     2    Evelyn    Beatrice    3         3          3 19.9
7952  0 -19.97952
## 4         Wesley_Cota     2    Wesley        Cota    3         3          3 19.9
7952  0 -19.97952
## 5   Christian_Drosten     2 Christian     Drosten    3         3          3 19.9
7952  0 -19.97952
## 6     Jandira_Feghali     2   Jandira     Feghali    3         3          3 19.9
7952  0 -19.97952
## 7       Thermo_Fisher     2    Thermo      Fisher    3         3          3 19.9
7952  0 -19.97952
## 8             Fla_Flu     2       Fla         Flu    3         3          3 19.9
7952  0 -19.97952
## 9        Von_Holleben     2       Von    Holleben    3         3          3 19.9
7952  0 -19.97952
## 10       Apoorv_Kumar     2    Apoorv       Kumar    3         3          3 19.9
7952  0 -19.97952
## 11        Raman_Neves     2     Raman       Neves    3         3          3 19.9
7952  0 -19.97952
## 12      Goldman_Sachs     2   Goldman       Sachs    3         3          3 19.9
7952  0 -19.97952
## 13       Vick_VapoRub     2      Vick     VapoRub    3         3          3 19.9
7952  0 -19.97952
## 14       Yang_Wanming     2      Yang     Wanming    3         3          3 19.9
7952  0 -19.97952
## 15 bebidas_alcoólicas     2   bebidas  alcoólicas    3         3          3 19.9
7952  0 -19.97952
## 16          these_are     2     these         are    3         3          3 19.9
7952  0 -19.97952
## 17      shelf_company     2     shelf     company    3         3          3 19.9
7952  0 -19.97952
## 18           I'm_here     2       I'm        here    3         3          3 19.9
7952  0 -19.97952
## 19     pó_liofilizado     2        pó liofilizado    3         3          3 19.9
7952  0 -19.97952
## 20         en_passant     2        en     passant    3         3          3 19.9
7952  0 -19.97952
```

Após alguns testes, de ordenar pela frequência, pelo pmi ou lfmd, cheguei a esta configuração:

```
# o 4 grams
PMI |> filter(ngram == 4)|>
        filter(freq > 200) |>
        arrange(-pmi) |>
        select(keyword, freq, pmi, md, lfmd ) |>
        filter(stringr::str_detect(keyword, "[:upper:][:lower:]+.*_[:upper:]")) |>
        head(20)
##                                keyword freq       pmi          md      lfmd
## 1                    do_Rio_de_Janeiro  207 13.018235 -0.85262461 -14.72348
## 2                    Rio_Grande_do_Sul  221 12.541865 -1.23457823 -15.01102
## 3         Conselho_Federal_de_Medicina  212 12.432902 -1.40352401 -15.23995
## 4                   de_Contas_da_União  257 12.226724 -1.33199735 -14.89072
## 5                do_Estado_do_Amazonas  201 10.846482 -3.06681226 -16.98011
## 6    Comissão_Parlamentar_de_Inquérito 2078 10.487430 -0.05593594 -10.59930
## 7            do_Presidente_da_República  613 10.004275 -2.30032848 -14.60493
## 8             o_Presidente_da_República  832  9.860730 -2.00317690 -13.86708
## 9          a_esta_Comissão_Parlamentar  229  9.523339 -4.20180370 -17.92695
## 10              no_Ministério_da_Saúde  706  8.842966 -3.25785565 -15.35868
## 11              ao_Ministério_da_Saúde  467  8.813761 -3.88330632 -16.58037
## 12            pelo_Ministério_da_Saúde  269  8.808331 -4.68455289 -18.17744
## 13              do_Ministério_da_Saúde 1640  8.787563 -2.09730346 -12.98217
## 14               o_Ministério_da_Saúde 1229  8.773717 -2.52735998 -13.82844
## 15    Muito_obrigado_Senhor_Presidente  257  7.326505 -6.23221721 -19.79094
## 16 Senhor_Presidente_Senhor_Presidente  494  6.941105 -5.67487400 -18.29085
## 17               Tribunal_de_Contas_da  257  5.441804 -8.11691778 -21.67564
## 18        esta_Comissão_Parlamentar_de  586  4.906807 -7.46278286 -19.83237
## 19       nesta_Comissão_Parlamentar_de  236  4.898397 -8.78330613 -22.46501
## 20       desta_Comissão_Parlamentar_de  462  4.898005 -7.81459206 -20.52719

# trigramas
PMI |> filter(ngram == 3)|>
        filter(freq > 200) |>
        arrange(-pmi) |>
        select(keyword, freq, pmi, md, lfmd ) |>
        filter(stringr::str_detect(keyword, "[:upper:][:lower:]+.*_[:upper:]")) |>
        head(20)
##                       keyword freq      pmi          md      lfmd
## 1              Rio_de_Janeiro  370 12.99440 -0.03863831 -13.07167
## 2    Senador_Fernando_Bezerra  320 12.62796 -0.61452582 -13.85701
## 3          Tribunal_de_Contas  429 12.59159 -0.22798867 -13.04757
## 4               Grande_do_Sul  225 12.54742 -1.20321276 -14.95385
## 5          Luis_Carlos_Heinze  229 12.38445 -1.34075961 -15.06597
## 6             Contas_da_União  258 12.22679 -1.32639465 -14.87958
## 7   Senador_Alessandro_Vieira  239 12.14072 -1.52283372 -15.18638
## 8     Deputado_Ricardo_Barros  231 12.12557 -1.58709681 -15.29976
## 9    Senador_Rogério_Carvalho  282 11.95172 -1.47314072 -14.89800
## 10         nos_Estados_Unidos  245 11.80901 -1.81876744 -15.44655
## 11          da_Prevent_Senior  311 11.76233 -1.52131956 -14.80496
## 12        Federal_de_Medicina  214 11.73059 -2.09235413 -15.91530
## 13        Senador_Luis_Carlos  218 11.70262 -2.09361400 -15.88985
## 14      Roberto_Ferreira_Dias  318 11.61286 -1.63867669 -14.89021
## 15          Senador_Omar_Aziz  267 11.54012 -1.96360333 -15.46732
## 16      Senador_Eduardo_Braga  276 11.45728 -1.99860731 -15.45450
## 17               de_São_Paulo  430 11.34779 -1.46843459 -14.28466
```

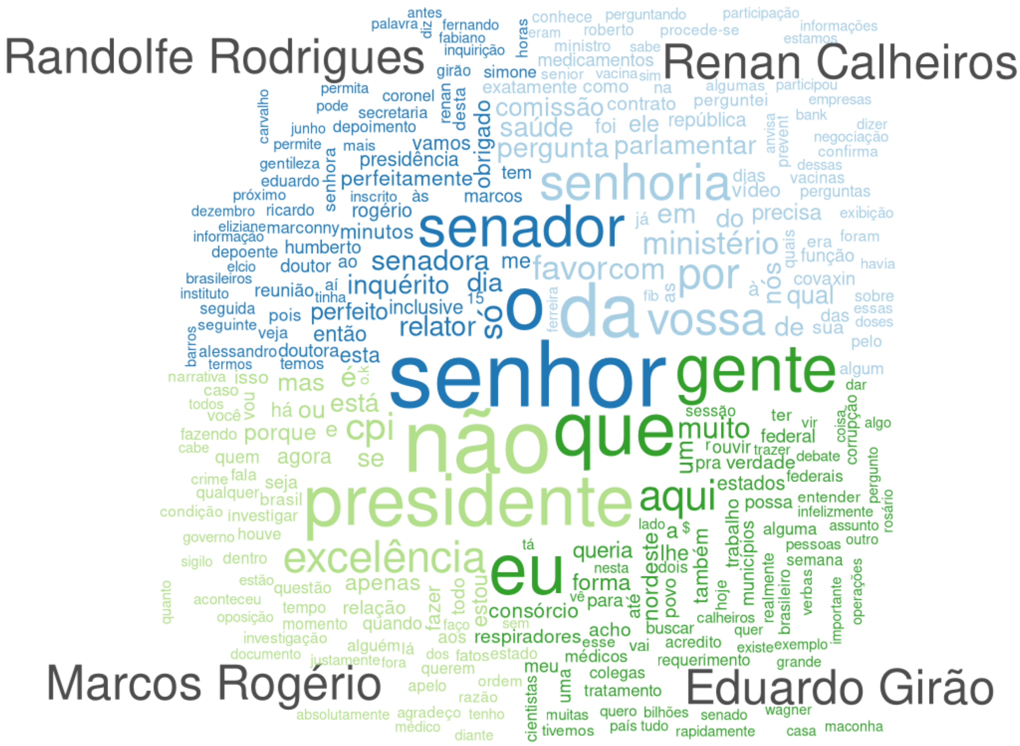
```
## 18       Estados_e_Municípios  602 11.26857 -1.06222933 -13.39303
## 19     Senador_Humberto_Costa  329 11.23187 -1.97059957 -15.17307
## 20 Senador_Randolfe_Rodrigues  292 10.88290 -2.49169065 -15.86628

