Análisis temático de textos con R

Etnografía y Análisis Cualitativo de Datos. Grado en Antropología Social y Cultural

Arturo Alvarez-Roldan
aalvarez@ugr.es
Departamento de Antropología Social, Universidad de Granada

15 / June / 2025

DOI: 10.5281/zenodo.15667595

Índice

Resumen: Este documento presenta una propuesta metodológica para realizar análisis temático de textos utilizando R, en el marco del análisis cualitativo y cuantitativo aplicado a la antropología. Se utiliza como ejemplo el texto *Raza e historia* de Claude Lévi-Strauss. La metodología se estructura en cuatro fases: transformación del texto en datos ordenados; codificación temática mediante métodos manuales, dinámicos y automáticos; análisis relacional entre códigos mediante matrices y visualizaciones (mapas de calor, MDS, conglomerados y redes); y finalmente la elaboración de un mapa conceptual que sintetiza las relaciones clave. El enfoque enfatiza la intervención del investigador y promueve la transparencia, replicabilidad y eficacia del análisis textual.

Abstract: This document presents a methodological proposal for thematic text analysis using R, within a qualitative and quantitative framework applied to anthropology. The example used is Claude Lévi-Strauss's *Race and History*. The method includes four stages: transforming the text into structured data; thematic coding using manual, dynamic, and automatic methods; relational analysis between codes through matrices and visualizations (heatmaps, MDS, clusters, and networks); and a final conceptual map summarizing

key relationships. The approach highlights the active role of the researcher and promotes transparency, reproducibility, and effectiveness in textual analysis.

1 Introducción

En este artículo proponemos una estrategia para realizar análisis temático de textos (Ryan y Bernard 2003) mediante el uso de **R**, un lenguaje de programación y entorno de software de código abierto, especializado en estadística, ciencia de datos y visualización. Esta propuesta se enmarca dentro de un enfoque metodológico mixto, ya que combina técnicas de análisis cualitativas y cuantitativas.

El procedimiento que sugerimos se compone de cuatro fases:

1. **Creación del proyecto y preparación de los datos**: explicamos cómo incorporar los textos a tablas que pueden gestionarse como bases de datos (Silge y Robinson 2017). La unidad de análisis puede ser una palabra, una oración, un párrafo o, de manera más general, una cita considerada como unidad de significado.
2. **Codificación**: se detallan los pasos para asignar códigos al leer el texto. Se propone explorar la frecuencia de palabras parar identificar códigos potenciales. Se sugiere una forma de llevar a cabo una codificación automática. Se describe la manera de crear y organizar los memorandos a partir de los textos codificados.
3. **Análisis relacional**: una vez codificado el texto, se explica cómo construir matrices de coocurrencia y similitud entre los códigos. Se muestra cómo estas matrices permiten el análisis estadístico de las relaciones entre códigos, generando mapas que representan la estructura semántica del texto (Guest, MacQueen, y Namey 2011).
4. **Mapeado conceptual**: se describe cómo dibujar mapas conceptuales que muestren la estructura y el significado de los textos analizados.

Este enfoque reconoce el carácter interpretativo del trabajo cualitativo: aunque se recurre al soporte computacional, la codificación se realiza con la intervención activa del investigador. Las visualizaciones generadas —gráficos, redes y mapas— no solo enriquecen el análisis, sino que también facilitan la presentación de los resultados. El script del proyecto generado, junto con los datos textuales utilizados, otorgan transparencia y fiabilidad al proceso, permitiendo su revisión, evaluación y reproducción por parte de otros investigadores.

2 Crear el proyecto e incorporar los datos

Para realizar el análisis utilizaremos como datos la versión española de un texto de Claude Levi-Strauss (1979), *Raza e historia*.

Abrimos un nuevo proyecto en R al que llamaremos `"Analisis de textos"`. Dentro del proyecto creamos seis carpetas para organizar sus elementos:

- `codigo`
- `datos`
- `tablas`
- `temas`
- `memorandos`
- `graficos`

Introducimos en la carpeta `datos` el archivo en formato `.txt` del texto que vamos a analizar: `"L-S_Raza-historia.txt"`.

2.1 Convertir el texto en datos ordenados en tablas (*Tidy data*)

Abrimos un nuevo script en el editor de texto y lo guardamos como `"codificacion.R"` en la carpeta `codigo`.

Instalamos y cargamos las bibliotecas que vamos a utilizar.

```r
install.packages(c("tidyverse", "tidytext", "writexl"))
```

```r
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.2     v tibble    3.2.1
## v lubridate 1.9.2     v tidyr     1.3.1
## v purrr     1.0.4
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become error
```

```r
library(tidytext)
library(writexl)
```

Leemos el texto.

```r
texto <- readLines("datos/L-S_Raza-historia.txt")
```

Creamos una tabla, que vamos a llamar `tib_texto`, con dos columnas: `parrafo` (número del párrafo) y `texto` (contenido del párrafo):

```r
tib_texto <- tibble(parrafo = seq_along(texto), texto = texto)
```

El resultado puede verse en la Tabla 1.

```r
library(kableExtra)
```

```
##
## Attaching package: 'kableExtra'
```

```
## The following object is masked from 'package:dplyr':
##
##     group_rows
```

```r
tib_texto_corto <- tib_texto %>%
  mutate(texto = str_trunc(texto, width = 70))

kable(head(tib_texto_corto),
      caption = "Primeras filas de la tabla \\texttt{tib\\_texto}") %>%
  kableExtra::kable_styling(
    latex_options = c("hold_position", "striped"),
    position = "center",
    font_size = 10
    )
```

Guardamos la tabla.

```r
save(tib_texto, file = "tablas/tib_texto.RData")
```

Tabla 1: Primeras filas de la tabla `tib_texto`

parrafo	texto
1	Hablar de la contribución de las razas humanas a la civilización m...
2	Pero nada más lejos de nuestro propósito que una empresa tal, que ...
3	Por eso, cuando hablamos en este estudio de la contribución de las...
4	Pero nos ha parecido que, en la medida en que esta serie de capítu...
5	Pero aquella diversidad se distingue por dos caracteres importante...
6	Al fin y al cabo, hay que preguntarse en qué consiste esta diversi...

3 Codificar el texto

3.1 Buscar códigos analizando la frecuencia de las palabras

El análisis de la frecuencia de palabras puede ayudarnos a descubrir conceptos o categorías importantes. A partir de `tib_texto`, creamos una nueva tabla donde cada palabra se convierte en una observación.

Lo primero que haremos será cargar un diccionario de palabras vacías, a saber, palabras que no tienen ningún significado relevante y queremos eliminar del texto.

```
vacias <- read_csv(
  "https://raw.githubusercontent.com/7PartidasDigital/AnaText/master/datos/diccionarios/vacias.txt",
  locale = default_locale())
```

```
## Rows: 465 Columns: 1
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (1): palabra
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

Procedemos a tokenizar en palabras el texto, eliminar las vacías y contar la frecuencia con que aparecen.

```
tib_texto_frec_palabras <- tib_texto %>%
  unnest_tokens(palabra, texto) %>%
  anti_join(vacias, by = "palabra") %>%
  count(palabra, sort = TRUE)
```

Guardamos esta tabla en un archivo de Excel para poder examinarla más fácilmente.

```
write_xlsx(tib_texto_frec_palabras, "tablas/tib_texto_frecuencias.xlsx")
```

Revisando el archivo de Excel podemos agrupar palabras con la misma raíz (por ejemplo, “cultura”, “culturas”, “cultural”) y ajustar el recuento manualmente.

El análisis de frecuencia revela una serie de conceptos clave como:

> **“cultura”, “civilización”, “humanidad”, “sociedad”, “historia”, “acumulativa”, “estacionaria”, “raza”, “biología”, “evolucionismo”, “occidental”, “revolución”, “primitivo”, “pueblos”, “indígena”, “colaboración”, “progreso”, “diversidad”**

Vamos a utilizar estos conceptos para codificar el texto.

Los códigos también se pueden crear leyendo el texto e identificando las ideas o conceptos más importantes.

3.2 Codificación del texto

Para codificar el texto podemos seguir tres procedimientos: **manual**, **dinámico** y **automático**.

3.2.1 Codificación manual

Primero, añadimos a la tabla original (`tib_texto`) nuevas columnas vacías para los distintos códigos temáticos, así como una última columna llamada `codigo` en la que agruparemos todos los códigos aplicables a cada párrafo.

