Wordclouds en R

Por: Victor Manuel Sarmiento García y Nikoll Meneses Zambrano
Diseño por: Laura Barrera Valdés

Índice

00

Introducción

Las nubes de palabras (también conocidas como nubes de texto o nubes de etiquetas) funcionan de forma sencilla: cuanto más aparece una palabra en una fuente de datos de texto (como un discurso, una entrada de un blog o una base de datos), más grande y destacada se muestra en la nube de palabras.

> **NOTA:** si usted ya tiene una base de datos con cada palabra y su respectiva frecuencia de aparición, pase a la sección 3h. En caso contrario, continue a la sección 1 para limpiar el texto. En este caso será usado como ejemplo el discurso del Expresidente Colombiano Juan Manuel Santos en la firma del acuerdo de paz con la extinta guerrilla Farc-EP. Se recomienda acceder este link para descargar los archivos necesarios para seguir esta guia.

01

Limpieza

Inicialmente, se carga el texto.

```
text <- readLines(con = "Santos.txt", encoding = "UTF-8")
```

A continuación, se usa el paquete tm para crear un objeto Corpus que puede ser fácilmente manipulado con algunas técnicas de minería de textos.

```
library(tm)
```

```
text1 <- Corpus(x = VectorSource(x = text))
```

Luego, se convierten todas las palabras del discurso a minúsculas,

```
text1 <- tm_map(x = text1, FUN = tolower)
```

y se eliminan los espacios en blanco,

```
text1 <- tm_map(x = text1, FUN = stripWhitespace)
```

los signos de puntuación,

```
text1 <- tm_map(x = text1, FUN = removePunctuation)
```

los números,

```
text1 <- tm_map(x = text1, FUN = removeNumbers)
```

y las palabras sin significado como artículos, pronombres, preposiciones, etc.

```
text1 <- tm_map(text1, removeWords, stopwords("spanish"))
```

Hasta ahora ya se ha creado el objeto Corpus y está ajustado. Ahora se procesaran los datos de tal forma que se pueda crear una base de datos en donde se encuentre cada palabra junto a su respectiva frecuencia de aparición.

02

Procesamiento

Primero, se debe convertir el objeto Corpus a una matriz en la que sus filas representen las oraciones de los datos que deben analizarse , mientras que las columnas representen cada palabra existente en el texto.

```
text_matrix <- TermDocumentMatrix(x = text1)
```

Después, se transforma este en una matriz más sencilla.

```
text_matrix <- as.matrix(text_matrix)
```

Así, se suma la cantidad de veces que aparece cada palabra, se organizan de mayor a menor frecuencia de aparición,

```
text_data <- sort(rowSums(text_matrix), decreasing = TRUE)
```

y se crea un data frame. Una columna contiene las palabras, y la otra su respectiva frecuencia absoluta.

```
text_data1 <- data.frame(word = names(text_data),freq=text_data)
```

