Universidad Nacional del Altiplano
Facultad de Ingeniería Estadística e Informática

Docente:

Ing. Torres Cruz Fred

Alumno:

Ticona Miramira Roberto Angel

Trabajo 7 - Gradiente de una función

Concepto del gradiente

El **gradiente** representa la tasa de cambio o pendiente de una función respecto a sus variables. En el caso de una sola variable, el gradiente se reduce simplemente a la derivada de la función.

Para una función $f(x)$, el método del descenso del gradiente busca el mínimo actualizando los valores de x de la siguiente forma:

$$x_{i+1} = x_i - \eta\, f'(x_i)$$

donde:

- x_i: valor actual de la variable.
- η: tasa de aprendizaje (learning rate).
- $f'(x_i)$: derivada o pendiente de la función en x_i.

El proceso se repite hasta que $f'(x)$ se acerque a cero, es decir, cuando se alcanza el mínimo de la función.

Implementación en R

El siguiente código en R simula el proceso del gradiente descendente para la función $f(x) = x^2$, cuya derivada es $f'(x) = 2x$. Se observa cómo los valores de x van disminuyendo gradualmente hasta aproximarse al punto mínimo en $x = 0$.

```r
# ------------------------------------------------------------
# Simulación y gráfico de f(x), f'(x) y gradiente en R
# grad guarda el "nuevo" valor de x: x(i) - n * f'(x(i))
# ------------------------------------------------------------

# Parámetro n
n <- 0.01

# Definimos la función f(x) = x^2 y su derivada f'(x) = 2x
f <- function(x) x^2
f_deriv <- function(x) 2 * x

# Valor inicial y número de iteraciones
x0 <- 3
iter <- 21
```

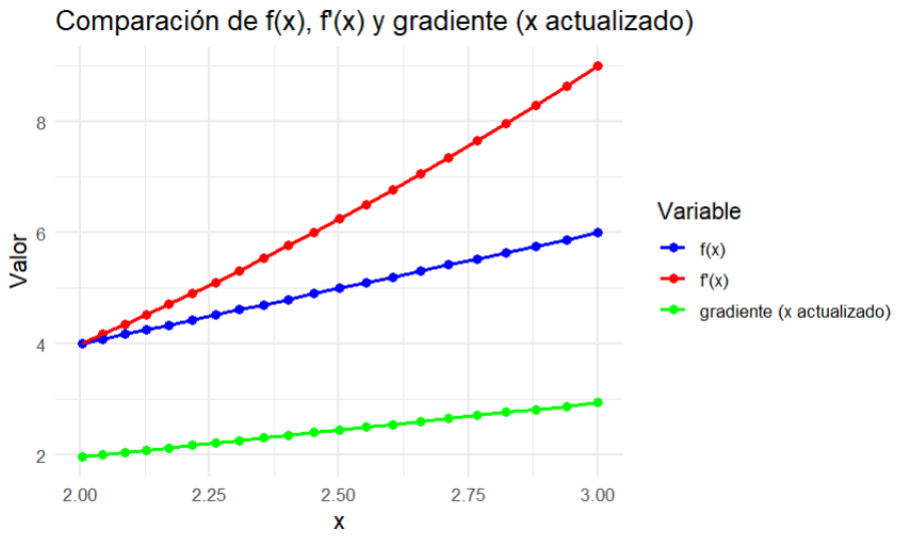
```r

# Vectores vacíos
x <- numeric(iter)
fx <- numeric(iter)
fpx <- numeric(iter)
grad <- numeric(iter)

# Primer valor
x[1] <- x0

# Bucle de cálculo
for (i in 1:iter) {
  fx[i]  <- f(x[i])
  fpx[i] <- f_deriv(x[i])
  grad[i] <- x[i] - n * fpx[i]
  if (i < iter) {
    x[i + 1] <- grad[i]
  }
}

# Crear tabla
tabla <- data.frame(
  xo  = x,
  fx  = fx,
  fpx = fpx,
  grad = grad
)
print(tabla)

# Gráfico con ggplot2
if(!require(ggplot2)) install.packages("ggplot2", repos =
    ↪ "https://cloud.r-project.org")
if(!require(tidyr))  install.packages("tidyr",  repos =
    ↪ "https://cloud.r-project.org")

library(ggplot2)
library(tidyr)

datos_long <- tabla |>
  pivot_longer(cols = c(fx, fpx, grad),
               names_to = "variable",
               values_to = "valor")

ggplot(datos_long, aes(x = xo, y = valor, color = variable)) +
  geom_point(size = 2) +
  geom_line(linewidth = 1) +
  scale_color_manual(values = c("blue", "red", "green"),
                     labels = c("f(x)", "f'(x)", "gradiente (x actualizado)")) +
  labs(title = "Comparación de f(x), f'(x) y gradiente (x actualizado)",
       x = "x",
       y = "Valor",
       color = "Variable") +
  theme_minimal(base_size = 13)
```

Ejecución

gradiente.R | datos_long | tabla

Filter

	xo	fx	fpx	grad
1	3.000000	9.000000	6.000000	2.940000
2	2.940000	8.643600	5.880000	2.881200
3	2.881200	8.301313	5.762400	2.823576
4	2.823576	7.972581	5.647152	2.767104
5	2.767104	7.656867	5.534209	2.711762
6	2.711762	7.353655	5.423525	2.657527
7	2.657527	7.062451	5.315054	2.604377
8	2.604377	6.782777	5.208753	2.552289
9	2.552289	6.514179	5.104578	2.501243
10	2.501243	6.256218	5.002487	2.451218
11	2.451218	6.008472	4.902437	2.402194
12	2.402194	5.770536	4.804388	2.354150
13	2.354150	5.542023	4.708300	2.307067
14	2.307067	5.322559	4.614134	2.260926
15	2.260926	5.111786	4.521852	2.215707
16	2.215707	4.909359	4.431415	2.171393
17	2.171393	4.714948	4.342786	2.127965
18	2.127965	4.528236	4.255931	2.085406
19	2.085406	4.348918	4.170812	2.043698
20	2.043698	4.176701	4.087396	2.002824
21	2.002824	4.011304	4.005648	1.962767