# bigramas vamos ordenar pelo pmi
# (rodei antes, Por frequência não trazia bons resultados)
PMI |> filter(ngram == 2)|>
        arrange(-pmi) |> head(20) |>
        select(keyword, freq, pmi, md, lfmd )
##                keyword freq      pmi md      lfmd
## 1          Los_Angeles    3 19.97952  0 -19.97952
## 2           BCI_Balpex    3 19.97952  0 -19.97952
## 3     Evelyn_Beatrice    3 19.97952  0 -19.97952
## 4          Wesley_Cota    3 19.97952  0 -19.97952
## 5   Christian_Drosten    3 19.97952  0 -19.97952
## 6     Jandira_Feghali    3 19.97952  0 -19.97952
## 7       Thermo_Fisher    3 19.97952  0 -19.97952
## 8             Fla_Flu    3 19.97952  0 -19.97952
## 9        Von_Holleben    3 19.97952  0 -19.97952
## 10       Apoorv_Kumar    3 19.97952  0 -19.97952
## 11        Raman_Neves    3 19.97952  0 -19.97952
## 12      Goldman_Sachs    3 19.97952  0 -19.97952
## 13       Vick_VapoRub    3 19.97952  0 -19.97952
## 14       Yang_Wanming    3 19.97952  0 -19.97952
## 15 bebidas_alcoólicas    3 19.97952  0 -19.97952
## 16          these_are    3 19.97952  0 -19.97952
## 17      shelf_company    3 19.97952  0 -19.97952
## 18           I'm_here    3 19.97952  0 -19.97952
## 19     pó_liofilizado    3 19.97952  0 -19.97952
## 20         en_passant    3 19.97952  0 -19.97952

PMI |> filter(ngram == 2) |>
        arrange(-freq) |>
        #arrange(-lfmd) |>
        #arrange(-pmi) |> head(20) |>
        select(keyword, freq, pmi, md, lfmd ) |>
        # pegando palavras que comecem com maiúsculo
        filter(stringr::str_detect(keyword, "[:upper:][:lower:]+_[:upper:]")) |>
        head(20)
##                 keyword freq       pmi         md       lfmd
## 1      Vossa_Excelência 7496  7.798368 -0.8942114  -9.586791
## 2     Senhor_Presidente 5655  6.292253 -2.8069206 -11.906094
## 3        Vossa_Senhoria 4462  8.009776 -1.4312340 -10.872244
## 4  Comissão_Parlamentar 2110  9.438752 -1.0827057 -11.604163
## 5         Senador_Renan 1875  7.132737 -3.5590733 -14.250883
## 6       Governo_Federal 1472  7.777553 -3.2633704 -14.304293
## 7        Senador_Marcos 1221  6.885229 -4.4254088 -15.736046
## 8      Senador_Randolfe 1211  7.101369 -4.2211325 -15.543634
## 9        Senhor_Relator 1198  6.673727 -4.6643460 -16.002419
## 10       Marcos_Rogério 1104 10.223431 -1.2325290 -12.688489
## 11     Senador_Humberto  895  7.184209 -4.5745321 -16.333273
## 12    Presidente_Senhor  869  3.591763 -8.2095093 -20.010782
## 13       Prevent_Senior  791 11.758741 -0.1782100 -12.115161
## 14        Senador_Girão  785  6.710516 -5.2374201 -17.185356
## 15            São_Paulo  739 10.402014 -1.6330400 -13.668094
```

```
## 16      Senador_Eduardo  730  6.673275 -5.3794571 -17.432189
## 17      Renan_Calheiros  722 10.435552 -1.6330783 -13.701708
## 18   Senhores_Senadores  708 10.241183 -1.8556965 -13.952576
## 19       Senhor_Senador  672  3.160043 -9.0121245 -21.184292
## 20      Polícia_Federal  655  9.247164 -2.9619696 -15.171103
```

4.3 Wordcloud comparision

Vamos usar o Quanteda para alguns gráficos. Para tal, devemos primeiro criar um objeto tipo corpus, fazer alguma restrição/filtragem, tokenizar. Alguns procedimentos exigem que se converta para Document Term Matrix.

```
library("quanteda.textplots")
#senCorpus <- corpus(NT_falasJuntasCount[2:5,],
#      docid_field = "nome",
#      text_field = "falas")
# Criando um corpus
senCorpus <- corpus(NT_falasJuntasCount,
       docid_field = "nome",
       text_field = "falas")
# Pegando a fala de senadores
senTokens <- tokens(corpus_subset(senCorpus[c(2:5)]))
# senTokens <- tokens(senCorpus)

dfmat2 <- dfm(senTokens,
    dfm_remove = stopwords("portuguese"), remove_punct = TRUE, dfm.group = "nome")
%>% dfm_trim(min_termfreq = 3)
## Warning: '...' should not be used for tokens() arguments; use 'tokens()'
## first.
## Warning: dfm_remove, dfm.group arguments are not used.

## Warning: dfm_remove, dfm.group arguments are not used.

textplot_wordcloud(dfmat2, comparison = TRUE, max_words = 300)
## Warning in wordcloud_comparison(x, min_size, max_size, min_count,
## max_words, : oportunidade could not be fit on page. It will not be
## plotted.
## Warning in wordcloud_comparison(x, min_size, max_size, min_count,
## max_words, : randolfe could not be fit on page. It will not be plotted.
## Warning in wordcloud_comparison(x, min_size, max_size, min_count,
## max_words, : gestão could not be fit on page. It will not be plotted.
```



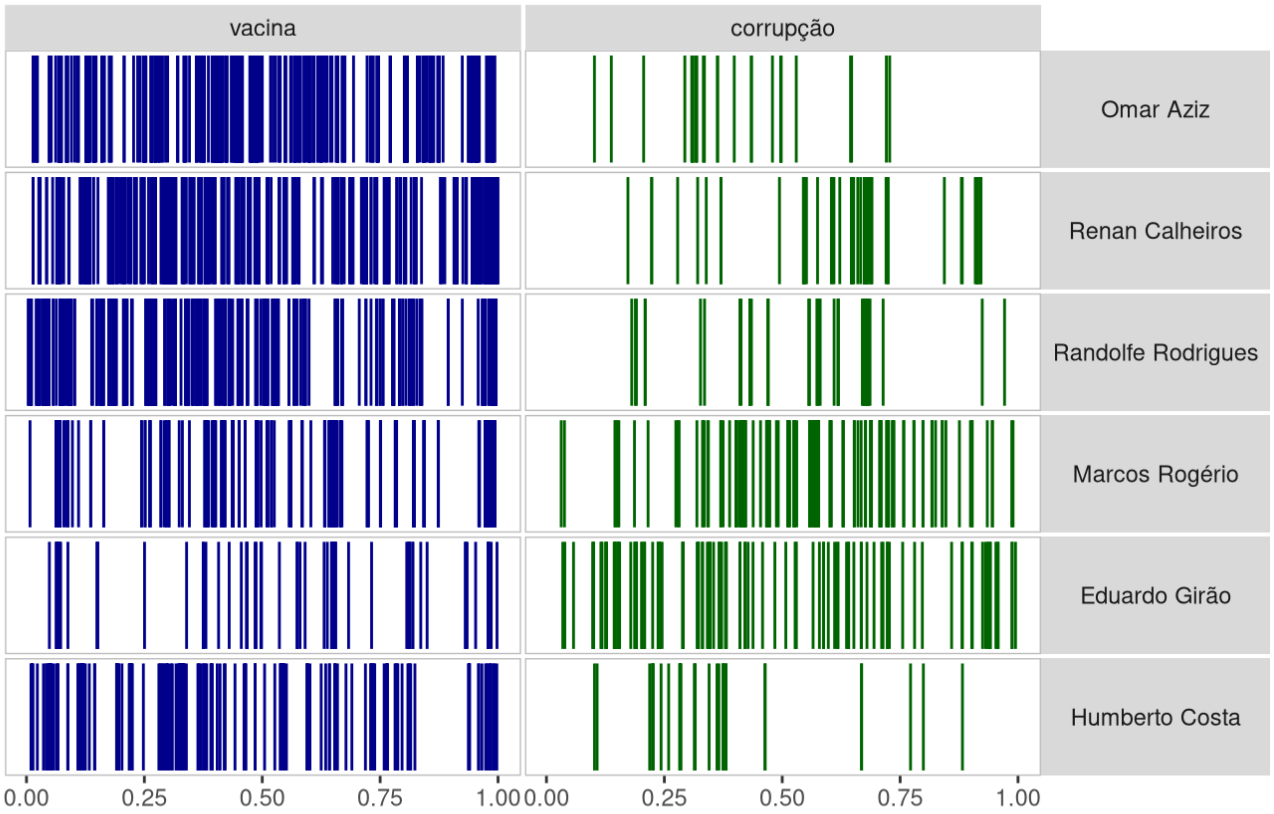
O Gráfico de dispersão lexical (*Lexical dispersion plot*) mostra como um ou mais termos aparecem ao longo de um documento. No caso aqui, o documento é a fala de cada senador ao longo da CPI. A plotagem demora alguns bons minutos para rodar, mais que topic modeling.