03 Visualización

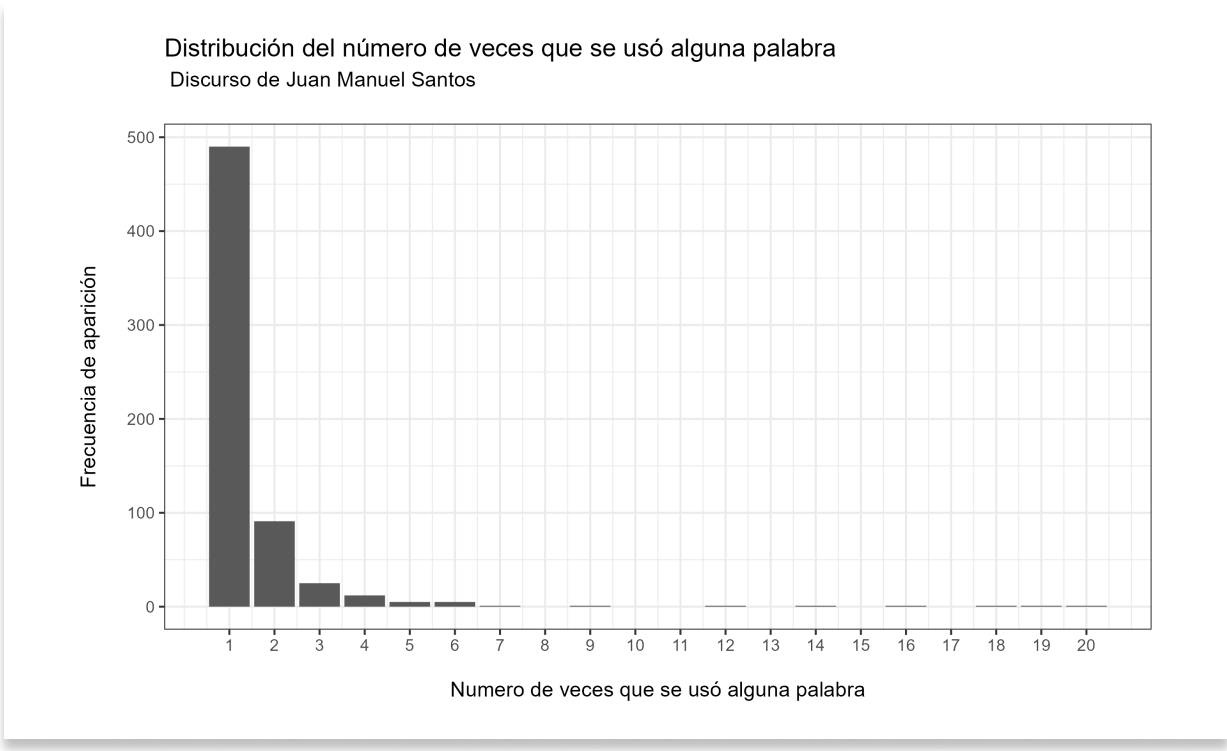
03.1. Graficos Tradicionales

La forma más sencilla de visualizar la frecuencia de aparición de estas palabras es usando graficos tradicionales con la libreria ggplot2.

El primer grafico, por ejemplo, muestra la frecuencia de aparición del numero de veces que aparece una palabra. Con esto es posible ver si hay unas cuantas palabras que se usan muchas veces, o bien, si hay muchas palabras que se usan pocas veces.

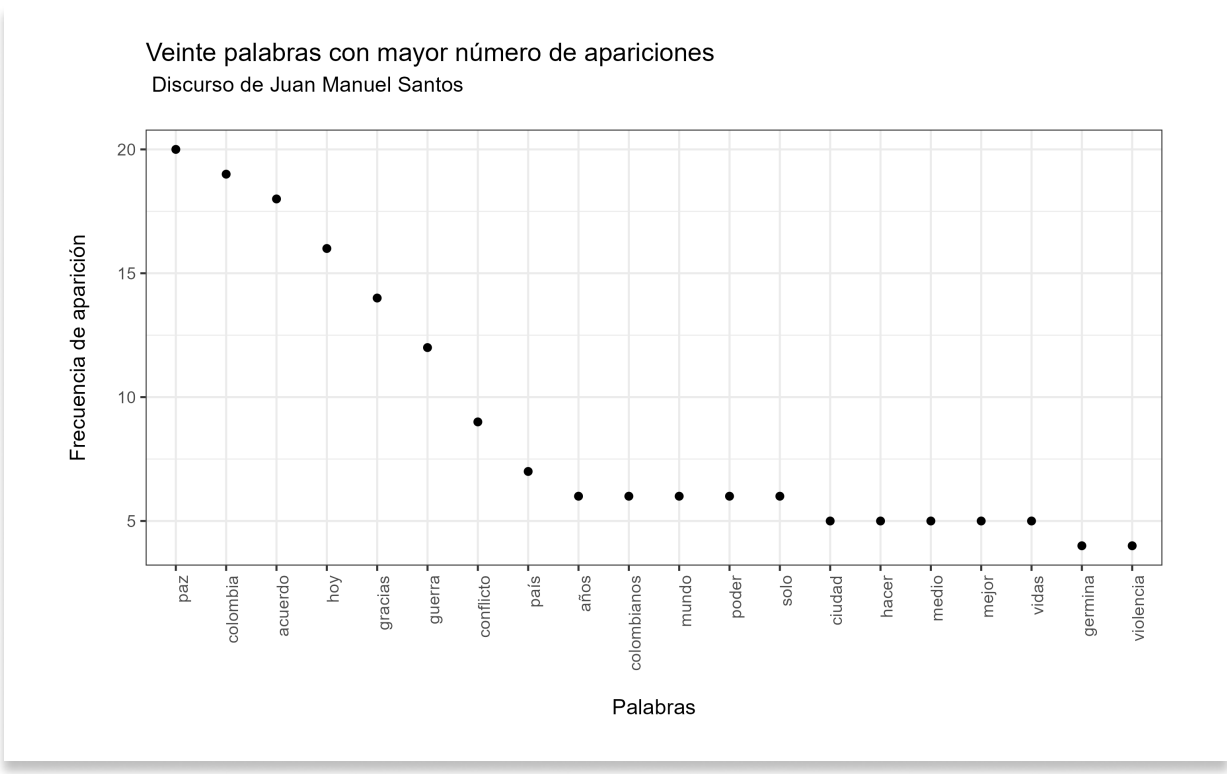
```
library(ggplot2)

ggplot(data = text_data, aes(x= freq)) + geom_bar()+ labs(title = "Distribución de la frecuencia de aparición de palabras",subtitle = " Discurso de Juan Manuel Santos\n", x=" \nNumero de palabras", y="Frecuencia de aparición\n")+ scale_x_continuous(labels = c(1:20), breaks = c(1:20))+ theme_bw()
```



