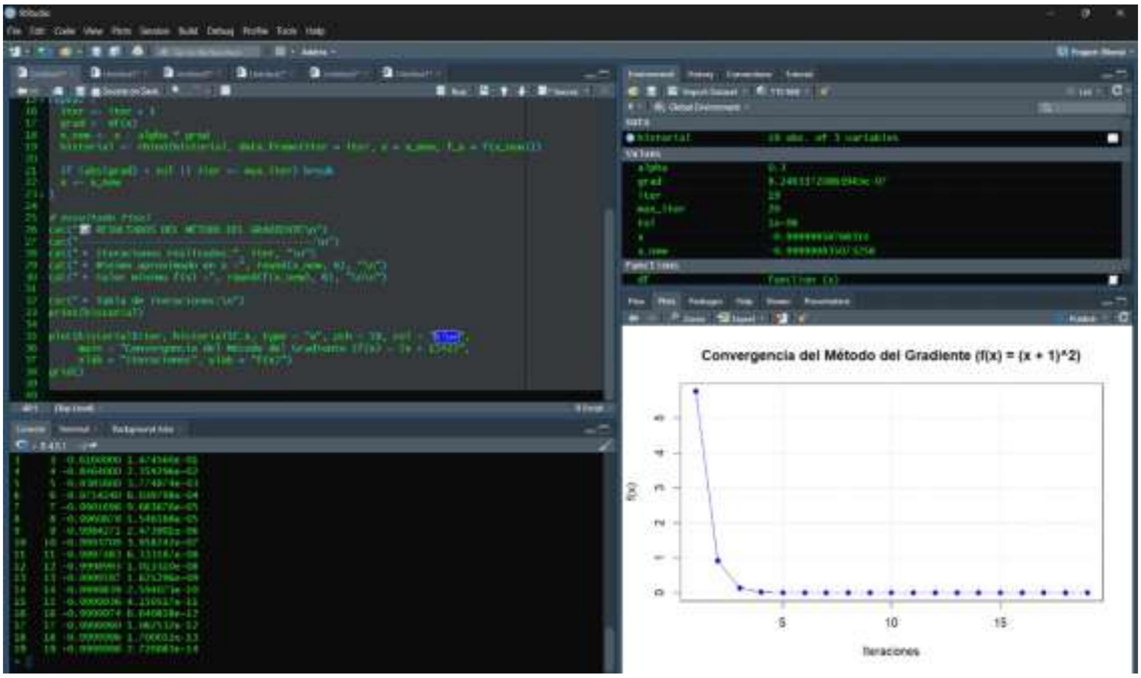
El método de la gradiente consiste en busca el valor óptimo de una función ajustando sus variables paso a paso, siguiendo la dirección contraria al gradiente, hasta que la pendiente sea casi cero.

Código en R:

```r
# f(x) = (x + 1)^2  → mínimo real en x = -1

f  <- function(x) (x + 1)^2
df <- function(x) 2 * (x + 1)

x <- 5
alpha <- 0.3
tol <- 1e-6
max_iter <- 20

iter <- 0
historial <- data.frame(iter = numeric(), x = numeric(), f_x = numeric())

repeat {
  iter <- iter + 1
  grad <- df(x)
  x_new <- x - alpha * grad
  historial <- rbind(historial, data.frame(iter = iter, x = x_new, f_x = f(x_new)))

  if (abs(grad) < tol || iter >= max_iter) break
  x <- x_new
}

# Resultado final
cat("📊 RESULTADOS DEL MÉTODO DEL GRADIENTE\n")
cat("----------------------------------------\n")
cat("🔹 Iteraciones realizadas:", iter, "\n")
cat("🔹 Mínimo aproximado en x =", round(x_new, 6), "\n")
```

```
cat(" ◆ Valor mínimo f(x) =", round(f(x_new), 6), "\n\n")

cat(" ◆ Tabla de iteraciones:\n")
print(historial)

plot(historial$iter, historial$f_x, type = "o", pch = 19, col = "blue",
  main = "Convergencia del Método del Gradiente (f(x) = (x + 1)^2)",
  xlab = "Iteraciones", ylab = "f(x)")
grid()
```



Trabajo 2 :

