UNIVERSIDAD DE GUANAJUATO

Programación en R

Reconocimiento de patrones

Dr. Mario Alberto Ibarra Manzano
ibarram@ugto.mx

https://github.com/ibarram/ReconocimientoPatrones

Agenda

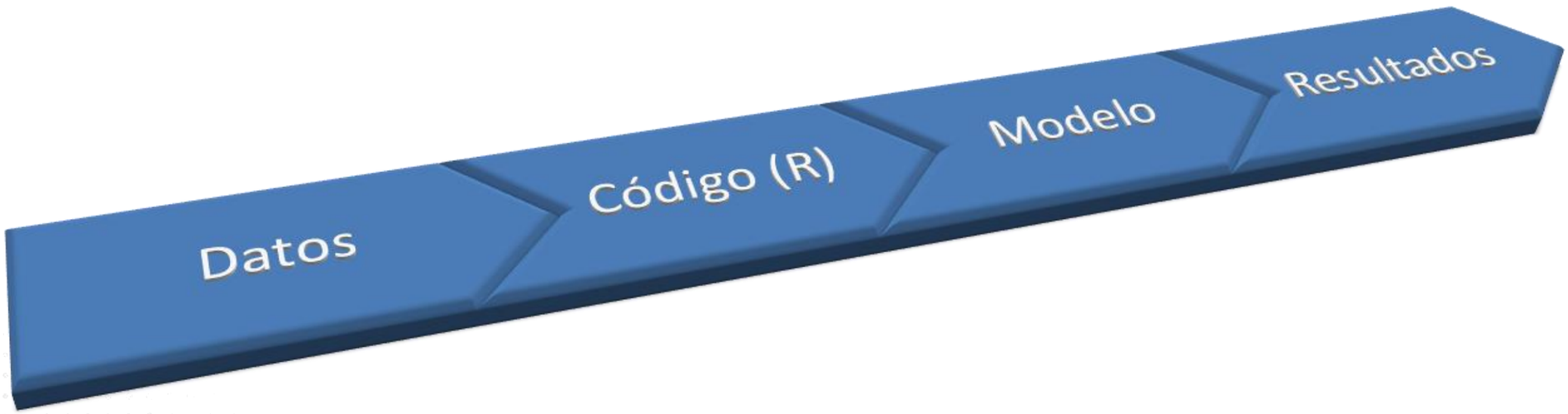
- Qué es R y qué no es
- RStudio: intefaz y flujo de trabajo
- Objetos, tipos y estrcturas de datos
- Importar datos, paquetes y el "tidyverse"
- Transformación con dplyr y visulización con ggplot2
- Errores típicos, ayuda y buenas práticas

¿Qué es R?

Piensa en R como:

- Un lenguaje para manipular datos, programar y modelar
- Un ecosistema de paquetes (CRAN, Bioconductor)
- Una forma reproducible de trabajar: script → resultados → reporte

Idea clave: todo lo queda en código → se puede repetir y auditar



Instalación y herramientas

Proceso de instalación:

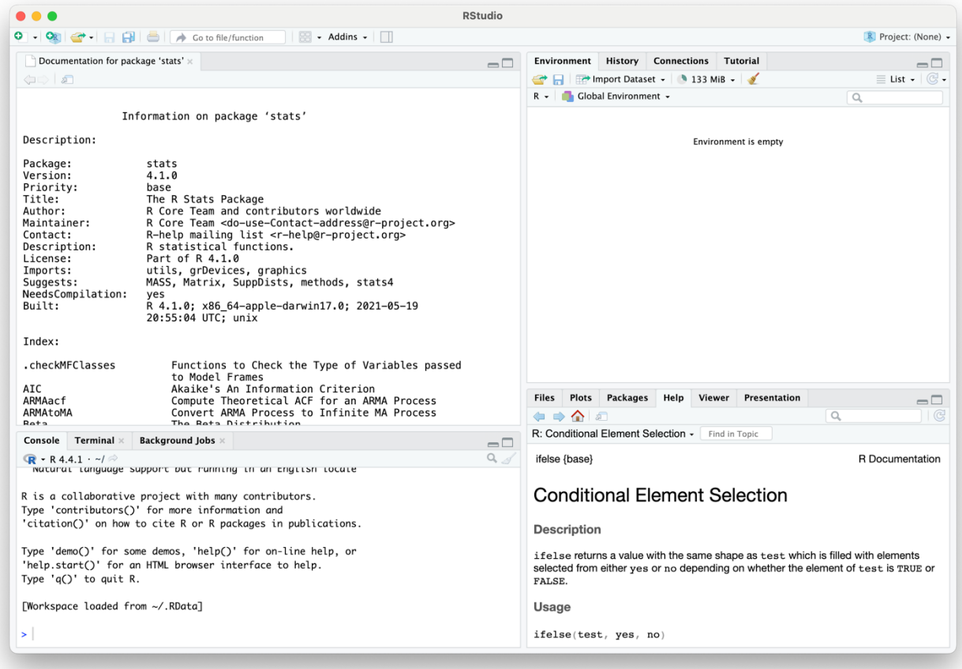
1. Instalar R (motor) desde CRAN
   https://www.r-project.org/