Para saber cuales son esas palabras, y qué tanto aparecen en el discurso, se debe analizar el segundo grafico.

```
ggplot(data = text_data[1:20, ], aes(x= reorder(word, -freq), y = freq )) +  geom_point() + labs(title = "Veinte palabras con mayor número de apariciones",subtitle = " Discurso de Juan Manuel Santos\n", x=" \nPalabras",y="Frecuencia de aparición\n") + theme_bw()+  theme(axis.text.x = element_text(angle = 90, hjust = 1))
```



Sin embargo, ninguno de los dos graficos son muy intuitivos ni llamativos. Es justo por esta razón que se prefiere usar las nubes de palabras cuando se analizan textos, discursos, etc.

03.2. Wordclouds

La mejor forma para construir nubes de palabras con R es usando el paquete wordcloud2. Se debe tener en cuenta que el formato de salida predeterminado es HTML, aunque tambien es posible obtener los resutlados en formato PNG o PDF.

> **NOTA:**
> Para no tener inconvenientes con algunas funciones de la libreria se recomienda hacer su instalación desde el repositorio de GitHub. La instalación desde CRAN genera algunos errores al usar funciones como lettercloud() o al usar figuras personalizadas.

```
devtools::install_github("lchiffon/wordcloud2")
```

Sintaxis de la función

```
wordcloud2(data, size = 1, minSize = 0, gridSize = 0,  fontFamily='Segoe UI', fontWeight = 'bold',
          color = 'random-dark', backgroundColor="white", minRotation=-pi/4, maxRotation=pi/4,
          shuffle=T, rotateRatio=0.4, shape='circle', ellipticity=0.65,widgetsize=NULL,figPath=NULL,
          hoverFunction=NULL)
```

Argumentos de uso común

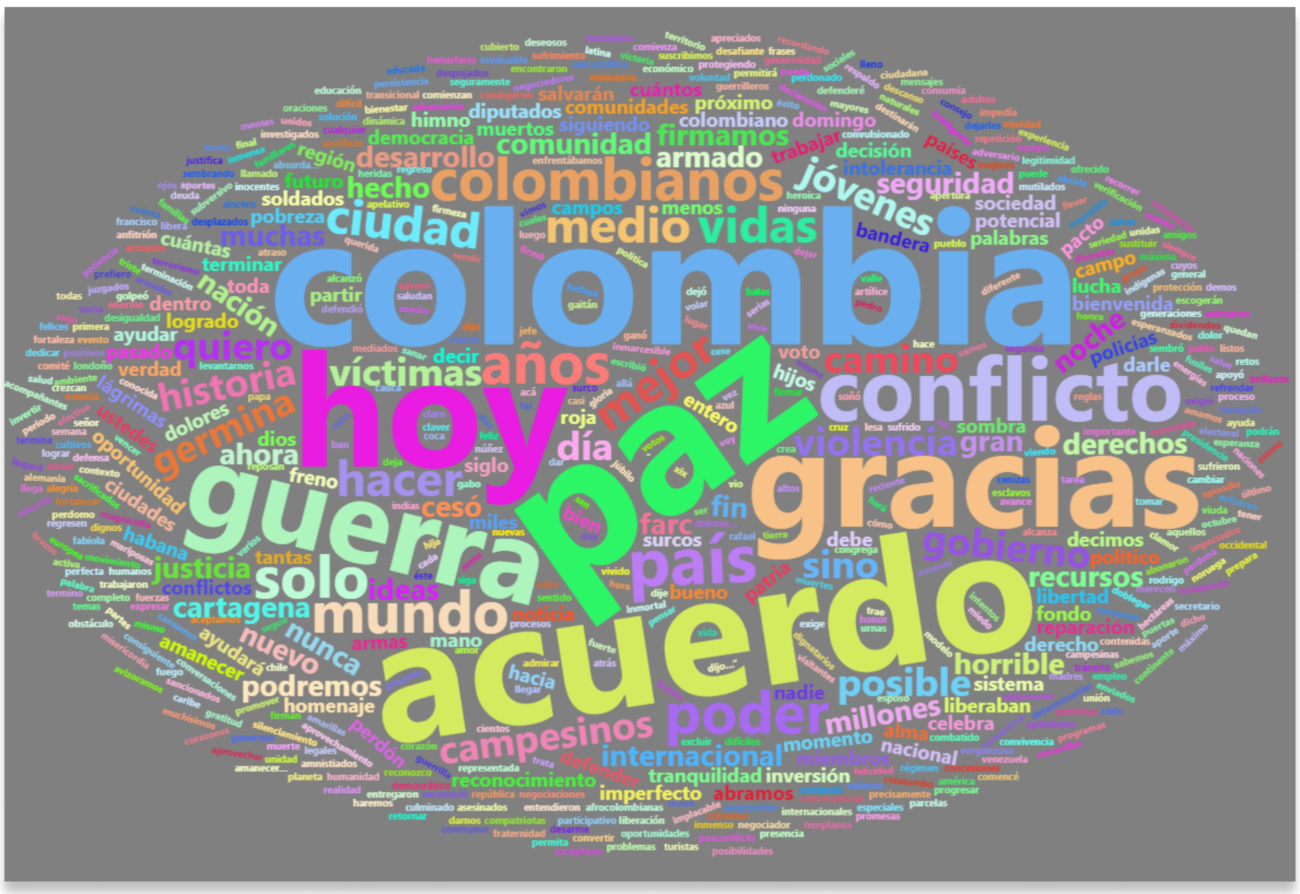
Data
Solicita un data frame en el que palabra y frecuencia está en cada columna.

```
wordcloud2(data = text_data)
```