Comparación entre métodos de OLS y GD en “R”, de tal manera que muestre el tiempo de procesamiento con la librería “profvis”.

```
# Instalación de profvis si no lo tienes
# install.packages("profvis")
library(profvis)

# Datos simulados
set.seed(123)
```

```r
n <- 10000
x <- runif(n, 0, 10)
y <- 3*x + 5 + rnorm(n, 0, 2)


#  Función OLS

ols_fun <- function(x, y) {
  modelo <- lm(y ~ x)
  return(modelo)
}


#  Función Gradient Descent

gd_fun <- function(x, y, alpha = 0.00001, iter_max = 5000) {
  m <- 0
  b <- 0
  n <- length(y)

  for (i in 1:iter_max) {
    y_pred <- m*x + b
    dm <- (-2/n) * sum(x * (y - y_pred))
    db <- (-2/n) * sum(y - y_pred)
    m <- m - alpha * dm
    b <- b - alpha * db
  }

  return(list(m = m, b = b))
}
```

```r
#  Medimos el tiempo de procesamiento


cat("==== MÉTODO OLS ====\n")
tiempo_ols <- system.time({
  modelo_ols <- ols_fun(x, y)
})
print(tiempo_ols)

cat("\n==== MÉTODO GRADIENTE DESCENDENTE ====\n")
tiempo_gd <- system.time({
  modelo_gd <- gd_fun(x, y)
})
print(tiempo_gd)


cat("\n RESULTADOS COMPARATIVOS:\n")
cat("Tiempo OLS: ", round(tiempo_ols["elapsed"], 5), "segundos\n")
cat("Tiempo GD : ", round(tiempo_gd["elapsed"], 5), "segundos\n")
cat("\nCoeficientes:\n")
cat("OLS -> pendiente:", coef(modelo_ols)[2], " intercepto:", coef(modelo_ols)[1], "\n")
cat("GD  -> pendiente:", modelo_gd$m, " intercepto:", modelo_gd$b, "\n")


profvis({
  ols_fun(x, y)
  gd_fun(x, y)
})
```

Trabajo 3: resumen

La Pseudoinversa de Moore-Penrose y el Descenso por Gradiente en la Regresión Lineal

La pseudoinversa de Moore-Penrose es preferible para problemas pequeños o medianos, incluso con datos mal condicionados: ofrece soluciones exactas, rápidas y estables. El descenso por gradiente es más adecuado adecuado para datasets masivos pero requiere normalización, ajuste de hiperparámetros y regularización para evitar inestabilidad.

El estudio recomienda considerar métodos híbridos (usar la pseudoinversa como punto inicial para GD) y explorar técnicas regularizadas (Ridge, LASSO) o variantes avanzadas de GD.

Donde se realizaron experimentos con datos sintéticos (controlar el tamaño, la dimensionalidad y el grado de condicionamiento) y con datasets reales (California Housing y Diabetes).

* Pseudoinversa: calculada con SVD mediante numpy.linalg.pinv.
* Descenso por gradiente: versión batch con tasa de aprendizaje fija ($\alpha = 0.01$), hasta convergencia o 10.000 iteraciones.

Se evaluaron tres métricas principales: tiempo de ejecución, error cuadrático medio (MSE) y número de iteraciones.

En conclusión:
la pseudoinversa destaca por su exactitud y rapidez en tamaños moderados, mientras que el descenso por gradiente ofrece escalabilidad en grandes volúmenes de datos.

Trabajo 4:

Repoatage de Comparación:

Minimos Cuadrados (OLS) VS Gradiente (GD)

• Descripción General

_OLS (Ordinary Least Squares)

Es un metodo analitico que encuentra directamente los coeficiente que minimisa la suma de los error al cuadrado entre los valores Observados y los predichos

_GD (Gradient Descent)

Es un metodo iterativo que ajusta los coeficientes paso a paso en dirección del gradiente negativo por error, No busca una solución cerrado, si no que se aproxima mediante repeticiones controladas

II Comparaciones entre OLS y GD

OLS

• Velocidad y rendimiento

Rapido para conjunto pequeños y medianos de datos, ya que obtiene la solución en un solo Calculo, Sin embargo puede ser muy costoso en tiempo y memoria

• Presicion y estabilidad

Ofrece una solución exacta Muy estable en datos bien condicionados

• Interpretación practica

tarda unos milesimas en obtener

GD

• Velocidad y rendimiento

Más lento en problemas pequeños pero eficiente en grandes volumenes de dato Su tiempo depende del numero de iteraciones.

• Presición y estabilidad

Puede ser muy inestable si x es mal elegido, Su resultado es aproximado pero es suficientemente bueno

• Interpretacion practica

tarda segundo dependiendo del numero

II Conclución

OLS es mas rapido y preciso para problemas simples y moderados, GD es más lento y depende de parametros es flexible y escalable para grandes conjunto de datos