2. Instalar RStudio Desktop (IDE)
   https://posit.co/downloads/

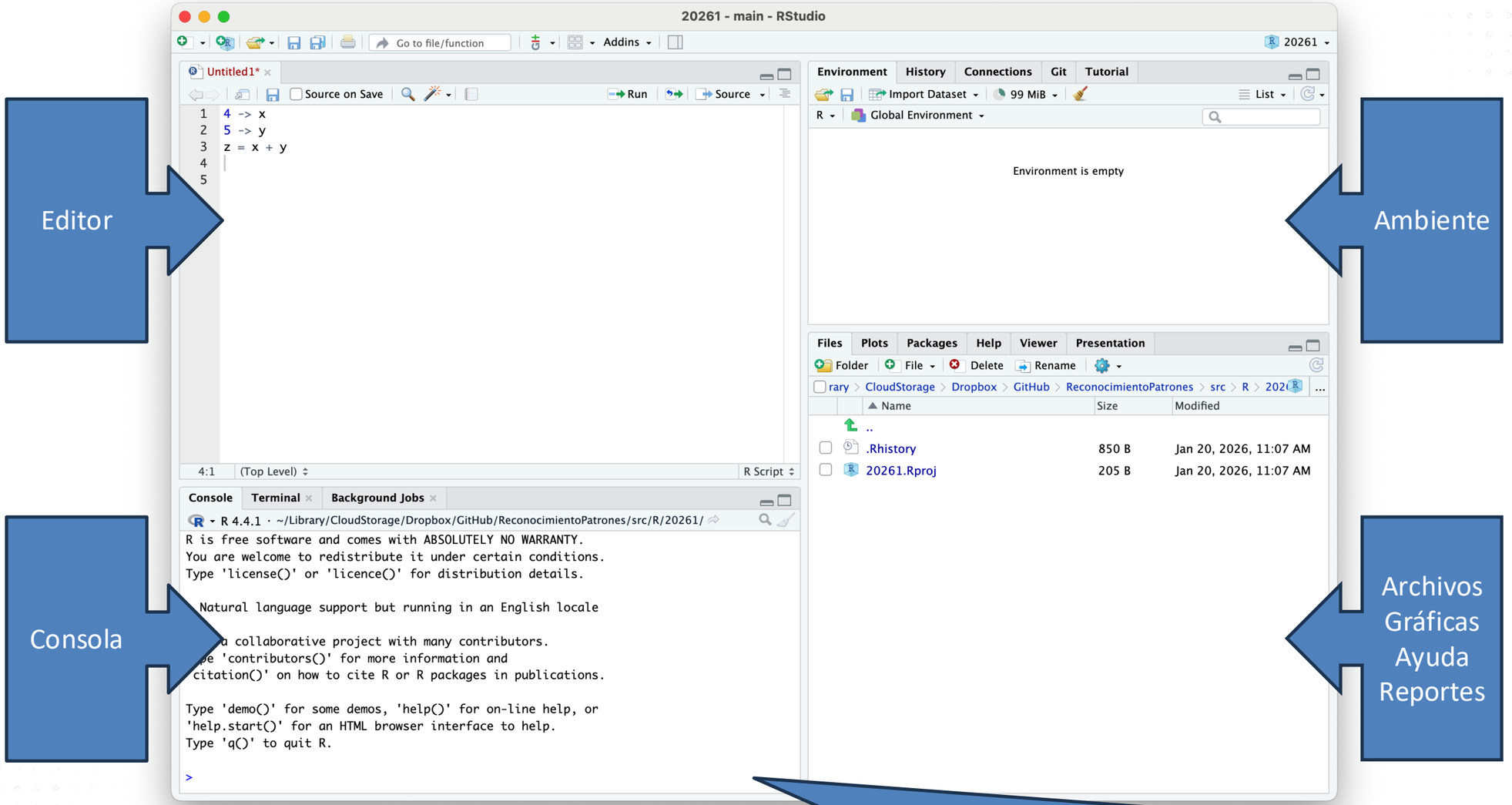
3. Abrir RStudio Desktop → Crear un "Project" nuevo
   Menu → File → New Project → New Directory → Directory name