Color
Ajusta el color del texto. Se pueden usar paletas como: "random-light", "random-dark", etc.

BackgroundColor
Ajusta el color de fondo. Pueden ser usados colores como "grey", "black", pero no otros más especificos como "grey20"

```
wordcloud2(data = text_data, color = "random-light", backgroundColor = "grey")
```



minRotation/maxRotation

Configura la rotación del texto minima o maxima (en radianes).

wordcloud2(data = text_data, minRotation = -pi/2, maxRotation = -pi/2)

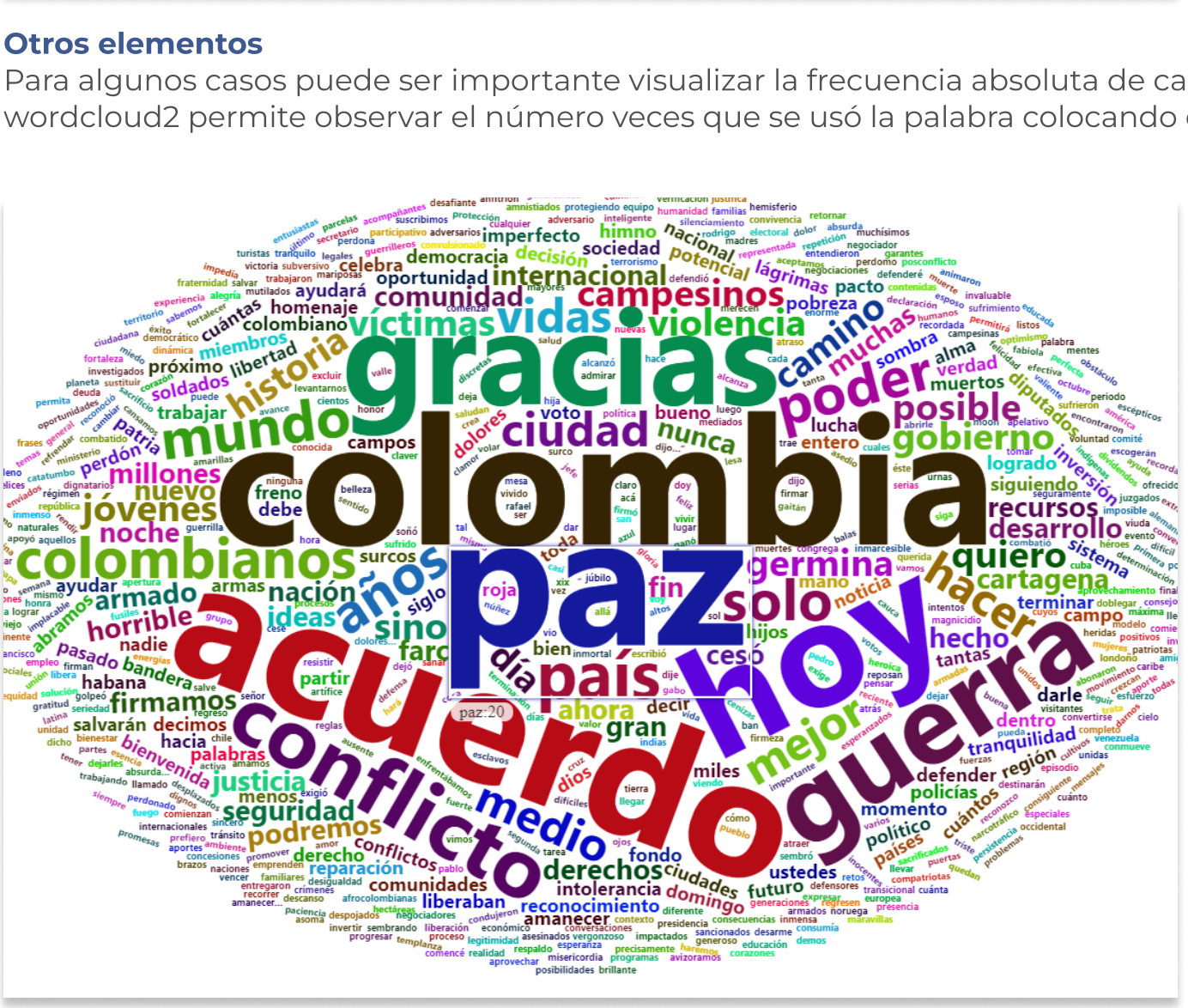
rotateRatio

Fija la probabilidad de que la palabra rote. Si toma el valor de 1, la palabra siempre rotará.

wordcloud2(data = text_data, minRotation = -pi/6, maxRotation = -pi/6, rotateRatio = 1)

Otros elementos

Para algunos casos puede ser importante visualizar la frecuencia absoluta de cada palabra. El paquete wordcloud2 permite observar el número veces que se usó la palabra colocando el mouse sobre esta.



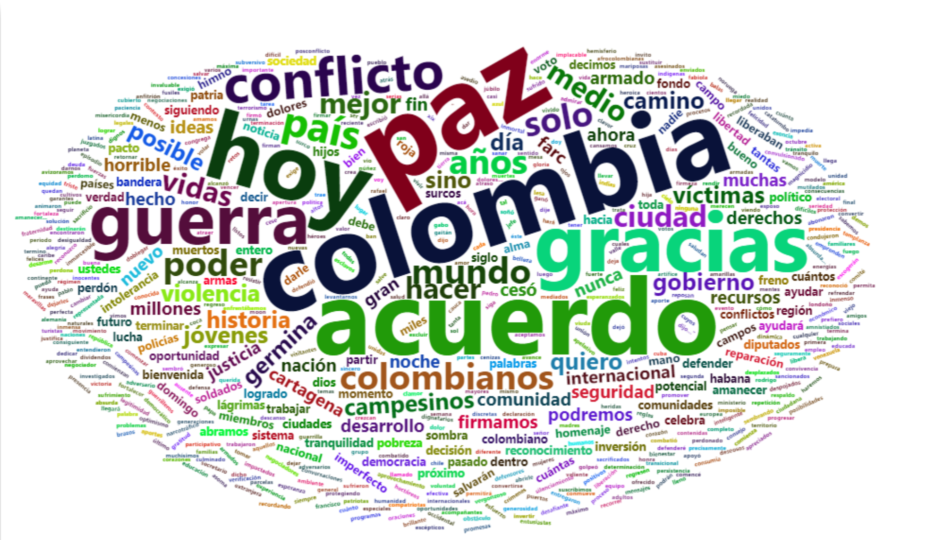
Formas predefinidas

Además de las modificaciones que se han visto, un wordcloud tambien puede ser presentado a traves de diferentes formas. Las formas disponibles en el paquete son las siguientes:

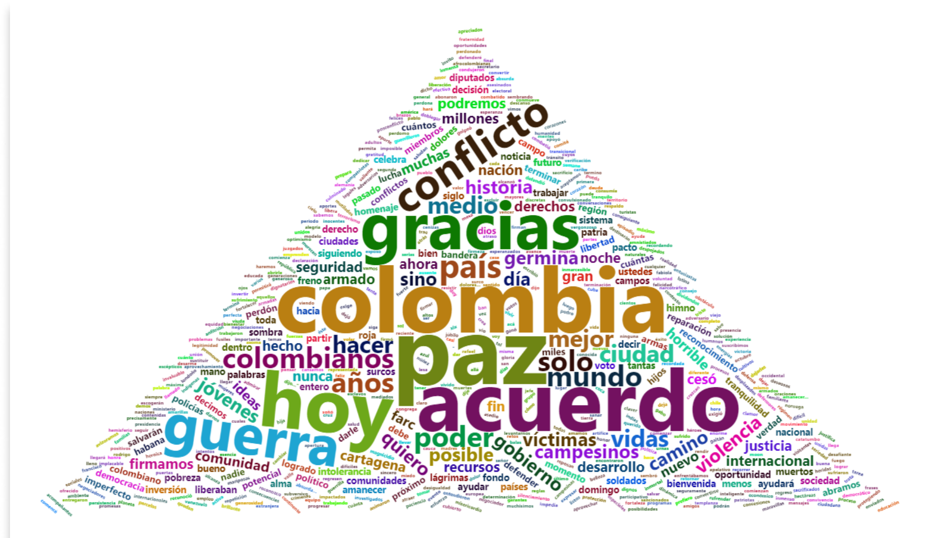
Circle

Triangle-forward

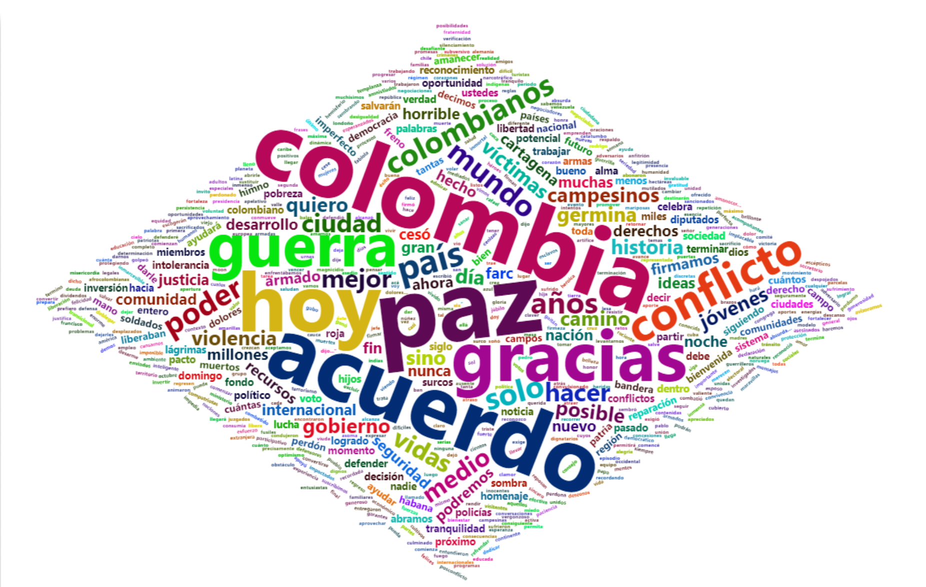
Cardioid



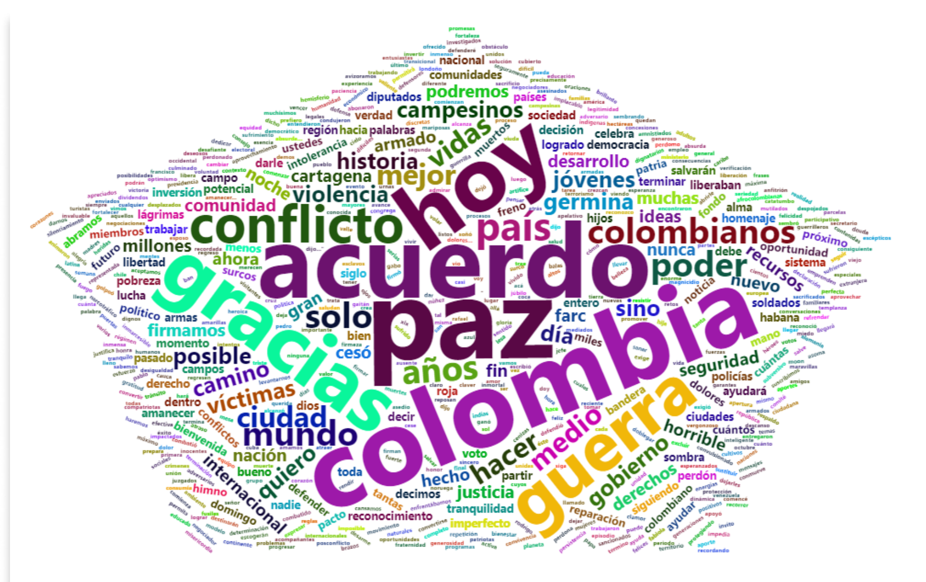
Triangle



Diamond

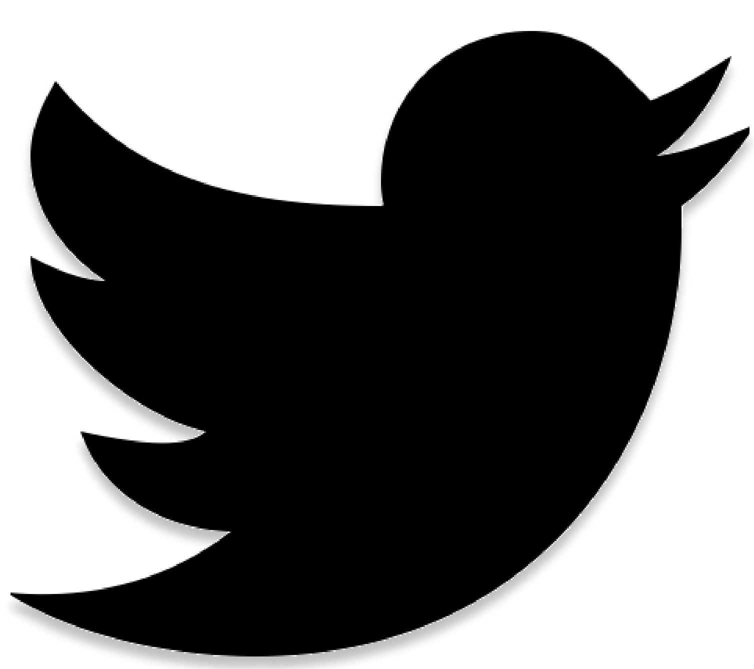


Pentagon



Con imágenes

El paquete wordcloud2 tambien permite usar imagenes personalizadas como formas para crear un wordcloud. Para ello, se recomienda elegir una imagen personalizada que coincida con el contexto del texto en cuestion. Concretamente, se debe descargar la imagen en blanco y negro con formato PNG, y ubicarla en el directorio de trabajo. Como ejemplo, se usará una imagen de un pajaro que representa el contexto del discurso analizado.