```
# Vamos restringir o número de pessoas para nosso gráfico
senTokens <- tokens(corpus_subset(senCorpus[c(1:6)]))

# Gerando o gráfico. Esta parte é demorada
grafico <- textplot_xray(
              kwic(senTokens, pattern = "vacina"),
              kwic(senTokens, pattern = "corrupção"))
```

O objeto gerado é um ggplot. Podemos personalizá-lo

```
grafico + aes(color = keyword) +
        scale_color_manual(values = c("darkblue", "darkgreen")) +
        theme(legend.position = "none") +
        #ggtitle("Dispersão Lexical")
        labs(title = "Gráfico de dispersão lexical",
             x="Aparição dos termos ao longo da fala de cada um na CPI", y="",
             caption = "Elaboração: Alisson Soares")
```

Gráfico de dispersão lexical



Elaboração: Alisson Soares

4.3.1 TF-IDF dos Senadores

```
# Observando novamente a estrutura do df que criamos só com os senadores
str(senadores_falasJuntas)
## tibble [42 × 3] (S3: tbl_df/tbl/data.frame)
##  $ nome      : chr [1:42] "Omar Aziz" "Renan Calheiros" "Randolfe Rodrigues" "Ma
rcos Rogério" ...
##  $ N_palavras: int [1:42] 413543 351349 341887 196716 146154 138842 125590 12462
6 120231 114925 ...
##  $ falas     : chr [1:42] "Como é que é? Peço só um minutinho, só um minutinho!
Senhor Presidente...  Eu acho que Vossa Excelência.. Nós "| __truncated__ "Quer diz
er que há outros impedimentos a serem...  Acredito não ser o caso de VossaExcelênci
a, mas o Estado de A"| __truncated__ "Presidente...  Presidente, qual a ordem?  Pre
sidente, só para declinar a ordem, quem são?  Agora é a Eliziane? "| __truncated__
"Senhor Presidente, Senhoras e Senhores Senadores, faço a presente questão de orde
m, Senhor Presidente, desde lo"| __truncated__ ...
str(NT_falasJuntasCount)
## tibble [154 × 3] (S3: tbl_df/tbl/data.frame)
##  $ nome      : chr [1:154] "Omar Aziz" "Renan Calheiros" "Randolfe Rodrigues" "M
arcos Rogério" ...
##  $ N_palavras: int [1:154] 413543 351349 341887 196716 146154 138842 125590 1246
26 120231 114925 ...
##  $ falas     : chr [1:154] "Como é que é? Peço só um minutinho, só um minutinho!
Senhor Presidente...  Eu acho que Vossa Excelência.. Nós "| __truncated__ "Quer diz
er que há outros impedimentos a serem...  Acredito não ser o caso de VossaExcelênci
a, mas o Estado de A"| __truncated__ "Presidente...  Presidente, qual a ordem?  Pre
sidente, só para declinar a ordem, quem são?  Agora é a Eliziane? "| __truncated__
"Senhor Presidente, Senhoras e Senhores Senadores, faço a presente questão de orde
m, Senhor Presidente, desde lo"| __truncated__ ...
```

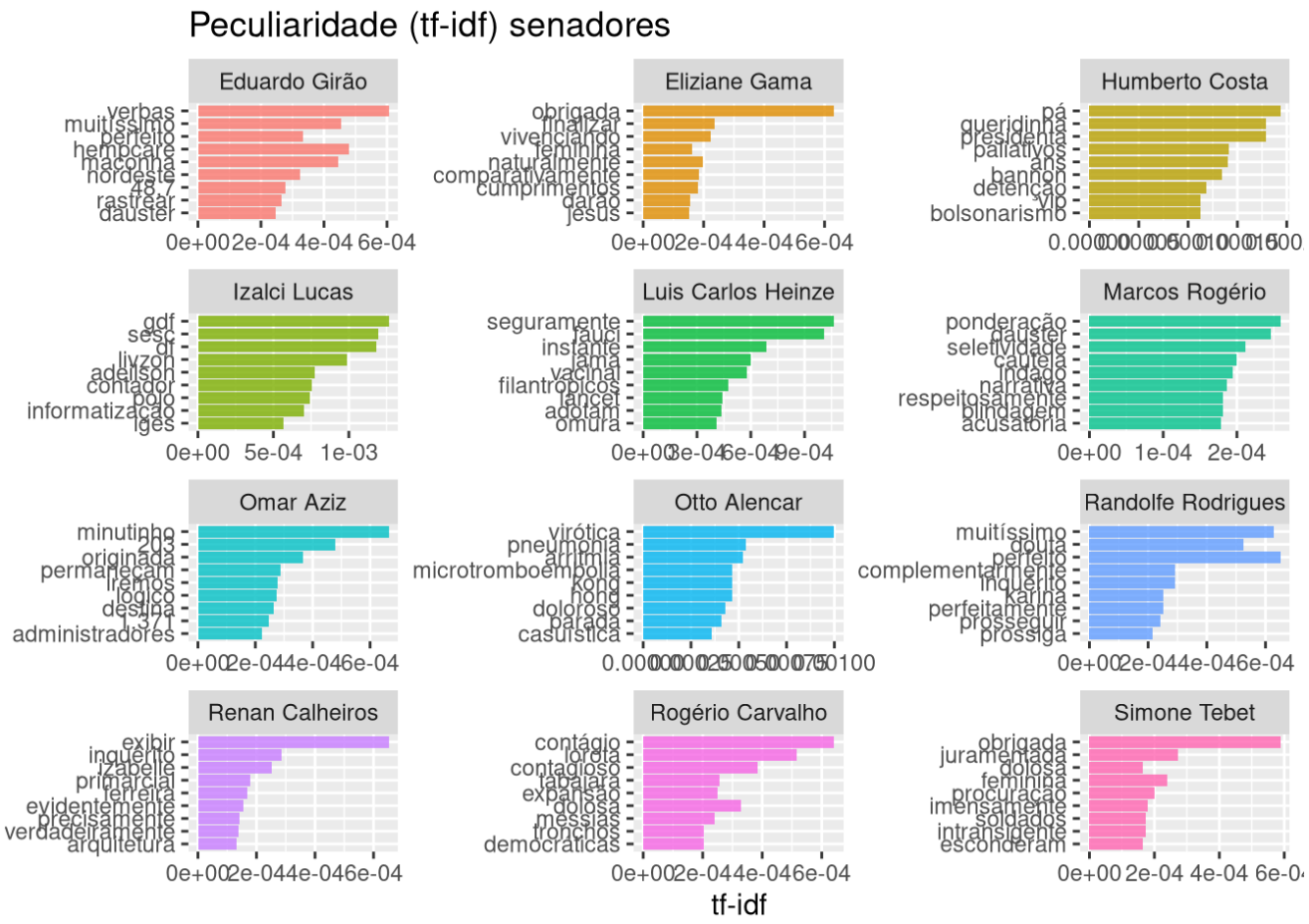
Vemos que temos 42 senadores que falaram na CPI. Vamos restringir apenas aos 20 que mais falaram.