4. Instalar paquetes: **tidyverse** y **tidymodels**
   Tools → Install Packages … → Packages → tidyverse, tidymodels

5. Crear un nuevo código fuente (R)
   File → New File → R Script (⌘+ ⇧ +N)



Interfaz de RStudio (4 paneles básicos)



Editor

Ambiente

Consola

Archivos
Gráficas
Ayuda
Reportes

Todo lo importante va en Source; Console, solo para probar; Environment, para verificar objetos; y el panel inferior derecho, para ver archivos, gráficas y ayuda.

Interfaz de RStudio (4 paneles básicos)

En RStudio (vista por defecto) la interfaz se organiza en cuatro paneles, cada uno con un propósito claro:

- **Editor / Source (arriba–izquierda):**
  ¿Qué es?: El espacio donde escribes y editas código en archivos (.R, .qmd, .Rmd, .Rproj, etc.).
  ¿Para qué se usa?: Preparar scripts, funciones, análisis reproducibles (Quarto/RMarkdown), ejecutar líneas o bloques seleccionados y guardar tu trabajo.

- **Console (abajo–izquierda):**
  ¿Qué es?: La línea de comandos interactiva de R (REPL).
  ¿Para qué se usa?: Ejecutar instrucciones "al vuelo", probar funciones rápidamente, ver mensajes/errores inmediatos y correr comandos que no necesariamente quieres guardar en un script.

- **Environment / History (arriba–derecha):**
  ¿Qué es?: Un panel de "estado" de tu sesión de R.
  ¿Para qué se usa?:
  Environment: Ver los objetos creados (dataframes, modelos, variables), su tipo y tamaño; cargar/guardar objetos (.RData, .Rds) de forma guiada.
  History: Revisar el historial de comandos ejecutados para reutilizarlos.

- **Files / Plots / Packages / Help / Viewer (abajo–derecha):**
  ¿Qué es?: Un panel multipestaña para trabajar con archivos y resultados.
  ¿Para qué se usa?:
  Files: Navegar por carpetas del proyecto, abrir y renombrar archivos.
  Plots: Visualizar y exportar gráficas.
  Packages: Instalar o cargar paquetes y consultar su documentación.
  Help: Consultar la ayuda sobre funciones y paquetes.
  Viewer: Mostrar contenido renderizado (p. ej., HTML de Quarto, Shiny, widgets).

Tipos de datos (atómicos)

Tipo (R)	Ejemplo (R)	¿Qué representa?	Alcance / uso típico	Operaciones comunes	Notas importantes
logical	c(TRUE,FALSE)	Booleanos	Filtros, máscaras, reglas, condiciones	&, `	, !, if, which()`
integer	10L, 1:5	Enteros	Conteos, índices, IDs discretos	+ - *, seq(), length()	Distingue 10 (double) vs 10L(integer)
double / numeric	3.14	Reales (punto flotante)	Cálculo numérico y estadística	mean(), sd(), %*%, lm( )	Puede haber errores de redondeo (0.1+0.2 != 0.3)
complex	1+2i	Complejos	FFT, señales, álgebra compleja	Re(), Im(), Mod(), Arg()	Menos común en tabular clásico, más en DSP
character	"texto"	Texto	Etiquetas, categorías crudas, parsing, rutas	paste(), sprintf(), ncha r(), grepl()	Para “categorías”, suele convertirse a factor o one-hot
raw	charToRaw("Hi")	Bytes	IO binario, hashes, protocolos	rawToChar(), lectura/escritura binaria	No se usa para cálculos; es para datos binarios
factor	factor(c("A","B"))	Categórico con niveles	Variables cualitativas, clase objetivo en clasificación	levels(), relevel(), table ()	Los **niveles** importan; cuidado al predecir con niveles nuevos
Date	as.Date("2026-01-20")	Fecha (día)	Series por día, calendarios, cortes temporales	+/- días, difftime()	No tiene hora; para hora usa POSIXct
POSIXct	as.POSIXct("...")	Fecha-hora	Timestamps, sensores, logs	format(), difftime(), lu bridate	Siempre considera tz (zona horaria)