Una vez con la imagen, se procede a realizar el grafico. Para que sea más llamativo se usará una paleta de colores personalizada del paquete MetBrewer. Se recomienda jugar con los argumentos “size” y “gridsize” para obtener mejores resultados asi como refrescar continuamente el visualizador HTML de Rstudio.

```
library(MetBrewer)

colors <- met.brewer("Benedictus")

c_trim <- c(rep(colors[1], 50), rep(colors[length(colors)], nrow(text_data) - 50))

wordcloud2(text_data, figPath = "bird.png", size =6, color = c_trim, backgroundColor = "white",gridSize = 10)
```

Con palabras

De hecho, el paquete wordcloud2 tiene una función llamada lettercloud que permite crear un wordcloud usando la forma de una palabra.

Sintaxis de la función

letterCloud(data, word, wordSize=0, letterFont=NULL,...)

data

Solicita un data frame en el que la palabra y su respectiva frecuencia absoluta está en cada columna.

word

Requiere una palabra para crear una forma para el wordcloud.

wordSize

Ajusta el tamaño de la palabra.

letterFont

Ajusta la fuente del grafico.

...

Otros argumentos de la función wordcloud2

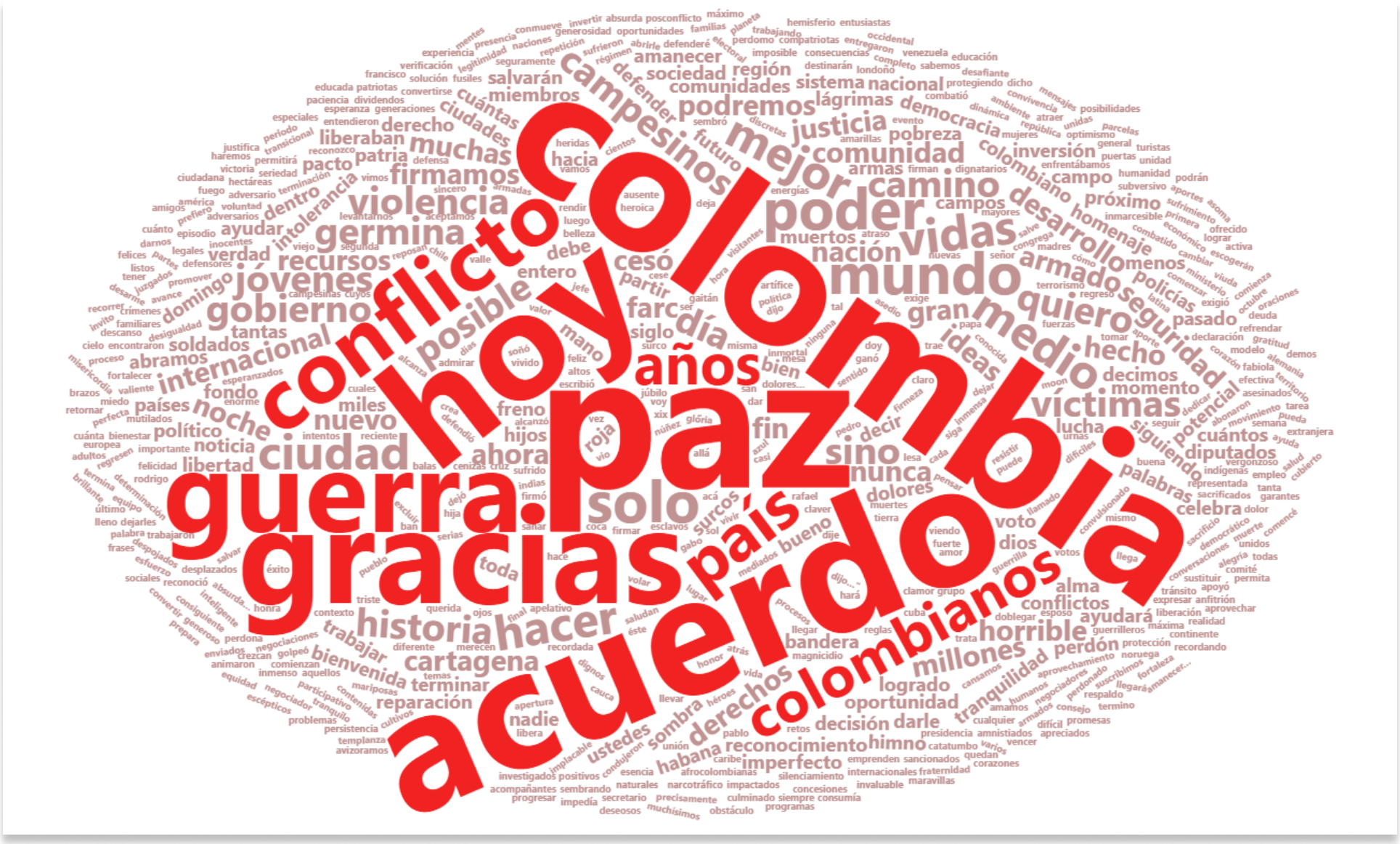
En este caso, se usará la palabra “PAZ” para representar el contexto del discurso.

```
letterCloud(text_data, word="PAZ", wordSize = 1, color="random-dark",  backgroundColor="white", size =5, gridSize = 10)
```