```
top <- arrange(NT_falasJuntasCount, desc(N_palavras)) |> head(12)
top.tfidf <- top |> tidytext::unnest_tokens(
  output = 'word', token = 'words',  input = falas) |>  # contando os termos
  dplyr::count(nome, word, sort = TRUE) %>%
  tidytext::bind_tf_idf(word, nome, n) |>
  arrange(desc(tf_idf))
top.tfidf
## # A tibble: 104,600 × 6
##    nome               word            n       tf   idf   tf_idf
##    <chr>              <chr>       <int>    <dbl> <dbl>    <dbl>
##  1 Izalci Lucas       gdf            55 0.000912 1.39  0.00126
##  2 Izalci Lucas       sesc           29 0.000481 2.48  0.00119
##  3 Izalci Lucas       df            132 0.00219  0.539 0.00118
##  4 Luis Carlos Heinze seguramente    39 0.000428 2.48  0.00106
##  5 Luis Carlos Heinze fauci          37 0.000406 2.48  0.00101
##  6 Otto Alencar       virótica       28 0.000402 2.48  0.000998
##  7 Izalci Lucas       livzon         24 0.000398 2.48  0.000989
##  8 Izalci Lucas       adeilson       26 0.000431 1.79  0.000772
##  9 Izalci Lucas       contador       52 0.000862 0.875 0.000755
## 10 Izalci Lucas       pojo           18 0.000298 2.48  0.000742
## # … with 104,590 more rows
```

Buscando os termos mais específicos por pessoa

```
nome.senador <- unique(top.tfidf$nome)
nome.senador
##  [1] "Izalci Lucas"       "Luis Carlos Heinze" "Otto Alencar"       "Omar Aziz"
##  [5] "Renan Calheiros"    "Randolfe Rodrigues" "Rogério Carvalho"   "Eliziane Ga
ma"
##  [9] "Eduardo Girão"      "Simone Tebet"       "Marcos Rogério"     "Humberto Co
sta"

top.tfidf.pessoa <-  top.tfidf |> filter(nome == nome.senador[2]) |>
                              arrange(-tf_idf)

# Como o tibble normalmente mostra apenas as primeiras linhas
# vamos mostrar mais linhas com o comando `print()`
top.tfidf.pessoa |> print(n=30)
## # A tibble: 7,758 × 6
##    nome               word              n       tf   idf   tf_idf
##    <chr>              <chr>         <int>    <dbl> <dbl>    <dbl>
##  1 Luis Carlos Heinze seguramente       39 0.000428 2.48  0.00106
##  2 Luis Carlos Heinze fauci             37 0.000406 2.48  0.00101
##  3 Luis Carlos Heinze instante          45 0.000494 1.39  0.000685
##  4 Luis Carlos Heinze jama              22 0.000242 2.48  0.000600
##  5 Luis Carlos Heinze vacinal           48 0.000527 1.10  0.000579
##  6 Luis Carlos Heinze filantrópicos     24 0.000263 1.79  0.000472
##  7 Luis Carlos Heinze lancet            46 0.000505 0.875 0.000442
##  8 Luis Carlos Heinze adotam            36 0.000395 1.10  0.000434
##  9 Luis Carlos Heinze omura             15 0.000165 2.48  0.000409
## 10 Luis Carlos Heinze harvard           26 0.000285 1.39  0.000396
## 11 Luis Carlos Heinze reposicionados    20 0.000220 1.79  0.000393
## 12 Luis Carlos Heinze satoshi           14 0.000154 2.48  0.000382
## 13 Luis Carlos Heinze letalidade       119 0.00131  0.288 0.000376
## 14 Luis Carlos Heinze ribeirão          24 0.000263 1.39  0.000365
## 15 Luis Carlos Heinze veterinários      22 0.000242 1.39  0.000335
## 16 Luis Carlos Heinze nobel             34 0.000373 0.875 0.000327
## 17 Luis Carlos Heinze amapá             55 0.000604 0.539 0.000325
## 18 Luis Carlos Heinze fagundes          16 0.000176 1.79  0.000315
## 19 Luis Carlos Heinze zelenko           25 0.000274 1.10  0.000302
## 20 Luis Carlos Heinze 632               11 0.000121 2.48  0.000300
## 21 Luis Carlos Heinze cimatec           11 0.000121 2.48  0.000300
## 22 Luis Carlos Heinze luc               11 0.000121 2.48  0.000300
## 23 Luis Carlos Heinze mcti              11 0.000121 2.48  0.000300
## 24 Luis Carlos Heinze montagnier        11 0.000121 2.48  0.000300
## 25 Luis Carlos Heinze surgisphere       11 0.000121 2.48  0.000300
## 26 Luis Carlos Heinze preto             24 0.000263 1.10  0.000289
## 27 Luis Carlos Heinze distribuídas      29 0.000318 0.875 0.000279
## 28 Luis Carlos Heinze usp               27 0.000296 0.875 0.000259
## 29 Luis Carlos Heinze pharma            21 0.000231 1.10  0.000253
## 30 Luis Carlos Heinze letal             26 0.000285 0.875 0.000250
## # … with 7,728 more rows
```