Tipos de datos

```
# logical
flag <- TRUE
x <- c(TRUE, FALSE, TRUE)

# integer
n <- 10L
idx <- 1:5

# double/numeric
a <- 3.14159
v <- c(0.1, 2.5, -7)

# complex
z <- 3+5i
Conj(z)
z1 <- complex(real=3, imaginary =-4)
Re(z1)
Img(z1)
z2 <- complex(modulus = 2, argument = pi/3)
Mod(z2)
Arg(z2)

# character
a <- "Reconocimiento de Patrones"
names <- c("Ana", " María", "Luis")
paste(names[1], names[2])
sprintf("acc=%.3f", 0.92567)
nchar(a)
sort(names)
order(names)
substr(a, 5, 8)
strsplit(x, “e”)
grep(”a”, names)
grepl(”a”, names)
gregexpr("a", names)
toupper(a)
tolower(a)
unique(c("A", "B", "A"))
table(c("A", "B", "A"))

# factor
factor(x, levels = ..., labels = ..., ordered = FALSE)
x <- c("A","B","A","C")
f <- factor(x)
is.factor(f)
levels(f)
nlevels(f)
str(f)
f2 <- relevel(f, ref = "B")
f_ord <- factor(c("bajo","alto","medio"), levels =
c("bajo","medio","alto"), ordered = TRUE)
table(f)
summary(f)
```

Tipos de datos (Data Frames)

```r
# Crear
df <- data.frame(
        id = 1:3,
        sexo = c("F", "M", "F"),
        score = c(0.8, 0.4, 0.9)
)

# Inspección rápida
head(df)        # primeras filas
tail(df)        # ultimas filas
str(df)         # estructura y tipos
summary(df)     # resumen
dim(df)         # numero de elementos
names(df)       # nombres de columnas

# Acceso e indezación
df$score                # una columna
df[, "score"]           # columna por nombre
df[1:2, ]               # filas 1 a 2
df[ , c("id", "score")] # varias columnas
df[df$score>0.5, ]      # filtrar por condición
```

```r
# Seleccionar columnas
df2 <- df[c("id", "score")]

library(dplyr)
df2 <- df %>% select(id, score)

# Filtrar filas
df_d <- df[df$sexo == "F" & df$score >= 0.8, ]

df_f <- df %>% filter(sexo == "F", score >= 0.8)

#Crear/transformar columnas
df$aprobado <- df$score >= 0.7
df$score_z <- scale(df$score)

df <- df %>% mutate(
  aprobado = score >= 0.7,
  score_z = as.numeric(scale(score))
)
```

Tipos de datos (Data Frames)

```
# Estadísticos
mean(df$score)
tapply(df$score, df$sexo, mean)   # por grupo

df %>% summarise(media = mean(score), sd = sd(score))
df %>% group_by(sexo) %>%
  summarise(media = mean(score), n = n())

# Ordenar filas
df_ord <- df[order(df$score, decreasing = TRUE), ]

df_ord <- df %>% arrange(desc(score))

# Renombrar columnas
names(df)[names(df) == "score"] <- "puntaje"

df <- df %>% rename(puntaje = score)

# Exportar/guardar resultados
write.csv(df, "salida.csv", row.names = FALSE)

saveRDS(df, "salida.rds")
df2 <- readRDS("salida.rds")
```

```
# Manejo de NA
sum(is.na(df$score))
df$score[is.na(df$score)] <- mean(df$score, na.rm = TRUE)  #
imputación simple

df %>% summarise(na_score = sum(is.na(score)))

# Unir tablas (joins)
df_a <- data.frame(id = 1:3, sexo = c("F","M","F"))
df_b <- data.frame(id = c(1,3), score = c(0.8, 0.9))

left_join(df_a, df_b, by = "id")      # conserva todo df_a
inner_join(df_a, df_b, by = "id")   # solo coincidencias

# Reestructurar (wide/long)
df_long <- pivot_longer(df, cols = c(score), names_to = "var",
values_to = "val")
df_wide <- pivot_wider(df_long, names_from = var, values_from =
val)
```

Análisis de sesgo

Una universidad pública utiliza desde el año 2000 un sistema automático que valida a aspirantes para ingresar a 10 facultades con 25 carreras. Cada año hay ~10000 aspirantes y se admite aproximadamente al 30 %. En los datos globales se observa que, entre los admitidos, ~70 % son hombres (varía según el año). Se sospecha “sesgo de género”. Sin embargo, la hipótesis alternativa es que no existe sesgo del sistema.
Tu tarea es determinar, con evidencia, si el sistema presenta sesgo de género.