```
tib_texto_codigos <- tib_texto %>%
  mutate(
    cultura = NA_character_,
    civilizacion = NA_character_,
    humanidad = NA_character_,
    sociedad = NA_character_,
    historia = NA_character_,
    acumulativa = NA_character_,
    estacionaria = NA_character_,
    raza = NA_character_,
    biologia = NA_character_,
    evolucionismo = NA_character_,
    occidental = NA_character_,
    revolucion = NA_character_,
    primitivo = NA_character_,
    pueblos = NA_character_,
    indigena = NA_character_,
    colaboracion = NA_character_,
    progreso = NA_character_,
    diversidad = NA_character_,
    codigo = NA_character_
  )
```

A continuación, leeremos el texto de cada párrafo para identificar qué temas o códigos aparecen en él. Podemos hacerlo en el editor de texto buscando la raíz de los distintos términos y anotando los números de párrafo en los que aparece cada concepto. La lectura detallada de estos párrafos nos permitirá confirmar si realmente contienen el tema en cuestión.

Una vez tengamos la lista con los números de párrafos en los que aparece cada código, trasladaremos esa información a la tabla de la siguiente manera:

```
tib_texto_codigos$cultura[c(2:3, 5:13, 15:19, 21:25, 28:30, 32, 34:39, 43:46, 52, 58,
                            65:66, 68, 71, 73:78, 80:81, 84:87, 93, 95,
                            98:100)] <- "cultura"
tib_texto_codigos$civilizacion[c(1, 3:4, 6:9, 13:14, 22:23, 27:28, 31, 33:35, 37,
                                 40:41, 43:47, 49:51, 59:61, 67, 75:77, 80:82, 84,
                                 86)] <- "civilizacion"
tib_texto_codigos$humanidad[c(1:10, 13:20, 25, 28:29, 31, 33, 39:43, 45:46, 49, 51:52,
                              58, 60, 62:63, 70, 73, 75, 79, 80, 82, 93, 95:98,
                              100)] <- "humanidad"
tib_texto_codigos$sociedad[c(4, 7, 8:11, 18:21, 23:25, 28:30, 36, 40, 42, 47, 51, 57,
                             59, 75, 78, 82:84, 88, 95)] <- "sociedad"
tib_texto_codigos$historia[c(2, 28:30, 33:36, 45:46, 58, 62, 66, 73, 75,
                             78, 84, 99)] <- "historia"
tib_texto_codigos$acumulativa[c(29, 33:36, 61:63, 66, 68, 73:74, 78)] <- "acumulativa"
tib_texto_codigos$estacionaria[c(35:36, 38:39, 66, 70, 75, 78)] <- "estacionaria"
```

```
tib_texto_codigos$raza[c(1:3, 5:6, 14, 16, 19, 32, 64:65, 78, 80, 95)] <- "raza"
tib_texto_codigos$biologia[c(2, 4, 6, 19:20, 93)] <- "biologia"
tib_texto_codigos$evolucionismo[c(18:20, 23, 25, 30, 70)] <- "evolucionismo"
tib_texto_codigos$occidental[c(13, 23, 28, 34, 40:41, 43, 45:49, 57, 60:61,
                               63:65)] <- "occidental"
tib_texto_codigos$revolucion[c(51, 60:61, 63:65, 89)] <- "revolucion"
tib_texto_codigos$primitivo[c(2, 7, 13:15, 25:26, 40, 51, 57)] <- "primitivo"
tib_texto_codigos$pueblos [c(6, 14, 23:25, 28, 41, 45, 47, 51, 66, 91:92)] <- "pueblos"
tib_texto_codigos$indigena [c(14, 23:24, 26, 34, 47, 54)] <- "indigena"
tib_texto_codigos$colaboracion [c(72:74, 77, 86:88, 91:92, 94, 96)] <- "colaboracion"
tib_texto_codigos$progreso [c(6, 11, 30:31, 33, 51, 60, 71, 87, 90, 94, 97)] <- "progreso"
tib_texto_codigos$diversidad [c(4:7, 9:11, 13, 16:18, 29:30, 38, 76:77, 86:87, 89, 92:94,
                                96:97, 99:100)] <- "diversidad"
```

Unimos todas las columnas de códigos en una sola:

```
tib_texto_codigos <- tib_texto_codigos %>%
  unite("codigo", cultura:diversidad, sep = " ", na.rm = TRUE)
```

El resultado puede verse en la Tabla 2.

```
tib_texto_codigos_corto <- tib_texto_codigos %>%
  mutate(texto = str_trunc(texto, width = 30))

knitr::kable(head(tib_texto_codigos_corto[, c("parrafo", "texto", "codigo")]),
             caption = "Primeras filas de la tabla \\texttt{tib\\_texto\\_codigos}") %>%
  kable_styling(
    font_size = 10,
    latex_options = c("hold_position", "striped"),
    position = "center",
    full_width = FALSE
  )
```

Tabla 2: Primeras filas de la tabla `tib_texto_codigos`

parrafo	texto	codigo
1	Hablar de la contribución ...	civilizacion humanidad raza
2	Pero nada más lejos de nue...	cultura humanidad historia raza biologia primitivo
3	Por eso, cuando hablamos e...	cultura civilizacion humanidad raza
4	Pero nos ha parecido que, ...	civilizacion humanidad sociedad biologia diversidad
5	Pero aquella diversidad se...	cultura humanidad raza diversidad
6	Al fin y al cabo, hay que ...	cultura civilizacion humanidad raza biologia pueblos progreso diversidad

Guardamos la tabla.

```
save(tib_texto_codigos, file = "tablas/tib_texto_codigos.RData")
```

3.2.2 Codificación dinámica

Podemos simplificar el proceso utilizando listas y programación:

```
codigos_lista <- list(
  cultura = c(2:3, 5:13, 15:19, 21:25, 28:30, 32, 34:39, 43:46, 52, 58, 65:66,
              68, 71, 73:78, 80:81, 84:87, 93, 95, 98:100),
  civilizacion = c(1, 3:4, 6:9, 13:14, 22:23, 27:28, 31, 33:35, 37, 40:41,
```

```
                  43:47, 49:51, 59:61, 67, 75:77, 80:82, 84, 86),
  humanidad = c(1:10, 13:20, 25, 28:29, 31, 33, 39:43, 45:46, 49, 51:52, 58,
                60, 62:63, 70, 73, 75, 79, 80, 82, 93, 95:98, 100),
  sociedad = c(4, 7, 8:11, 18:21, 23:25, 28:30, 36, 40, 42, 47, 51, 57, 59,
               75, 78, 82:84, 88, 95),
  historia = c(2, 28:30, 33:36, 45:46, 58, 62, 66, 73, 75, 78, 84, 99),
  acumulativa = c(29, 33:36, 61:63, 66, 68, 73:74, 78),
  estacionaria = c(35:36, 38:39, 66, 70, 75, 78),
  raza = c(1:3, 5:6, 14, 16, 19, 32, 64:65, 78, 80, 95),
  biologia = c(2, 4, 6, 19:20, 93),
  evolucionismo = c(18:20, 23, 25, 30, 70),
  occidental = c(13, 23, 28, 34, 40:41, 43, 45:49, 57, 60:61, 63:65),
  revolucion = c(51, 60:61, 63:65, 89),
  primitivo = c(2, 7, 13:15, 25:26, 40, 51, 57),
  pueblos = c(6, 14, 23:25, 28, 41, 45, 47, 51, 66, 91:92),
  indigena = c(14, 23:24, 26, 34, 47, 54),
  colaboracion = c(72, 74, 77, 86:88, 91:92, 94, 96),
  progreso = c(6, 11, 30:31, 33, 51, 60, 71, 87, 90, 94, 97),
  diversidad = c(4:7, 9:11, 13, 16:18, 29:30, 38, 76:77, 86:87, 89, 92:94,
                 96:97, 99:100)
)
```