```
top.tfidf |> group_by(nome) |>
        # top_n para definir o número de tópicos por gráfico
        top_n(9, tf_idf) |> ungroup() |>
  ggplot(aes(reorder(word, tf_idf), tf_idf, fill = nome)) +
  geom_bar(stat = "identity", alpha = .8, show.legend = FALSE) +
  labs(title = "Peculiaridade (tf-idf) senadores",
       x = NULL, y = "tf-idf") +
  facet_wrap(~nome, ncol = 3, scales = "free") +
  coord_flip()
```



5 Dicionário

5.1 Matrizes DTM DFM

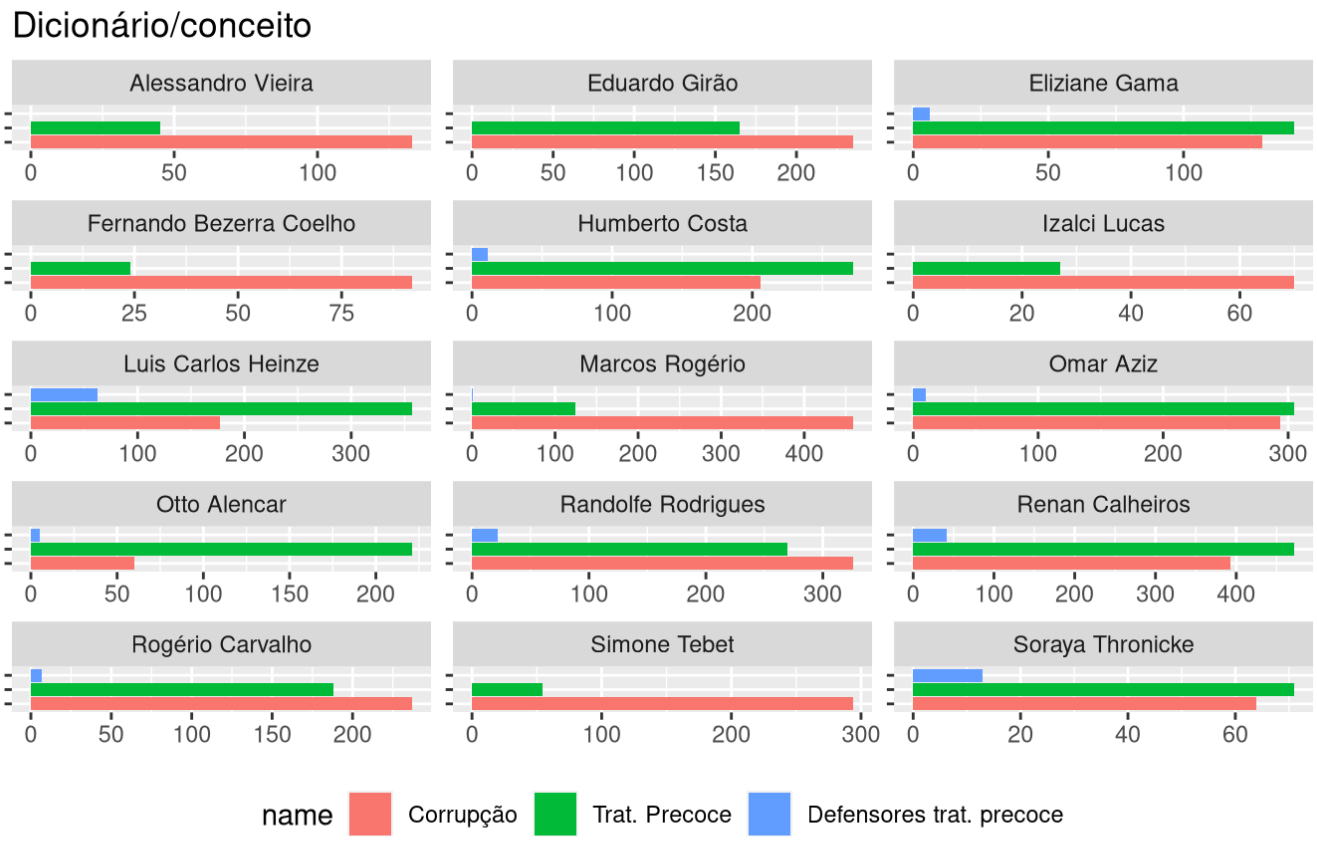
A conversão em matrizes e dessa para DTM é um passo intermediário crucial em diversas abordagens de análise textual, como TF-IDF e Topic Modeling.

Vamos gerar uma métrica de conceitos a partir de de dicionários de termos. Para cada conceito, escolhi um conjunto de palavras. Vale lembrar que tanto as categorias como as palavras destas foram escolhidas de modo rápido, não sistemátrico. O vetor de vacinas foi comentado pois rodei previamente, os termos relacionados a vacina predominavam em todos os casos.

```
library(quanteda)

cpi.corpus <- corpus(NT_falasJuntasCount, docid_field = "nome", text_field = "fala
s")

# Criando um Document Term Matrix
# o processo abaixo demora um pouco
cpi.dfm <- tokens(cpi.corpus, remove_punct = TRUE) %>%
  dfm() %>%
  dfm_remove(pattern = SW )
# utilizando a lista de stopwords que criamos anteriormente
#tokens_select(pattern = SW, selection = "remove")

dict <- dictionary(list(
  tratPre = c("cloroquina", "ivermectina", "azitromicina", "Kit", "precoce", "ozôni
o"),
  tratPre.defensores = c("Raoult", "Zelen[ck]o", "Yamagushi", "Zebalos", "Wong", "Z
anotto"),
  # vacinas = c("vacinas?", "CoronaVac", "Butantan", "AstraZeneca", "Oxford", "Comi
rnaty", "BioNTech", "Pfizer", "Janss?en", "Johnson", "Spikevax", "Moderna", "Sputni
k", "Gamaleya"),
  corrupcao = c("corrup.*", "propin.*", "superfatur.*", "prevaric.*", "crim[ei].*",
"fraud.*", "lava[ng].*")))
  # corrupcao = c("corrup.*", "propin.*", "superfatur.*", "prevaric.*")))
# rodando nosso dicionário
dict_dtm <- dfm_lookup(cpi.dfm,
                       dictionary = dict,
                       valuetype = "regex",
                       #nomatch = "_unmatched")
                       nomatch = "_semCorrespondencia")
# supondo que nosso dicionário seja minimamente bom
# vendo o quanto senadores abordaram certos temas
dict_dtm[5:10,]
## Document-feature matrix of: 6 documents, 4 features (8.33% sparse) and 1 docvar.
##                       features
## docs                   tratPre tratPre.defensores corrupcao
##   Eduardo Girão            165                   0       235
##   Humberto Costa           272                  11       206
##   Simone Tebet              54                   0       294
##   Eliziane Gama            141                   6       129
##   Rogério Carvalho         188                   7       237
##   Luis Carlos Heinze       357                  62       177
##                       features
## docs                   _semCorrespondencia
##   Eduardo Girão                  61409
##   Humberto Costa                 56989
##   Simone Tebet                   51434
##   Eliziane Gama                  49495
##   Rogério Carvalho               49953
##   Luis Carlos Heinze             49609
```

Gerando um ggplot

```
# retirar a ultima coluna que não nos é útil
dict.df  <- convert(dict_dtm[1:15,-5], to = "data.frame") %>%
  tidyr::pivot_longer(.,
                      cols = names(dict),
                      values_to = "Valores")

# vamos renomear os labels com os seguintes rótulos
rotulos <- c( "Corrupção", "Trat. Precoce", "Defensores trat. precoce")

ggplot(dict.df, aes(x = name, y = Valores, fill = name ) ) +
  geom_col() +
  labs(title = "Dicionário/conceito", x = NULL, y = NULL,
       caption = "Elaboração: Alisson Soares\nObservação: Cada pessoa está em uma e
scala diferente.\nO gráfico não se trata de análise empírica, mas de demonstração d
as ferramentas de análise",
              text = element_text("Temáticas:")) +
  # retirando a legenda do canto direito
  theme(legend.position="bottom") +
        # Mudando os nomes das variáveis na legenda
        scale_fill_discrete(labels = rotulos) +
        #scale_fill_manual(labels = rotulos) +
        # Retirar os rótulos
        scale_x_discrete(labels = NULL) +
  facet_wrap(~doc_id, ncol = 3, scales = "free") +
  # rotacionando o gráfico
  coord_flip()
```



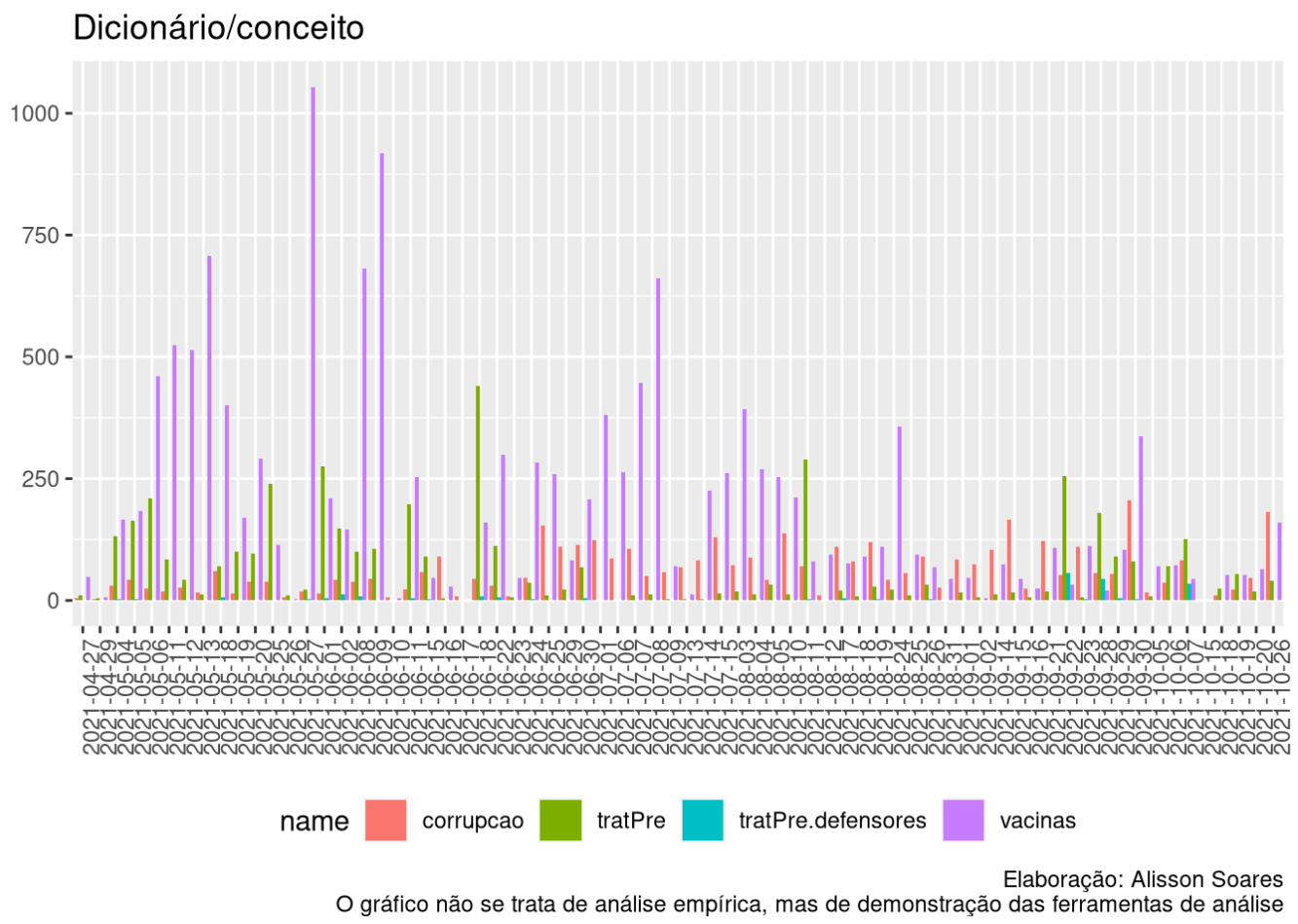
Vamos analisar com dicionário, agora longitudinalmente.