Más ejemplos

Es posible resaltar con un color en especifico las palabras que fueron mencionadas en el discurso más de veinte veces.

```
wordcloud2(text_data, color = ifelse(demoFreq[,2] > 20, '#f02222', '#c09292'))
```



También se pueden usar temas predefinidos para cambiar la apariencia del wordcloud. Es posible usarlos uno a la vez o combinados.

```
WC <- wordcloud2(text_data)

WC + WCtheme(1)+WCtheme(2) +WCtheme(3)
```

Otros paquetes

Finalmente, es posible explorar otro paquete llamado *wordcloud* para realizar wordclouds que puedan comparar las palabras usadas en dos o más discursos. En este caso, se usó tambien el discurso de Rodrigo Londoño Echeverri alias “Timochenko” en la firma del acuerdo de paz para ser comparado con el del expresidente Juan Manuel Santos.

NOTA:
A continuación se deja el codigo con el que se realizó la limpieza del texto del discurso de Timochenko y su respectiva unión en una sola base de datos con el texto del discurso de Santos.

```
text <- readLines(con = "Timochenko.txt", encoding = "UTF-8")

library(tm)

text1 <- Corpus(x = VectorSource(x = text))

text1 <- tm_map(x = text1, FUN = tolower)

text1 <- tm_map(x = text1, FUN = stripWhitespace)

text1 <- tm_map(x = text1, FUN = removePunctuation)

text1 <- tm_map(x = text1, FUN = removeNumbers)

text1 <- tm_map(text1, removeWords, stopwords("spanish"))

text_matrix <- TermDocumentMatrix(x = text1)

text_matrix <- as.matrix(text_matrix)

text_data <- sort(rowSums(text_matrix), decreasing = TRUE)

text_data <- data.frame(word = names(text_data),freq=text_data)

library(tidyr)

text_data2 <- text_data %>%
mutate(word = gsub("[[:punct:]]", "", word))%>%
group_by(word) %>%
summarise(freq=sum(freq))

names(text_data1)[2] <- "Santos"
names(text_data2)[2] <- "Timochenko"

data.completa <- full_join(text_data2,text_data1) %>%
drop_na()

data.completa <-data.completa[1:500,]

data.completa[is.na(data.completa)] <- 0

data.completa[-c(963,962,310,383,1052,1076,1077),]

x <- data.completa[,1] %>% pull()

data.completa <- as.matrix(data.completa[,-1])

row.names(data.completa) <- x
```

Luego de tener la base de datos que captura los dos discursos, Se pueden realizar graficos como los presentados a continuación. El primero compara las diferencias entre los discursos y el segundo las similitudes.

```
comparison.cloud(data.completa, random.order=FALSE,
colors = c("indianred3","lightsteelblue3"), title.size=1.5,
max.words=500,rot.per = .1,fixed.asp=TRUE)
```

```
commonality.cloud(data.completa, random.order=FALSE,
scale=c(5, .5),colors = brewer.pal(4, "Dark2"), max.words=400)
```