```
codigos <- names(codigos_lista)
tib_texto_codigos <- tib_texto %>%
  mutate(!!!setNames(rep(list(NA_character_), length(codigos)), codigos))
```

```
for (codigo in codigos) {
  indices <- codigos_lista[[codigo]]
  tib_texto_codigos[[codigo]][indices] <- codigo
}
```

```
tib_texto_codigos <- tib_texto_codigos %>%
  unite("codigo", all_of(codigos), sep = " ", na.rm = TRUE)
```

3.2.3 Codificación automática

Hasta ahora, la codificación que hemos propuesto requiere leer los párrafos del texto e identificar manualmente los términos clave. Sin embargo, también es posible realizar una **codificación automática**, localizando dichos términos mediante expresiones regulares.

Este procedimiento busca patrones específicos (palabras completas, raíces o sinónimos) en el texto. Si detecta alguno de estos patrones en un párrafo, asigna el código correspondiente; de lo contrario, asigna un valor perdido (NA).

Es importante tener en cuenta que esta estrategia no considera fenómenos como la polisemia (múltiples significados de una palabra), variaciones morfológicas o el uso de sinónimos. Por ello, los resultados deben interpretarse con cautela y, preferentemente, complementarse con una lectura cualitativa.

A continuación, presentamos el código para aplicar esta codificación automática al texto de Lévi-Strauss:

```
tib_texto_codigos_automatico <- tib_texto %>%
  mutate(
    cultura = if_else(str_detect(texto, regex("\\bcultura", ignore_case = TRUE)),
                      "cultura", NA_character_),
    civilizacion = if_else(str_detect(texto, regex("\\bcivilizac", ignore_case = TRUE)),
                           "civilizacion", NA_character_),
```

```
    humanidad = if_else(str_detect(texto, regex("\\bhuman", ignore_case = TRUE)),
                        "humanidad", NA_character_),
    sociedad = if_else(str_detect(texto, regex("\\bsociedad", ignore_case = TRUE)),
                       "sociedad", NA_character_),
    historia = if_else(str_detect(texto, regex("\\bhistoria\\b", ignore_case = TRUE)),
                       "historia", NA_character_),
    acumulativa = if_else(str_detect(texto, regex("\\bacumula", ignore_case = TRUE)),
                          "acumulativa", NA_character_),
    estacionaria = if_else(str_detect(texto,
                                      regex("\\bestacionaria", ignore_case = TRUE)),
                           "estacionaria", NA_character_),
    raza = if_else(str_detect(texto, regex("\\braza", ignore_case = TRUE)),
                   "raza", NA_character_),
    biologia = if_else(str_detect(texto, regex("\\bbiol", ignore_case = TRUE)),
                       "biologia", NA_character_),
    evolucionismo = if_else(str_detect(texto, regex("\\bevoluci", ignore_case = TRUE)),
                            "evolucionismo", NA_character_),
    occidental = if_else(str_detect(texto, regex("\\boccident", ignore_case = TRUE)),
                         "occidental", NA_character_),
    revolucion = if_else(str_detect(texto, regex("\\brevoluci[oó]n", ignore_case = TRUE)),
                         "revolucion", NA_character_),
    primitivo = if_else(str_detect(texto,
                                   regex("\\b(primitiv|salvaj)", ignore_case = TRUE)),
                        "primitivo", NA_character_),
    pueblos = if_else(str_detect(texto, regex("\\b(pueblo|tribu)", ignore_case = TRUE)),
                      "pueblos", NA_character_),
    indigena = if_else(str_detect(texto, regex("\\bind[ií]g", ignore_case = TRUE)),
                       "indigena", NA_character_),
    colaboracion = if_else(str_detect(texto,
                                      regex("\\b(coali|colabo)", ignore_case = TRUE)),
                           "colaboracion", NA_character_),
    progreso = if_else(str_detect(texto, regex("\\bprogres", ignore_case = TRUE)),
                       "progreso", NA_character_),
    diversidad = if_else(str_detect(texto, regex("\\bdiversidad", ignore_case = TRUE)),
                         "diversidad", NA_character_),
    codigo = NA_character_
  )
```

Una vez detectados los códigos individuales, los agrupamos en una única columna:

```
tib_texto_codigos_automatico <- tib_texto_codigos_automatico %>%
  unite("codigo", cultura:diversidad, sep = " ", na.rm = TRUE)
```

Guardamos la tabla.

```
save(tib_texto_codigos_automatico, file = "tablas/tib_texto_codigos_automatico.RData")
```

3.3 Buscar los textos de los códigos y redactar los memorandos

Para recuperar los textos correspondientes a distintos códigos (o combinaciones de ellos) y poder leerlos o archivarlos en formato `.txt`, procederemos del siguiente modo.

Supongamos que queremos extraer todos los párrafos codificados con el término evolucionismo. Para ello, filtraremos las filas de la columna `codigo` que contienen esa palabra:

```
textos_evolucionismo <- tib_texto_codigos %>%
  filter(str_detect(codigo, "evolucionismo"))
```

Podemos imprimir estos párrafos en la consola:

```
print(textos_evolucionismo$texto)
```

[1] " No obstante, por muy diferentes y a veces extrañas que puedan ser, todas estas especulaciones se…"

[2] " Esta definición puede parecer sumaria cuando recordamos las inmensas conquistas del…"

[3] " Además, la diferencia, olvidada con demasiada frecuencia, entre el verdadero y el falso evolucionismo…"

[4] " Sin embargo, es enormemente tentador querer establecer entre las culturas del primer grupo, relaciones…"

[5] " Una de las interpretaciones más populares que inspira el evolucionismo cultural trata de las pinturas…"

[6] " Primero debemos considerar las culturas que pertenecen al primero de los grupos que hemos…"

[7] " La humanidad no evoluciona en sentido único. Y si en un plano determinado parece estacionaria…"

O bien guardarlos como archivo de texto en la carpeta `temas`:

```
writeLines(textos_evolucionismo$texto, "temas/evolucionismo.txt")
```

También es posible buscar combinaciones de códigos mediante operadores lógicos:

- `&` para la conjunción (y),
- `|` para la disyunción (o),
- `!` para la negación (no).

Por ejemplo, si queremos extraer los párrafos donde aparecen simultáneamente los códigos biología, evolucionismo y raza, utilizaremos:

```
textos_evolucionismo_biologia_raza <- tib_texto_codigos %>%
  filter(
    str_detect(codigo, "biologia") &
    str_detect(codigo, "evolucionismo") &
    str_detect(codigo, "raza")
  )
```

```
print(textos_evolucionismo_biologia_raza$texto)

writeLines(textos_evolucionismo_biologia_raza$texto,
           "temas/evolucionismo_biologia_raza.txt")
```

El resultado es un fragmento en el que Lévi-Strauss explica que solo tiene sentido hablar de razas desde la teoría de la evolución biológica, y no desde el evolucionismo social defendido por antropólogos como Morgan:

> *"Esta definición puede parecer sumaria cuando recordamos las inmensas conquistas del darwinismo. Pero esta no es la cuestión, porque el evolucionismo biológico y el pseudo-evolucionismo que aquí hemos visto son dos doctrinas muy diferentes. La primera nace como una vasta hipótesis de trabajo, fundada en observaciones, cuya parte dejada a la interpretación es muy pequeña. (…) La noción de evolución biológica corresponde a una hipótesis dotada de uno de los más altos coeficientes de probabilidad que pueden encontrarse en el ámbito de las ciencias naturales, mientras que la noción de evolución social o cultural no aporta, más que a lo sumo, un procedimiento seductor aunque peligrosamente cómodo de presentación de los hechos."*

Para explorar las combinaciones únicas de códigos presentes en la tabla, podemos utilizar:

```
head(unique(tib_texto_codigos$codigo))
```

```
## [1] "civilizacion humanidad raza"
## [2] "cultura humanidad historia raza biologia primitivo"
## [3] "cultura civilizacion humanidad raza"
## [4] "civilizacion humanidad sociedad biologia diversidad"
## [5] "cultura humanidad raza diversidad"
## [6] "cultura civilizacion humanidad raza biologia pueblos progreso diversidad"
```

A medida que revisemos el contenido asociado a cada código o tema, redactaremos **memorandos analíticos**. Lo más práctico es partir de los archivos de texto generados para cada código e ir transformándolos en notas analíticas que incluyan citas *verbatim* seleccionadas.

4 Examinar las relaciones entre los códigos y la estructura semántica del texto

Uno de los aspectos más importantes del análisis temático consiste en estudiar las relaciones entre los temas para construir redes semánticas o modelos teóricos, como en la teoría fundamentada. Para ello, debemos crear una matriz de coocurrencias o de similitud entre los códigos.

Lo primero que haremos es cargar el archivo `tib_texto_codigos.RData` y transformar la tabla contenida en uno nuevo llamado `tib_texto_codigos2`, en el que tokenizaremos los términos incluidos en la columna codigo.