```
NT.longitudinal <- NotasTaq2 %>%
        group_by(reuniao, data,) %>%
        summarise(falas = paste(fala, collapse = " "))
## `summarise()` has grouped output by 'reuniao'. You can override using the
## `.groups` argument.
head(NT.longitudinal)
## # A tibble: 6 × 3
## # Groups:   reuniao [6]
##   reuniao data       falas
##     <dbl> <date>     <chr>
## 1       1 2021-04-27 "Invocando a proteção de Deus, declaro aberta a sess…
## 2       2 2021-04-29 "Havendo número regimental, declaro aberta a 2ª Reun…
## 3       3 2021-05-04 "Bom dia. Havendo número regimental, declaro aberta …
## 4       4 2021-05-05 "Bom dia. Havendo número regimental, declaro aberta …
## 5       5 2021-05-06 "Bom dia! Havendo número regimental, declaro aberta …
## 6       6 2021-05-11 "Havendo número regimental, declaro aberta a 6ª Reun…
```

```
# criando corpus
corpus.longi <- corpus(NT.longitudinal, docid_field = "data", text_field = "falas")

# Criando um Document Term Matrix
# o processo abaixo demora um pouco
dfm.longi <- tokens(corpus.longi , remove_punct = TRUE) %>%
  dfm() %>%
  dfm_remove(pattern = SW )
# utilizando a lista de stopwords que criamos anteriormente
#tokens_select(pattern = SW, selection = "remove")


dict <- dictionary(list(
  tratPre = c("cloroquina", "ivermectina", "azitromicina", "Kit", "precoce", "ozôni
o"),
  tratPre.defensores = c("Raoult", "Zelen[ck]o", "Yamagushi", "Zebalos", "Wong", "Z
anotto"),
  vacinas = c("vacinas?", "CoronaVac", "Butantan", "AstraZeneca", "Oxford", "Comirn
aty", "BioNTech", "Pfizer", "Janss?en", "Johnson", "Spikevax", "Moderna", "Sputni
k", "Gamaleya"),
  corrupcao = c("corrup.*", "propin.*", "superfatur.*", "prevaric.*", "crim[ei].*",
"fraud.*", "lava[ng].*")))

# rodando nosso dicionário
dict_dtm <- dfm_lookup(dfm.longi,
                       dictionary = dict,
                       valuetype = "regex",
                       #nomatch = "_unmatched")
                       nomatch = "_semCorrespondencia")
# supondo que nosso dicionário seja minimamente bom
dict_dtm[5:10,]
## Document-feature matrix of: 6 documents, 5 features (6.67% sparse) and 1 docvar.
##             features
## docs         tratPre tratPre.defensores vacinas corrupcao
##   2021-05-06     210                  1     461        24
##   2021-05-11      85                  0     523        19
##   2021-05-12      43                  0     514        26
##   2021-05-13      13                  1     707        17
##   2021-05-18      70                  6     400        60
##   2021-05-19     100                  1     169        14
##             features
## docs         _semCorrespondencia
##   2021-05-06               34559
##   2021-05-11               24078
##   2021-05-12               33293
##   2021-05-13               22602
##   2021-05-18               29293
##   2021-05-19               23434
```

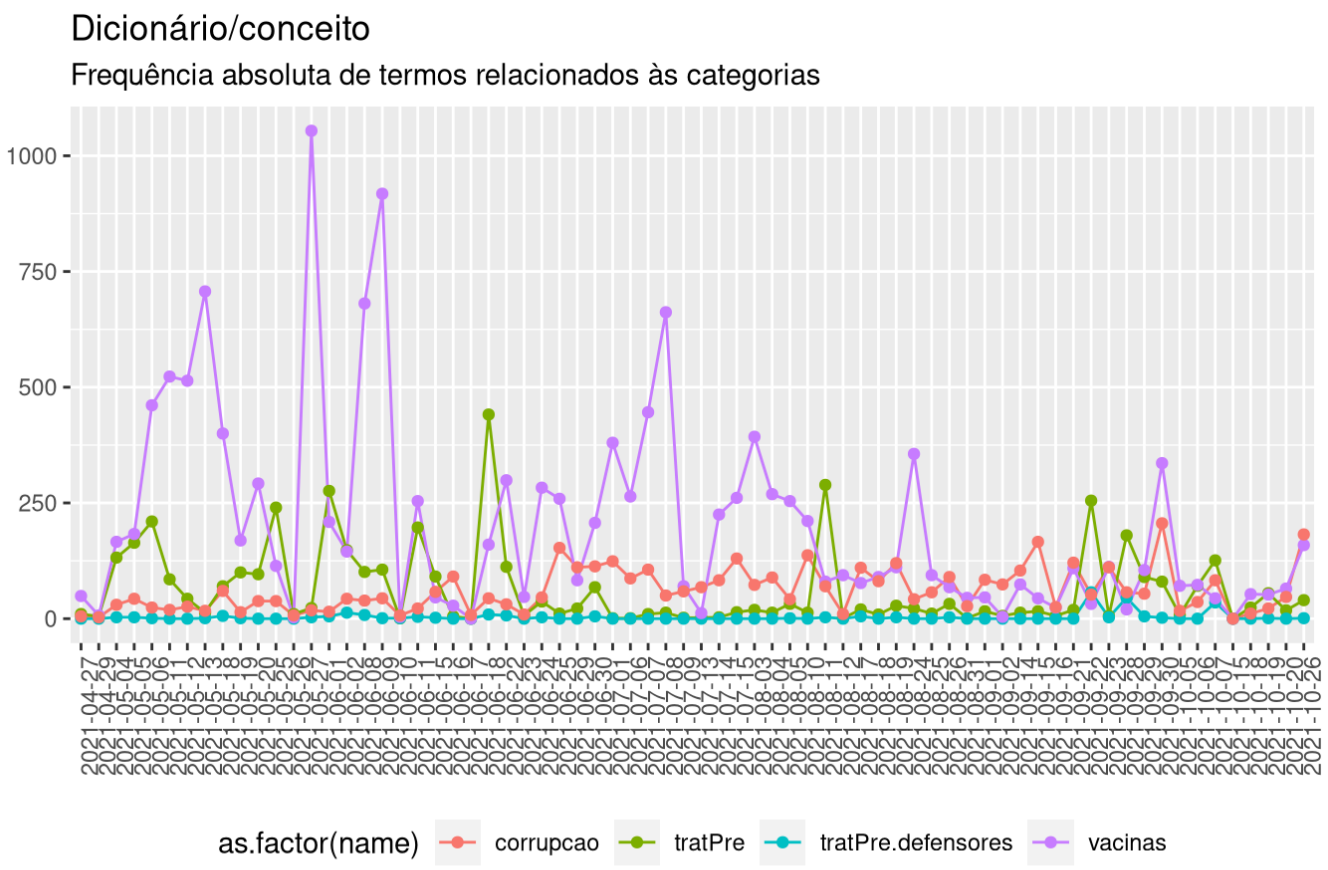
Criando um ggplot