Gradiente de una función para dos variables

El **gradiente** de una función de dos variables $f(x, y)$ es un vector que apunta en la dirección del máximo crecimiento de la función. Está formado por las derivadas parciales respecto a cada variable:

$$\nabla f(x, y) = \left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y} \right)$$

En el método del **descenso del gradiente**, se busca el punto mínimo de la función actualizando los valores de x y y en la dirección opuesta al gradiente, de acuerdo con las ecuaciones:

$$\begin{cases} x_{i+1} = x_i - \eta \frac{\partial f}{\partial x}(x_i, y_i) \\ y_{i+1} = y_i - \eta \frac{\partial f}{\partial y}(x_i, y_i) \end{cases}$$

donde η es la tasa de aprendizaje que controla el tamaño del paso.

Implementación en R

En este ejemplo se aplica el método del gradiente descendente a la función:

$$f(x, y) = 3x^2y^3 + 6xy^2$$

cuyas derivadas parciales son:

$$\frac{\partial f}{\partial x} = 6xy^3 + 6y^2, \quad \frac{\partial f}{\partial y} = 9x^2y^2 + 12xy$$

A continuación, se muestra el código en R que realiza la simulación del proceso iterativo y grafica la trayectoria de los puntos sobre el mapa de calor de $f(x, y)$.

```r
# ------------------------------------------------------------
# Método del gradiente descendente para f(x,y) = 3x^2y^3 + 6xy^2
# ------------------------------------------------------------

# Definimos la función y sus derivadas parciales
f  <- function(x, y) 3*x^2*y^3 + 6*x*y^2
fx <- function(x, y) 6*x*y^3 + 6*y^2
fy <- function(x, y) 9*x^2*y^2 + 12*x*y

# Parámetros
n <- 0.01    # tasa de aprendizaje
iter <- 20   # número de iteraciones

# Valores iniciales
x <- numeric(iter)
y <- numeric(iter)
z <- numeric(iter)

x[1] <- 2
y[1] <- 1

# Iteraciones del gradiente descendente
for (i in 1:iter) {
```

```r
  z[i] <- f(x[i], y[i])  # valor de f(x,y)

  # Derivadas parciales en el punto actual
  dfx <- fx(x[i], y[i])
  dfy <- fy(x[i], y[i])

  # Actualización de variables
  if (i < iter) {
    x[i + 1] <- x[i] - n * dfx
    y[i + 1] <- y[i] - n * dfy
  }
}

# Crear tabla con resultados
tabla <- data.frame(
  iter = 1:iter,
  x = x,
  y = y,
  fxy = z
)

print(tabla)

# ------------------------------------------------------------
# Gráfico del recorrido sobre un mapa de calor de f(x,y)
# ------------------------------------------------------------

if(!require(ggplot2)) install.packages("ggplot2", repos =
    ↪ "https://cloud.r-project.org")
library(ggplot2)

# Crear grilla para graficar f(x,y)
x_vals <- seq(-2, 3, by = 0.1)
y_vals <- seq(-2, 3, by = 0.1)
grid <- expand.grid(x = x_vals, y = y_vals)
grid$f <- with(grid, f(x, y))

# Mapa de calor + trayectoria del gradiente descendente
ggplot() +
  geom_tile(data = grid, aes(x = x, y = y, fill = f)) +
  scale_fill_viridis_c(option = "plasma") +
  geom_path(data = tabla, aes(x = x, y = y), color = "red", linewidth = 1.2) +
  geom_point(data = tabla, aes(x = x, y = y), color = "white", size = 2) +
  labs(title = "Descenso del gradiente en f(x,y)",
       x = "x", y = "y", fill = "f(x,y)") +
  theme_minimal(base_size = 13)
```

Listing 1: Método del gradiente descendente para f(x,y) en R

Ejecución

gradiente3.R | tabla | grid

Filter

	iter	x	y	fxy
1	1	2.000000	1.000000000	2.400000e+01
2	2	1.820000	0.400000000	2.383181e+00
3	3	1.803411	0.264941440	9.409830e-01
4	4	1.797187	0.187059422	4.407377e-01
5	5	1.794382	0.136546147	2.253276e-01
6	6	1.792989	0.101741280	1.215156e-01
7	7	1.792255	0.076855790	6.789380e-02
8	8	1.791852	0.058618735	3.888264e-02
9	9	1.791624	0.045021475	2.266778e-02
10	10	1.791492	0.034756524	1.338917e-02
11	11	1.791415	0.026935663	7.986500e-03
12	12	1.791370	0.020935756	4.799347e-03
13	13	1.791342	0.016308727	2.900467e-03
14	14	1.791326	0.012726172	1.760531e-03
15	15	1.791316	0.009943793	1.072206e-03
16	16	1.791310	0.007777740	6.547021e-04
17	17	1.791306	0.006088389	4.005786e-04
18	18	1.791304	0.004768944	2.454799e-04
19	19	1.791303	0.003737260	1.506185e-04
20	20	1.791302	0.002929879	9.250337e-05