```
tib_texto_codigos2 <- tib_texto_codigos %>%
  unnest_tokens(codigo, codigo)
```

Para ver el número de párrafos que contiene cada código, usamos:

```
frecuencias <- table(tib_texto_codigos2$codigo)
print(frecuencias)
```

```
##
##   acumulativa      biologia  civilizacion  colaboracion       cultura
##            13             6            40            10            58
##    diversidad  estacionaria evolucionismo      historia     humanidad
##            26             8             7            18            49
##      indigena    occidental     primitivo      progreso       pueblos
##             7            18            10            12            13
##          raza    revolucion      sociedad
##            14             7            30
```

Guardamos la nueva tabla en la carpeta `tablas`:

```
save(tib_texto_codigos2, file = "tablas/tib_texto_codigos2.RData")
```

A continuación, presentamos las funciones necesarias para construir la matriz de coocurrencias y la matriz de proximidad a partir de `tib_texto_codigos2`.

4.1 Crear la matriz de coocurrencias

Para construir la matriz de coocurrencias:

1. Creamos una matriz vacía.
2. Contamos las coocurrencias de códigos por párrafo.
3. Rellenamos la matriz con esos valores.

```
crear_matriz_coocurrencias <- function(data_frame) {
  codigos_unicos <- unique(data_frame$codigo)
  matriz_coocurrencias <- matrix(0,
```

```
                                  nrow = length(codigos_unicos),
                                  ncol = length(codigos_unicos),
                                  dimnames = list(codigos_unicos, codigos_unicos))

  for (parrafo in unique(data_frame$parrafo)) {
    codigos_parrafo <- data_frame$codigo[data_frame$parrafo == parrafo]
    coocurrencias_parrafo <- table(codigos_parrafo)

    for (codigo1 in names(coocurrencias_parrafo)) {
      for (codigo2 in names(coocurrencias_parrafo)) {
        if (codigo1 != codigo2) {
          matriz_coocurrencias[codigo1, codigo2] <-
            matriz_coocurrencias[codigo1, codigo2] + coocurrencias_parrafo[codigo1]
        }
      }
    }
  }
  return(matriz_coocurrencias)
}
```

Aplicamos la función al dataframe:

```
matriz_coocurrencias <- tib_texto_codigos2 %>%
  crear_matriz_coocurrencias()
```

El resultado puede verse en la Tabla 3.

```
  knitr::kable(matriz_coocurrencias [, 1:10],
               format = "latex",
               booktabs = TRUE,
               caption = "Matriz de coocurrencias (truncada)") %>%
    kableExtra::kable_styling(latex_options = c("striped", "hold_position"),
                              font_size = 8)
```

Tabla 3: Matriz de coocurrencias (truncada)

	civilizacion	humanidad	raza	cultura	historia	biologia	primitivo	sociedad	diversidad	pueblos
civilizacion	0	23	5	23	8	2	5	13	8	8
humanidad	23	0	10	30	10	6	8	17	15	7
raza	5	10	0	11	2	3	2	3	3	2
cultura	23	30	11	0	16	4	5	19	20	7
historia	8	10	2	16	0	1	1	7	3	3
biologia	2	6	3	4	1	0	1	3	3	1
primitivo	5	8	2	5	1	1	0	5	2	3
sociedad	13	17	3	19	7	3	5	0	8	6
diversidad	8	15	3	20	3	3	2	8	0	2
pueblos	8	7	2	7	3	1	3	6	2	0
progreso	5	6	1	5	2	1	1	3	6	2
occidental	13	10	2	8	4	0	3	5	1	5
indigena	4	1	1	3	1	0	2	3	0	4
evolucionismo	1	5	1	5	1	2	1	6	2	2
acumulativa	4	5	1	9	9	0	0	3	1	1
estacionaria	2	3	1	7	5	0	0	3	1	1
revolucion	3	3	2	1	0	0	1	1	1	1
colaboracion	2	1	0	4	0	0	0	1	6	2

4.2 Crear la matriz de proximidades

En lugar de utilizar la coocurrencia como criterio de similitud, vamos a aplicar un coeficiente de proximidad similar al utilizado en el software de análisis cualitativo Atlas.ti: **C = n12 / (n1 + n2 - n12)** donde `n12` representa el número de coocurrencias, `n1` es la frecuencia del código 1 y `n2` la frecuencia del código 2. Este coeficiente tiene en cuenta, además de la coocurrencia, la frecuencia con la que los códigos se aplican en todo el texto.

El procedimiento para crear la matriz de proximidades es similar al que utilizamos para la matriz de coocurrencias. Primero creamos una matriz vacía, luego calculamos las frecuencias de ocurrencia de cada código, y finalmente aplicamos la función al data frame tib_texto_codigos2.

```
crear_matriz_proximidades <- function(data_frame) {
  codigos_unicos <- unique(data_frame$codigo)
  n_codigos <- length(codigos_unicos)
  matriz_proximidades <- matrix(0, nrow = n_codigos, ncol = n_codigos,
                                dimnames = list(codigos_unicos, codigos_unicos))
  frecuencias <- table(data_frame$codigo)

  for (i in 1:(n_codigos - 1)) {
    codigo1 <- codigos_unicos[i]
    for (j in (i + 1):n_codigos) {
      codigo2 <- codigos_unicos[j]
      parrafos_codigo1 <- data_frame$parrafo[data_frame$codigo == codigo1]
      parrafos_codigo2 <- data_frame$parrafo[data_frame$codigo == codigo2]
      n12 <- length(intersect(parrafos_codigo1, parrafos_codigo2))
      n1 <- frecuencias[codigo1]
      n2 <- frecuencias[codigo2]
      if ((n1 + n2 - n12) != 0) {
        proximidad <- n12 / (n1 + n2 - n12)
        matriz_proximidades[codigo1, codigo2] <- proximidad
        matriz_proximidades[codigo2, codigo1] <- proximidad
      }
    }
  }
  return(matriz_proximidades)
}
```

```
matriz_proximidades <- tib_texto_codigos2 %>% crear_matriz_proximidades()
```

```
class(matriz_proximidades)
```

```
## [1] "matrix" "array"
```

El resultado puede verse en la Tabla 4.

```
knitr::kable(matriz_proximidades [, 1:9],
             format = "latex", booktabs = TRUE,
             caption = "Matriz de proximidades (truncada)") %>%
    kableExtra::kable_styling(latex_options = c("striped", "hold_position"),
                              font_size = 8)
```

Convertimos las matrices a tablas y las guardamos en la carpeta `tablas`.