```
dict.df  <- convert(dict_dtm, to = "data.frame") %>%
  tidyr::pivot_longer(.,
                      cols = names(dict),
                      values_to = "Valores")

ggplot(dict.df, aes(x = doc_id, y = Valores, fill = name ) ) +
  geom_col(position = "dodge") +
  labs(title = "Dicionário/conceito", x = NULL, y = NULL,
       caption = "Elaboração: Alisson Soares\nO gráfico não se trata de análise emp
írica, mas de demonstração das ferramentas de análise") +
  theme(legend.position="bottom", axis.text.x = element_text(angle = 90))
```



E os mesmos dados, mas em gráfico de linhas:

```
ggplot(dict.df, aes(x = doc_id, y = Valores) ) +
  geom_line(aes(group=name,
                color=as.factor(name),
                #linetype=as.factor(name)
                )) +
   geom_point(aes(group=name,
                  color=as.factor(name))) +
  labs(title = "Dicionário/conceito", subtitle = "Frequência absoluta de termos rel
acionados às categorias", x = NULL, y = NULL,
       caption = "Elaboração: Alisson Soares\nO gráfico não se trata de análise emp
írica, mas de demonstração das ferramentas de análise", fill="Conceito") +
  theme(legend.position="bottom", axis.text.x = element_text(angle = 90))
```



Elaboração: Alisson Soares
O gráfico não se trata de análise empírica, mas de demonstração das ferramentas de análise

```
 #scale_fill_manual('Legend Title', values=c('corrupção', 'Trat Prec', 'Defensores Trat. Prec.', 'Vacinas'))
```

5.2 Análise de coocorrência

Ao utilizar dicionário, sempre há o perigo de não usarmos os termos mais adequados. Podemos incorporar termos pouco relevantes e deixar de lado termos importantes. Um modo de lidar com este problema se dá com análise de coocorrência de termos. Com a função `pairwise_count` vamos contar o número de vezes que cada par de termos aparecem por pessoa. A análise de coocorrência pode multiplicar nossos problemas, uma vez que pode gerar tabela gigantesca e com isto o processamento pode ficar extremamente lento. Para se ter uma ideia, fazer análise de coocorrência de um senador pode gerar um tibble de quase 200 milhões de coocorrências. Teremos de criar um pipe de comandos especial para este tipo de análise

```
# Utilizando todas as falas.

# Tokenizando por sentença
senadores_tokens <- senadores_falasJuntas |>
        select(nome, falas) |>
        tidytext::unnest_tokens(palavras, falas, token = "sentences")
# Filtrando sentenças com termos que nos interessam
# termos <- "vacin|corrup"
termos <- "corrup"
senadores_tokens2 <-  senadores_tokens %>%
        filter(grepl(termos, .$palavras))
# tokenizar novamente, desta vez por palavra
senadores_tokens3 <- senadores_tokens2 |>
        tidytext::unnest_tokens(words, palavras) |>
        anti_join(SW.df) # retirar stopwords
## Joining, by = "words"

cooc_pessoa <- senadores_tokens3 |>
        pairwise_count(words, nome, sort = TRUE) |>
        arrange(-n)
# arrange(cooc_pessoa, -n)
filtrado <- cooc_pessoa %>% filter(grepl(termos, .$item1))

# termos buscados e respectiva frequência
termos.encontrados <- filtrado$item1 %>% plyr::count() |> arrange(-freq)
termos.encontrados
##               x freq
## 1     corrupção 3423
## 2      corrupto 2368
## 3     corruptos 1749
## 4   corruptores 1238
## 5 anticorrupção 1086
## 6      corrupta 1045
## 7     corruptas  603
## 8    corrupções   14

# examinando os termos mais frequentes correlacionados aos termos buscado
filtrado$item2[1:150]
##   [1] "governo"          "presidente"       "senhor"
##   [4] "cpi"              "gente"            "dinheiro"
##   [7] "federal"          "brasil"           "todos"
##  [10] "ministério"       "saúde"            "senador"
##  [13] "pessoas"          "ter"              "hoje"
##  [16] "agora"            "bolsonaro"        "ser"
##  [19] "pode"             "vacina"           "ainda"
##  [22] "brasileiros"      "público"          "sobre"
##  [25] "pandemia"         "esquema"          "qualquer"
##  [28] "disse"            "dentro"           "todo"
##  [31] "contra"           "aí"               "passiva"
##  [34] "toda"             "momento"          "crime"
##  [37] "combate"          "ativa"            "corrupção"
##  [40] "caso"             "desvio"           "recursos"
##  [43] "fazer"            "corrupto"         "vossa"
##  [46] "dizer"            "indícios"         "presidente"
```

```
##  [49] "brasileiro"       "desta"           "neste"
##  [52] "tentativa"        "forma"           "falar"
##  [55] "coisa"            "senhor"          "governo"
##  [58] "estado"           "gente"           "vamos"
##  [61] "lá"               "verdade"         "meio"
##  [64] "cada"             "vacinas"         "tudo"
##  [67] "comissão"         "ministro"        "acho"
##  [70] "parte"            "dessa"           "relação"
##  [73] "grande"           "crimes"          "maior"
##  [76] "públicos"         "senadores"       "claro"
##  [79] "todas"            "investigação"    "desse"
##  [82] "tipo"             "anos"            "responsabilidade"
##  [85] "administrativa"   "casos"           "hoje"
##  [88] "onde"             "sim"             "ainda"
##  [91] "denúncia"         "dias"            "infelizmente"
##  [94] "menos"            "precisa"         "nesse"
##  [97] "república"        "esquema"         "país"
## [100] "empresa"          "nenhuma"         "exemplo"
## [103] "milhões"          "nessa"           "alguns"
## [106] "tempo"            "quero"           "assim"
## [109] "processo"         "senador"         "girão"
## [112] "disse"            "bem"             "fazendo"
## [115] "nome"             "pode"            "dessas"
## [118] "investigar"       "inclusive"       "faz"
## [121] "relator"          "pedido"          "cpi"
## [124] "outras"           "brasileira"      "bolsonaro"
## [127] "lá"               "ministério"      "nada"
## [130] "dinheiro"         "importante"      "diz"
## [133] "fatos"            "ver"             "senhoria"
## [136] "nesta"            "público"         "polícia"
## [139] "área"             "além"            "vai"
## [142] "outro"            "algum"           "fraude"
## [145] "apenas"           "algo"            "vezes"
## [148] "vez"              "fala"            "dentro"
```

Podemos utilizar esta busca de termos que co ocorreram com termos relacionados à corrupção (ou outros termos de preferência), para com isto encontrar termos novos relacionados a esta temática, que não constam no dicionário que usamos anteriormente.