Qué incluye el dataset – data/sesgo/
Nivel programa-año (625 filas = 25 años × 25 carreras)
Columnas principales:
año
facultad
carrera
tasa_admision_programa (misma para ambos géneros en esa carrera-año; el “sistema” decide con esa tasa)
aspirantes_mujeres, admitidas_mujeres
aspirantes_hombres, admitidos_hombres
(derivadas en el CSV/Excel) tasa_mujeres, tasa_hombres, aspirantes_total, admitidos_total

Ejemplo001.R

```
# Archivo de la base de datos
filename =
"../../../data/sesgo/caso_sesgo_genero_admision_2000_2
024_programa_anio.csv"

# Lectura del archivo CSV
datos <- read.csv(filename)

# Visualizacion de datos
print(head(datos))

# Tamaño de la tabla
sz <- dim(datos)

# Selección de los único de las columnas de anio, facultad
y escuela
lb_a <- unique(datos$anio)
lb_f <- unique(datos$facultad)
lb_c <- unique(datos$carrera)

# Función para la suma en base de una selección
f <- function(vct, sl, vtc_d) smn <- sum(vtc_d[vct==sl])
# Vectorización de la selección
vf <- Vectorize(f, vectorize.args = "sl")

# Admitidos por anio
naat <- vf(datos$anio, lb_a, datos$admitidos_total)
naah <- vf(datos$anio, lb_a, datos$admitidos_hombres)
naam <- vf(datos$anio, lb_a, datos$admitidas_mujeres)
t_ah <- naah/naat
t_am <- naam/naat

# Grafica de aceptación por género desde 2000 a 2024
par(mar = c(6, 4, 4, 2) + 0.1, xpd = NA)
plot(lb_a, t_ah, type = 'o', xlab = "Año", ylab="Tasa", lwd = 2,
     col = "blue", ylim = c(.2, .8), lty = "solid", pch = 19,
     main = "Tasa de aceptación por género")
grid()
lines(lb_a, t_am, type = 'o', lwd = 2,
     col = "red", lty = "solid", pch = 19)
legend("bottomleft", inset = c(0, -1), legend = c("Hombres",
"Mujeres"),
       col = c("blue", "red"), lty = 1, pch = 19, lwd = 2, bty = "n")
```

Ejemplo001.R

```
# Archivo de la base de datos
filename =
"../../../data/sesgo/caso_sesgo_genero_admision_2000_2
024_programa_anio.csv"

# Lectura del archivo CSV
datos <- read.csv(filename)

# Visualizacion de datos
print(head(datos))

# Tamaño de la tabla
sz <- dim(datos)

# Selección de los único de las columnas de anio, facultad
y escuela
lb_a <- unique(datos$anio)
lb_f <- unique(datos$facultad)
lb_c <- unique(datos$carrera)

# Función para la suma en base de una selección
f <- function(vct, sl, vtc_d) smn <- sum(vtc_d[vct==sl])
# Vectorización de la selección
vf <- Vectorize(f, vectorize.args = "sl")

# Admitidos por anio
naat <- vf(datos$anio, lb_a, datos$admitidos_total)
naah <- vf(datos$anio, lb_a, datos$admitidos_hombres)
naam <- vf(datos$anio, lb_a, datos$admitidas_mujeres)
t_ah <- naah/naat
t_am <- naam/naat

# Grafica de aceptación por género desde 2000 a 2024
par(mar = c(6, 4, 4, 2) + 0.1, xpd = NA)
plot(lb_a, t_ah, type = 'o', xlab = "Año", ylab="Tasa", lwd = 2,
     col = "blue", ylim = c(.2, .8), lty = "solid", pch = 19,
     main = "Tasa de aceptación por género")
grid()
lines(lb_a, t_am, type = 'o', lwd = 2,
     col = "red", lty = "solid", pch = 19)
legend("bottomleft", inset = c(0, -1), legend = c("Hombres",
"Mujeres"),
       col = c("blue", "red"), lty = 1, pch = 19, lwd = 2, bty = "n")
```

TÍTULO

UNIVERSIDAD DE GUANAJUATO