```
matriz_coocurrencias <- as_tibble(matriz_coocurrencias)
matriz_proximidades <- as_tibble(matriz_proximidades)

save(matriz_coocurrencias, file = "tablas/matriz_coocurrencias.RDta")
```

Tabla 4: Matriz de proximidades (truncada)

	civilizacion	humanidad	raza	cultura	historia	biologia	primitivo	sociedad	diversidad
civilizacion	0.0000000	0.3484848	0.1020408	0.3066667	0.1600000	0.0454545	0.1111111	0.2280702	0.1379310
humanidad	0.3484848	0.0000000	0.1886792	0.3896104	0.1754386	0.1224490	0.1568627	0.2741935	0.2500000
raza	0.1020408	0.1886792	0.0000000	0.1803279	0.0666667	0.1764706	0.0909091	0.0731707	0.0810811
cultura	0.3066667	0.3896104	0.1803279	0.0000000	0.2666667	0.0666667	0.0793651	0.2753623	0.3125000
historia	0.1600000	0.1754386	0.0666667	0.2666667	0.0000000	0.0434783	0.0370370	0.1707317	0.0731707
biologia	0.0454545	0.1224490	0.1764706	0.0666667	0.0434783	0.0000000	0.0666667	0.0909091	0.1034483
primitivo	0.1111111	0.1568627	0.0909091	0.0793651	0.0370370	0.0666667	0.0000000	0.1428571	0.0588235
sociedad	0.2280702	0.2741935	0.0731707	0.2753623	0.1707317	0.0909091	0.1428571	0.0000000	0.1666667
diversidad	0.1379310	0.2500000	0.0810811	0.3125000	0.0731707	0.1034483	0.0588235	0.1666667	0.0000000
pueblos	0.1777778	0.1272727	0.0800000	0.1093750	0.1071429	0.0555556	0.1500000	0.1621622	0.0540541
progreso	0.1063830	0.1090909	0.0400000	0.0769231	0.0714286	0.0588235	0.0476190	0.0769231	0.1875000
occidental	0.2888889	0.1754386	0.0666667	0.1176471	0.1250000	0.0000000	0.1200000	0.1162791	0.0232558
indigena	0.0930233	0.0181818	0.0500000	0.0483871	0.0416667	0.0000000	0.1333333	0.0882353	0.0000000
evolucionismo	0.0217391	0.0980392	0.0500000	0.0833333	0.0416667	0.1818182	0.0625000	0.1935484	0.0645161
acumulativa	0.0816327	0.0877193	0.0384615	0.1451613	0.4090909	0.0000000	0.0000000	0.0750000	0.0263158
estacionaria	0.0434783	0.0555556	0.0476190	0.1186441	0.2380952	0.0000000	0.0000000	0.0857143	0.0303030
revolucion	0.0681818	0.0566038	0.1052632	0.0156250	0.0000000	0.0000000	0.0625000	0.0277778	0.0312500
colaboracion	0.0416667	0.0172414	0.0000000	0.0625000	0.0000000	0.0000000	0.0000000	0.0256410	0.2000000

```
save(matriz_proximidades, file = "tablas/matriz_proximidades.RDta")

write_xlsx(matriz_coocurrencias, "tablas/matriz_coocurrencias.xlsx")
write_xlsx(matriz_proximidades, "tablas/matriz_proximidades.xlsx")
```

4.3 Mapa de calor de la matriz de proximidades

El primer gráfico que podemos crear para visualizar las relaciones entre los códigos es un **mapa de calor** de la matriz de proximidades. Para ello, instalamos y cargamos la biblioteca `reshape2`. Luego, cargamos la tibble con las proximidades previamente guardada en la carpeta `tablas`, la convertimos en una matriz, y usamos los nombres de las columnas como nombres de fila (si estos no existen).

```
install.packages("reshape2")
```

```
library(reshape2)
```

```
##
## Attaching package: 'reshape2'
```

```
## The following object is masked from 'package:tidyr':
##
##     smiths
```

```
load("tablas/matriz_proximidades.RDta")
matriz_proximidades <- as.matrix(matriz_proximidades)

if (is.null(rownames(matriz_proximidades))) {
  rownames(matriz_proximidades) <- colnames(matriz_proximidades)
}
```

Convertimos la matriz a formato largo:

```
matriz_larga <- melt(matriz_proximidades)
colnames(matriz_larga) <- c("Elemento1", "Elemento2", "Proximidad")
```

Aseguramos el orden de los factores y filtramos solo la mitad superior de la matriz para evitar duplicados:

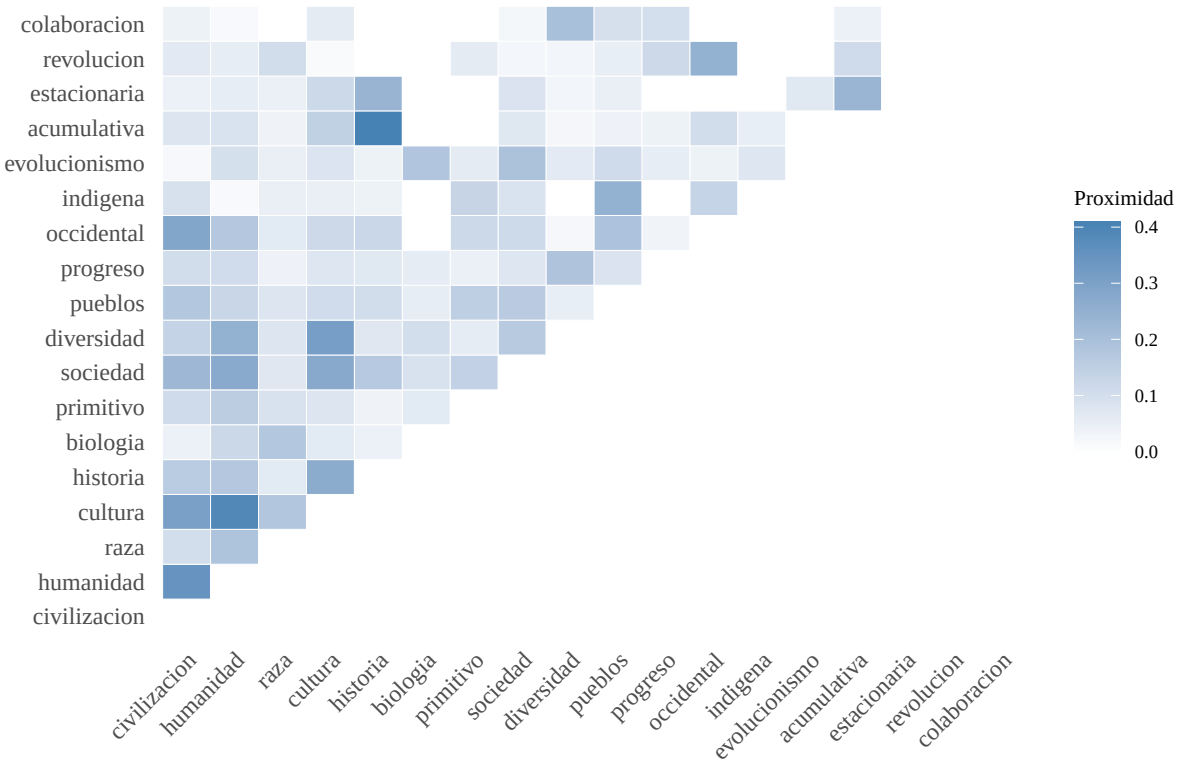
```
matriz_larga$Elemento1 <- factor(matriz_larga$Elemento1,
                                 levels = rownames(matriz_proximidades))
matriz_larga$Elemento2 <- factor(matriz_larga$Elemento2,
                                 levels = colnames(matriz_proximidades))

matriz_larga_filtrada <- matriz_larga %>%
  filter(as.integer(Elemento1) <= as.integer(Elemento2))
```

Finalmente, generamos el gráfico:

```
ggplot(matriz_larga_filtrada, aes(x = Elemento1, y = Elemento2, fill = Proximidad)) +
  geom_tile(color = "white") +
  scale_fill_gradient(low = "white", high = "steelblue") +
  labs(title = "Figura 1: Mapa de calor de proximidades (mitad superior)",
       x = "", y = "") +
  theme_minimal() +
  theme(
    axis.text.x = element_text(angle = 45, hjust = 1, family = "Times"),
    axis.text.y = element_text(family = "Times"),
    plot.title = element_text(size = 10, family = "Times"),
    legend.title = element_text(size = 8, family = "Times"),
    legend.text = element_text(size = 7, family = "Times"),
    panel.grid = element_blank()
  )
```

Figura 1: Mapa de calor de proximidades (mitad superior)



En el mapa de calor observamos algunas relaciones clave entre los códigos que articulan el argumento del texto de Lévi-Strauss. Por ejemplo, los dos tipos de evolucionismo que menciona: **biológico** y **social** (o falso evolucionismo). También se evidencian los dos tipos de historia a los que se refiere: **acumulativa** y

estacionaria. Además, destaca la importancia de la **diversidad** cultural en el desarrollo del **progreso**, entendido como una **colaboración** entre **pueblos** y **culturas**.

El mapa actúa como una guía visual que orienta la exploración de relaciones significativas entre los códigos.

4.4 Escalado multidimensional de la matriz de proximidades

Otro gráfico útil para analizar las relaciones entre los códigos es el **escalado multidimensional (MDS)**. Este método calcula las coordenadas de los elementos en un espacio, generalmente bidimensional, facilitando su interpretación visual.

Para ello, primero debemos convertir la matriz de proximidades en una matriz de distancias:

```
matriz_distancias <- as.dist(1 - matriz_proximidades)
```

Para realizar el escalado multidimensional no métrico (MDS de Kruskal), utilizamos la biblioteca MASS. La instalamos y cargamos, y luego ejecutamos el análisis en dos dimensiones:

```
install.packages("MASS")
```

```
library(MASS)
```

```
##
## Attaching package: 'MASS'
```

```
## The following object is masked from 'package:dplyr':
##
##     select
```

```
fit <- isoMDS(matriz_distancias, k = 2)  # k indica el número de dimensiones
```

```
## initial  value 40.339100
## iter   5 value 26.203661
## final  value 21.786444
## converged
```

```
x <- fit$points[, 1]
y <- fit$points[, 2]
print(fit)
```

```
## $points
##                       [,1]        [,2]
## civilizacion   0.24774133  0.47023208
## humanidad     -0.19440106  0.03063823
## raza          -1.10060077 -0.18458105
## cultura        0.61514611 -0.41909726
## historia       1.14868686  0.01801366
## biologia      -1.76563522 -1.10578260
## primitivo     -1.31974266  1.15608587
## sociedad      -0.24686763 -0.22275242
## diversidad     0.10844714 -1.27097298
## pueblos       -0.39313891  0.81168073
## progreso      -0.06528296 -1.70741900
## occidental     0.01514857  1.33709084
## indigena      -0.96606691  1.88013635
## evolucionismo -1.55534604 -0.70991957
## acumulativa    1.66952482  0.37007373
## estacionaria   1.95688461 -0.35041587
## revolucion     0.86479719  1.94426527
```

```
## colaboracion   0.98070553 -2.04727601
##
## $stress
## [1] 21.78644
```

Al imprimir el resultado, obtenemos las coordenadas de cada código en el plano, así como el **índice de stress**, que en este caso es **21,8%**. Este valor indica la distorsión entre las distancias reales y su representación. En el escalado no métrico, se considera **aceptable un nivel de stress inferior al 20%**, aunque valores cercanos, hasta el 30%, pueden ser interpretables con precaución.

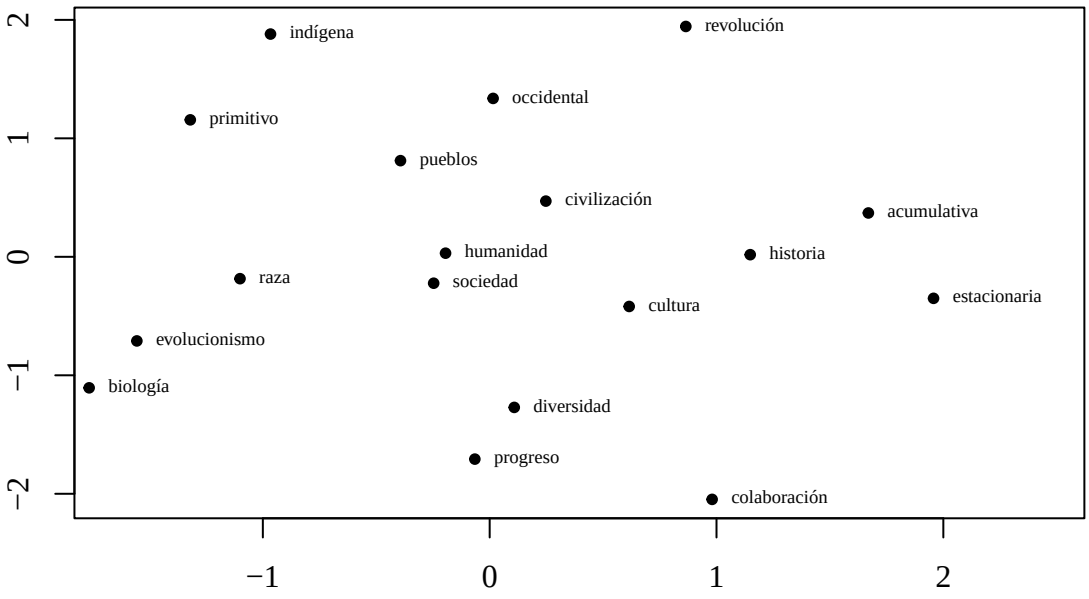
Una vez obtenidas las coordenadas, podemos graficar los puntos y añadir los nombres de los códigos:

```
plot(x, y, pch = 20,
     xlim = range(x) + c(0.1, 0.5),
     xlab = "", ylab = "",
     main = expression(
       "Figura 2: Relaciones entre los códigos de" ~italic("Raza e historia")),
     cex.main = 0.8,
     family = "Times"
     )

raza.names <- c("civilización", "humanidad", "raza", "cultura", "historia", "biología",
                "primitivo", "sociedad", "diversidad", "pueblos", "progreso",
                "occidental","indígena", "evolucionismo", "acumulativa", "estacionaria",
                "revolución", "colaboración")

text(x, y, labels = raza.names, pos = 4, cex = 0.6, family = "Times")
```

Figura 2: Relaciones entre los códigos de *Raza e historia*



4.5 Análisis de conglomerados jerárquico

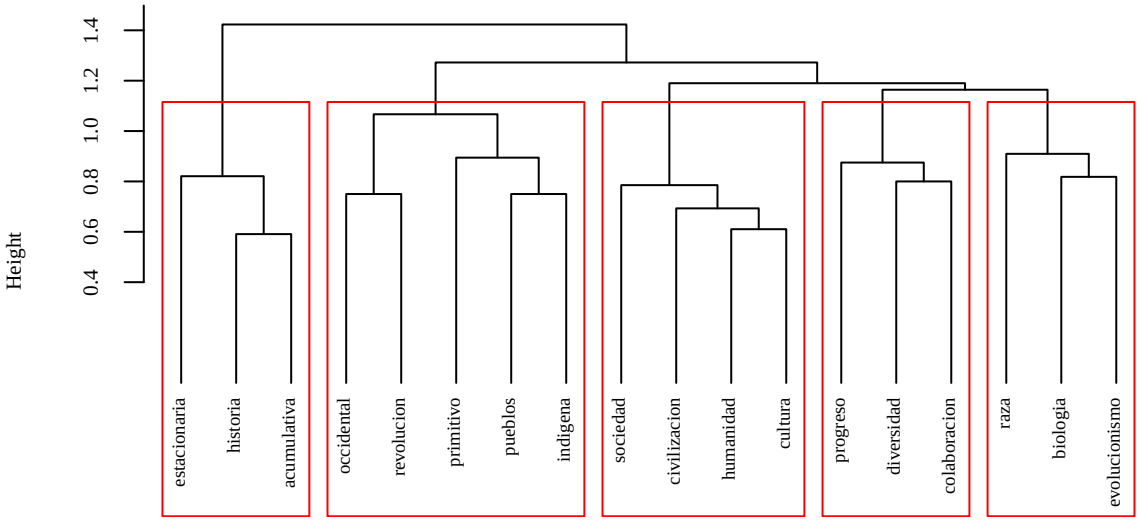
Para observar mejor el agrupamiento de los códigos, realizamos un análisis de conglomerados jerárquico utilizando el algoritmo ward.D.