```
library(udpipe)
valor_skipgram <- 5
cooc_pessoas <- udpipe::cooccurrence(senadores_tokens3,
                      # group: nome da coluna com "id"
                      group = "nome",
                      # term: coluna com palavras a serem contadas
                      term = "words",
                      skipgram = valor_skipgram )

cooc.corrup <- cooc_pessoas %>% filter(grepl("corrup", .$term1))



cooc.corrup[1:30,] |> ggplot( aes(x=item2, y=n))+
        geom_col() +
        # rotacionando o gráfico
        coord_flip()
        labs(title = 'Palavras que coocorreram com termo "corrupção"')

ggraph::ggraph(cooc.corrup[1:70,]  , layout = "fr") +
  geom_edge_link(aes(width = n, edge_alpha = n), edge_colour = "lightskyblue") +
  geom_node_text(aes(label = name), col = "darkgreen", size = 4)
```

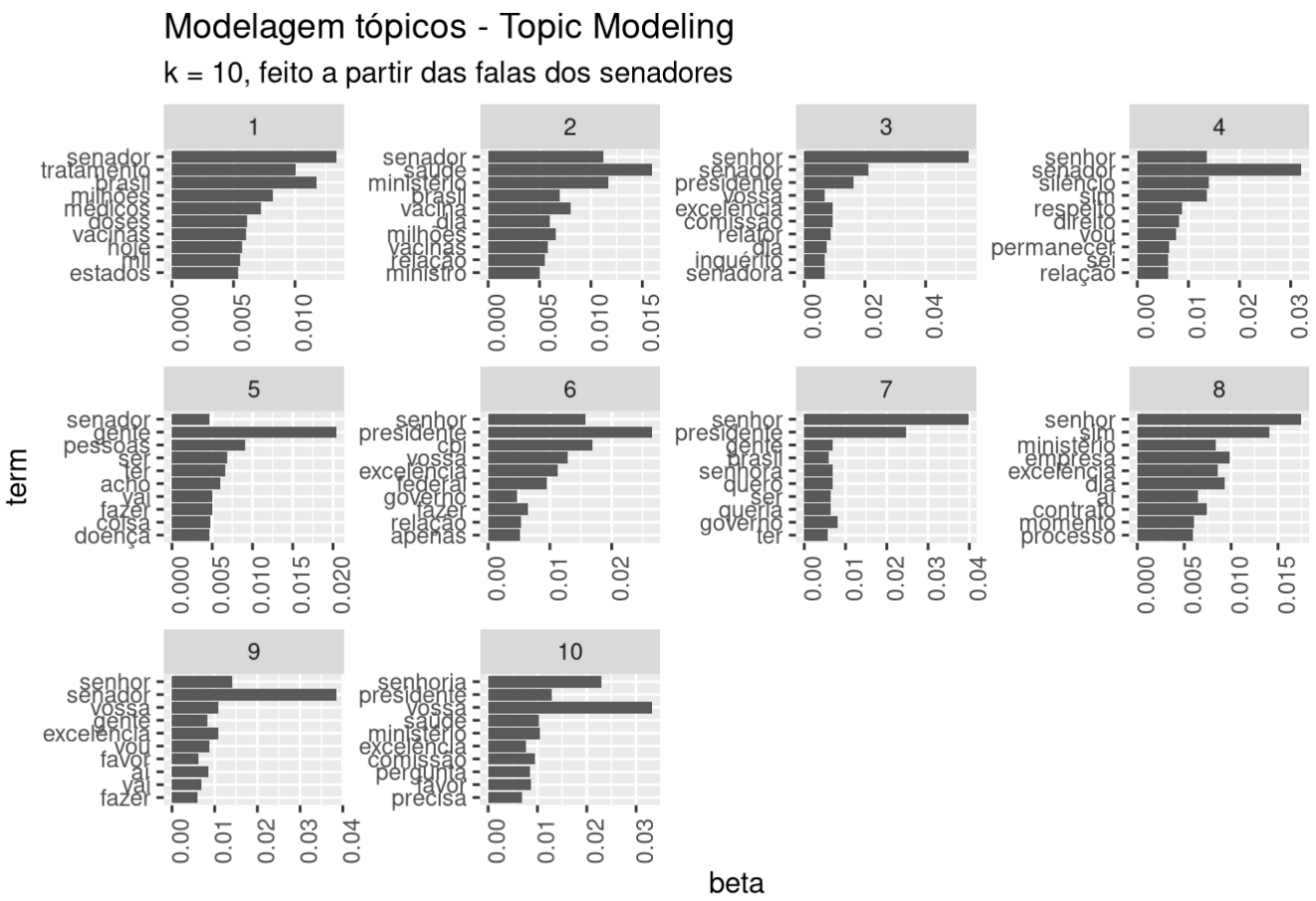
5.3 Modelagem de tópicos (*Topic Modelling*)

A modelagem de tópicos pode demorar, dependendo das configurações de sua máquina e do tamanho do seu corpus a ser processado. A função `pryr::object_size(par_dtm)` nos retornou que nosso objeto possui 7.735.496 ou 7.986.840 B, ou 7,6 Mb de tamanho. Usando a função `system.time(funcao)` é possível medir o tempo gasto por determinada tarefa. Assim, um computador i5 com 8Gb de Ram demorou 662.695 segundos - cerca de 11 minutos - para rodar esta modelagem de tópicos de um arquivo de 7,6 Mb.

```
library(topicmodels)
texts <- corpus_reshape(cpi.corpus, to = "paragraphs")
#par_dtm <- dfm(texts, stem = TRUE,
# create a document-term matrix
#       remove_punct = TRUE,
# remove = stopwords("english")
par_dtm <- dfm_trim(cpi.dfm, min_count = 5)
# remove rare terms
par_dtm <- convert(par_dtm, to = "topicmodels") # convert to topicmodels format
set.seed(1)
valor_k <- 10
lda_model <- topicmodels::LDA(par_dtm, method = "Gibbs", k = valor_k)
# vendo nossa modelagem de tópicos, 5 primeiras linhas
terms(lda_model, 5)
##      Topic 1      Topic 2      Topic 3      Topic 4    Topic 5
## [1,] "senador"    "saúde"      "senhor"     "senador"  "gente"
## [2,] "brasil"     "ministério" "senador"    "silêncio" "pessoas"
## [3,] "tratamento" "senador"    "presidente" "senhor"   "ser"
## [4,] "milhões"    "vacina"     "excelência" "sim"      "ter"
## [5,] "médicos"    "brasil"     "comissão"   "respeito" "acho"
##      Topic 6      Topic 7      Topic 8      Topic 9      Topic 10
## [1,] "presidente" "senhor"     "senhor"     "senador"    "vossa"
## [2,] "cpi"        "presidente" "sim"        "senhor"     "senhoria"
## [3,] "senhor"     "governo"    "empresa"    "vossa"      "presidente"
## [4,] "vossa"      "senhora"    "dia"        "excelência" "ministério"
## [5,] "excelência" "gente"      "excelência" "vou"        "saúde"
```

Vamos visualizar com o topic modeling com o ggplot:

```r
# convertendo para o formato tidy
topicos <- tidytext::tidy(lda_model, matrix = "beta")

termos_p_topico <- 10
top_termos <- topicos %>%
  group_by(topic) %>%
  top_n(termos_p_topico, beta) %>%
  ungroup() %>%
  arrange(topic, -beta)
# top_n() doesn't handle ties -__- so just take top 10 manually
top_termos <- top_termos %>%
  group_by(topic) %>%
  slice(1:termos_p_topico) %>%
  ungroup()

top_termos$topic <- factor(top_termos$topic)

top_termos %>%
  mutate(term = reorder(term, beta)) %>%
  ggplot(aes(term, beta)) +
  geom_bar(stat = "identity") +
  facet_wrap(~ topic, scales = "free") +
  theme(axis.text.x = element_text(angle = 90, hjust = 1)) +
  coord_flip() +
  labs(title = "Modelagem tópicos - Topic Modeling",
       subtitle = paste0("k = ", valor_k, ", feito a partir das falas dos senadore
s"),
       caption = "Elaboração: Alisson Soares")
```



Elaboração: Alisson Soares

5.3.1 Análise de correspondência

Seguindo o tutorial de “Correspondence analysis” com o Quanteda (https://tutorials.quanteda.io/machine-learning/ca/).