```r
hclust_ward <- hclust(matriz_distancias, method = "ward.D")
```

Visualizamos los resultados en un dendrograma:

```r
plot(hclust_ward,
     hang = -1,
     cex = 0.6,
     main = expression("Figura 3: Relaciones entre los códigos de" ~italic("Raza e historia")),
     cex.main = 0.8,
     family = "Times",
     xlab = "",
     sub = "",
     cex.lab = 0.7,
     cex.axis = 0.7
     )
grupos <- cutree(hclust_ward, k = 5)  # k = número de clústeres
rect.hclust(hclust_ward, k = 5, border = "red")  # delimitamos los grupos con rectángulos
```

Figura 3: Relaciones entre los códigos de *Raza e historia*



4.6 Análisis de redes

Finalmente, analizamos la red de relaciones entre los códigos utilizando la biblioteca `igraph`. En este caso, las distancias entre los vértices del grafo **no son significativas**.

Primero, instalamos la biblioteca igraph.

```r
install.packages("igraph")
```

```r
library(igraph)
```

```
##
## Attaching package: 'igraph'
```

```
## The following objects are masked from 'package:lubridate':
##
##     %--%, union
```

```
## The following objects are masked from 'package:dplyr':
```

```
##
##     as_data_frame, groups, union

## The following objects are masked from 'package:purrr':
##
##     compose, simplify

## The following object is masked from 'package:tidyr':
##
##     crossing

## The following object is masked from 'package:tibble':
##
##     as_data_frame

## The following objects are masked from 'package:stats':
##
##     decompose, spectrum

## The following object is masked from 'package:base':
##
##     union
```

Luego, nos aseguramos de que la matriz de proximidades con la que vamos a operar es una matriz y no una tabla:

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

Creamos el grafo a partir de la matriz de proximidades:

```
g1 <- graph_from_adjacency_matrix(matriz_proximidades, mode = "undirected",
                                  weighted = TRUE)
```

Eliminamos las aristas con pesos bajos tras examinar la matriz:

```
g1 <- delete.edges(g1, which(E(g1)$weight < 0.09))
```

Usamos dos indicadores para visualizar la red:

- **Tamaño de la arista**: representa la **proximidad** entre códigos.
- **Tamaño del vértice**: depende del **grado** del nodo (número de conexiones).

```
a <- E(g1)$weight      # Tamaño de la arista
d <- degree(g1)        # Grado del vértice
```

Asignamos nombres a los nodos si la matriz no los tiene:

```
if (is.null(V(g1)$name)) {
  V(g1)$name <- colnames(matriz_proximidades)
}
```

Asignamos colores a los vértices según el grupo obtenido del análisis jerárquico:

```
colores_grupo <- rainbow(length(unique(grupos)))
V(g1)$color <- colores_grupo[grupos]
```

Dibujamos la red:

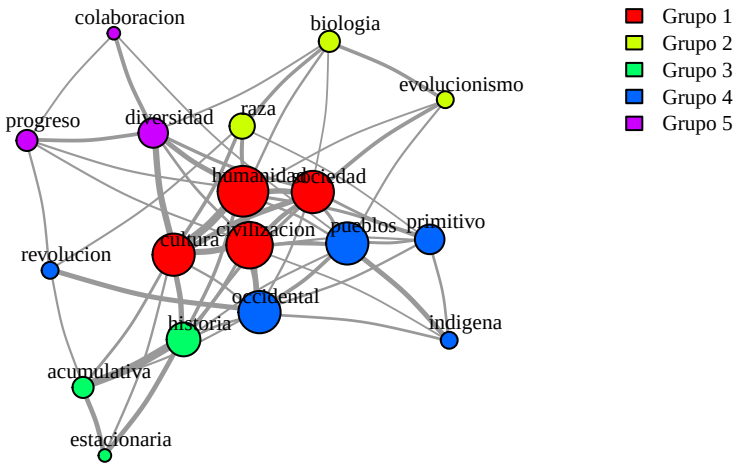
```
par(family = "Times")
set.seed(123)  # Para reproducibilidad del layout
plot(g1,
     layout = layout_with_fr(g1),  # Layout de Fruchterman-Reingold
```

```
     vertex.size = d * 2,
     vertex.label.cex = 0.7,
     vertex.label.color = "black",
     vertex.label.dist = 1.5,
     edge.width = a * 10,
     edge.color = "gray60",
     edge.curved = 0.1
)

title(main = expression(
  "Figura 4: Relaciones entre los códigos de" ~italic("Raza e historia")),
  cex.main = 0.8, family = "Times")

legend("topright", legend = paste("Grupo", 1:5), fill = colores_grupo, bty = "n",
       cex = 0.7)
```

Figura 4: Relaciones entre los códigos de *Raza e historia*



5 Mapeado conceptual del texto

Una vez analizada la codificación del texto, podemos representar un **mapa conceptual** que muestre la red semántica, los conceptos identificados y las relaciones que el autor ha establecido entre ellos para construir sus argumentos.

Usamos la biblioteca `DiagrammeR`. El procedimiento consiste en definir los nodos (en mayúsculas), las relaciones (dirigidas o no), y personalizar su estilo visual.

```
install.packages("DiagrammeR")
```

```
library(DiagrammeR)
```

```
##
## Attaching package: 'DiagrammeR'
## The following object is masked from 'package:igraph':
##
##     count_automorphisms
```

```
grafo <- DiagrammeR::grViz("
digraph {
    graph [label = 'Figura 5: Mapa de conceptos de Raza e historia', fontsize = 16,
    fontname = Times]
    node [shape = ellipse, fontname = Times, fontsize = 12, style = filled]
    edge [fontname = Times, fontsize = 10]

    A [label = 'razas', fillcolor = 'lightblue']
    B [label = 'biología', fillcolor = 'lightblue']
    C [label = 'evolucionismo', fillcolor = 'lightblue']
    D [label = 'humanidad', fillcolor = 'limegreen']
    E [label = 'sociedades', fillcolor = 'limegreen']
    F [label = 'culturas', fillcolor = 'limegreen']
    G [label = 'historia', fillcolor = 'yellow']
    H [label = 'acumulativa', fillcolor = 'yellow']
    I [label = 'estacionaria', fillcolor = 'yellow']
    J [label = 'progreso', fillcolor = 'limegreen']
    K [label = 'diversas', fillcolor = 'limegreen']
    L [label = 'colaboran', fillcolor = 'limegreen']
    M [label = 'homogeneización', fillcolor = 'tomato']
    O [label = 'civilizaciones', fillcolor = 'yellow']
    P [label = 'occidental', fillcolor = 'palegreen']
    R [label = 'revolución industrial', fillcolor = 'palegreen']
    S [label = 'pueblos', fillcolor = 'yellow']
    T [label = 'primitivos', fillcolor = 'yellow']
    U [label = 'indígenas contemporáneos', fillcolor = 'yellow']
    V [label = 'arcaicas', fillcolor = 'yellow']
    W [label = 'conjetural', fillcolor = 'yellow']

    A -> B [label = 'son producto de']
    B -> C [label = 'está sujeto a']
    D -> E [label = 'está dividida en']
    E -> F [label = 'tienen']
    F -> G [label = 'tienen']
    F -> A [label = 'y']
    F -> K [label = 'son']
    G -> H [label = 'puede ser vista']
    G -> I [label = 'puede ser vista']
    W -> J [label = 'se identifica erróneamente']
    W -> S [label = 'basada en analogías entre']
    H -> W [label = 'o']
    J -> M [label = 'riesgo']
    K -> L [label = 'y']
    L -> J [label = 'dan lugar a']
    L -> O [label = 'dieron lugar a']
    M -> F [label = 'de']
    O -> P [label = 'una de las cuales es la']
    P -> R [label = 'produjo']
    R -> J [label = 'es un ejemplo de']
    S -> T
    S -> U
    W -> C [label = 'es falso']
    W -> O [label = 'basada en evidencias']
```

```
    O -> V
}
")
grafo
```

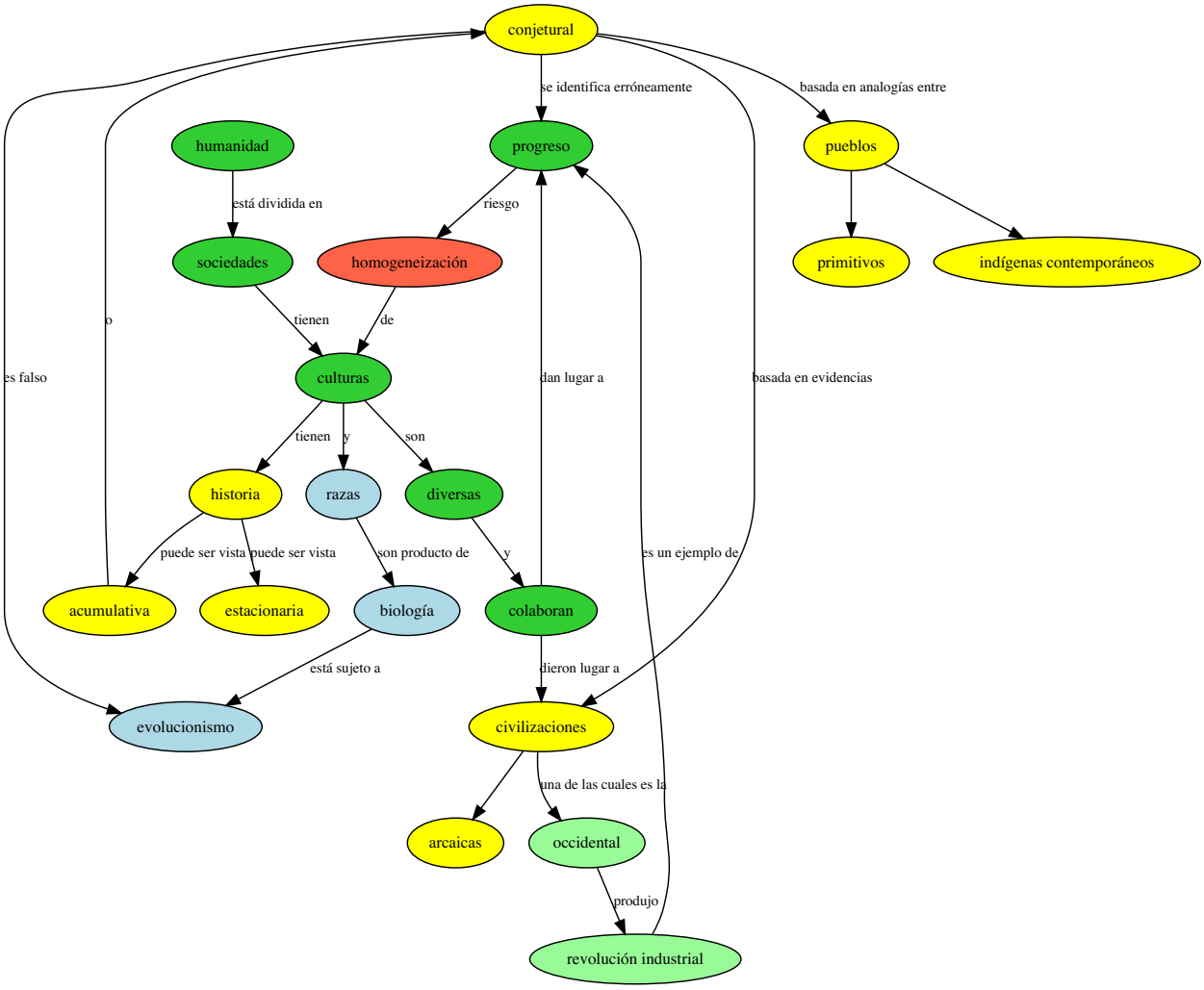


Figura 5: Mapa de conceptos de Raza e historia

El texto analizado expone una argumentación en torno a la diversidad cultural y su papel en el desarrollo histórico y social de la humanidad. A través de una crítica al evolucionismo social, se articula una defensa de la diversidad y la colaboración entre culturas como motor de progreso. A continuación, se expone un resumen estructurado de los principales puntos argumentativos:

1. Distinción entre raza y cultura

El autor comienza diferenciando la **raza**, entendida como una categoría **biológica**, de la **cultura**, de carácter social. Señala que, aunque existen tanto **diversidad** de **razas** como de **culturas**, estas últimas son mucho más numerosas y complejas. A partir de ello, plantea la pregunta sobre si la **diversidad** de **culturas** representa una ventaja o una desventaja para la **humanidad**, lo cual requiere previamente aclarar qué se entiende por **diversidad cultural**.

2. Etnocentrismo y relativismo cultural

Se afirma que, aunque la **diversidad cultural** es natural, los **seres humanos** tienden al etnocentrismo. El relativismo cultural —la idea de respetar las **culturas** ajenas— puede resultar paradójico, pues a menudo

conlleva el rechazo de lo que se considera “barbarie”, término que denota una **cultura** diferente percibida como inferior.

3. Crítica al evolucionismo social

El texto distingue entre el **evolucionismo biológico**, aceptado como teoría científica, y el **evolucionismo social**, cuya validez cuestiona. Critica especialmente la **historia** conjetural que proponen los **evolucionistas sociales**, señalando que esta se basa en analogías erróneas entre **pueblos primitivos** y **grupos indígenas** contemporáneos, ignorando sus diferencias contextuales.

4. El progreso humano no es lineal

Se rechaza la idea de un **progreso** lineal y unidireccional de la **humanidad**. En lugar de ello, se argumenta que los avances culturales han ocurrido en diferentes **sociedades**, en distintos lugares y momentos históricos, sin que puedan atribuirse a una sola **raza** o **cultura**.

5. Dos tipos de historia: acumulativa y estacionaria

El autor diferencia entre una **historia acumulativa**, vista como progresiva y triunfante (asociada a la **civilización occidental**), y una **estacionaria**, correspondiente a **culturas** que no siguieron ese modelo. Sin embargo, cuestiona esta dicotomía como una visión etnocéntrica. Todas las **historias** serían, en algún sentido, **acumulativas** desde su propia lógica cultural.

6. La aportación de diversas culturas a la civilización occidental

Se sostiene que la **civilización occidental** no habría alcanzado su posición actual sin la contribución de **culturas diversas**. La historia del **progreso** no es atribuible a una sola tradición cultural.

7. El papel de las circunstancias sociales e históricas

El desarrollo de innovaciones **culturales** y **sociales** no puede explicarse solo por el azar o por la acción de individuos destacados. Las condiciones históricas, económicas y sociales han sido determinantes. Así, procesos clave como la **revolución** neolítica o la **revolución** industrial no se explican por una **cultura** específica.

8. Interacción cultural y formación de civilizaciones

Las **culturas** y **sociedades** no son entidades aisladas. A través de las migraciones, el comercio y los conflictos, las **culturas** han intercambiado elementos. La **civilización** es el resultado de una red de contactos, préstamos y colaboraciones entre **culturas diversas**, no de una sola línea cultural.

9. Colaboración cultural y progreso

El **progreso** ha sido impulsado por la interacción y la **colaboración** entre **culturas** a lo largo de la **historia**. Esta **diversidad cultural**, motor del cambio, plantea sin embargo un riesgo: la homogeneización cultural. El desafío consiste en mantener la **diversidad cultural** sin renunciar a la **colaboración**.

Referencias

Guest, Greg, Kathleen M. MacQueen, y Emily E. Namey. 2011. *Applied thematic analysis*. Sage Publications.

Levi-Strauss, Claude. 1979. «Raza e historia». En *Antropología estructural*, 304-39. Siglo XXI Editores.

Ryan, Gery W., y H. Russell Bernard. 2003. «Techniques to identify themes». *Field methods* 15 (1): 85-109.

Silge, Julia, y David Robinson. 2017. *Text mining with R: A tidy approach*. O’reilly Boston (MA).