

# PROGRAMACIÓN NUMÉRICA

Ejercicios de trabajo encargado en clase



Cristian Ronaldo Paucar Yupanqui

ING: Fred Cruz Torres

SEMESTRE: 4to

2025

## Ejercicio 8.1: Análisis de Crecimiento de Usuarios

Datos: Meses 1..7, Usuarios (miles): 10,15,23,34,48,65,85. Paso  $h = 1$  mes.

Tasa de crecimiento (dif. centrada) en mes 4:  $f(4) = 12.50$  miles/mes.

Tasa de crecimiento (adelante) en mes 1:  $f(1) = 5$  miles/mes.

Tasa de crecimiento (atrás) en mes 7:  $f(7) = 20$  miles/mes.

Segunda derivada (meses 2..6): constante  $\approx 3$  miles/mes<sup>2</sup>.

Interpretación: La aceleración es positiva y aproximadamente constante  $\rightarrow$  crecimiento acelerado.

Fragmento de código (R) usado:

```
tasa_mes4 <- (usuarios[5] - usuarios[3]) / (2*h)
aceleracion <- (usuarios[i+1] - 2*usuarios[i] + usuarios[i-1]) / h^2
```

Tabla de usuarios:

Mes	Usuarios (miles)
1	10
2	15
3	23
4	34
5	48
6	65
7	85

# EJERCICIO 8.1: Análisis de Crecimiento de Usuarios

```
meses <- 1:7
```

```
usuarios <- c(10,15,23,34,48,65,85) # en miles
```

```
h <- 1
```

```
# Tasa de crecimiento centrada en mes 4
```

```
i <- 4
```

```
tasa_mes4 <- (usuarios[i+1] - usuarios[i-1]) / (2*h)
```

```
# Tasa adelante en mes 1
```

```
tasa_mes1 <- (usuarios[2] - usuarios[1]) / h
```

```
# Tasa atrás en mes 7
```

```
tasa_mes7 <- (usuarios[7] - usuarios[6]) / h
```

```
# Segunda derivada (aceleración) para meses 2..6
```

```
aceleracion <- numeric(5)
```

```
for (j in 2:6) {
```

```
  aceleracion[j-1] <- (usuarios[j+1] - 2*usuarios[j] + usuarios[j-1]) / (h^2)
```

```
}
```

```
# Resultados
```

```
print(data.frame(Mes=meses, Usuarios=usuarios))
```

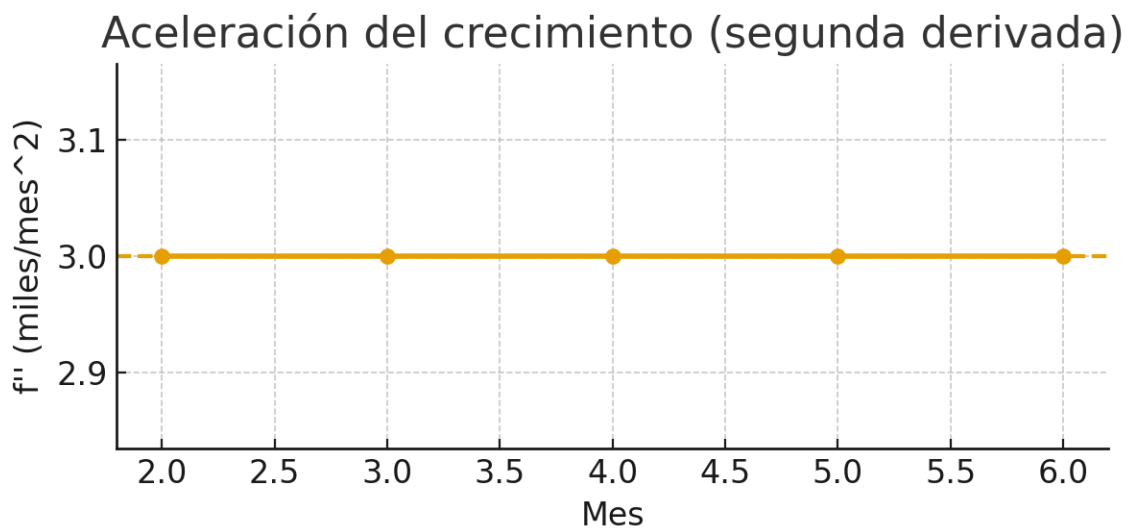
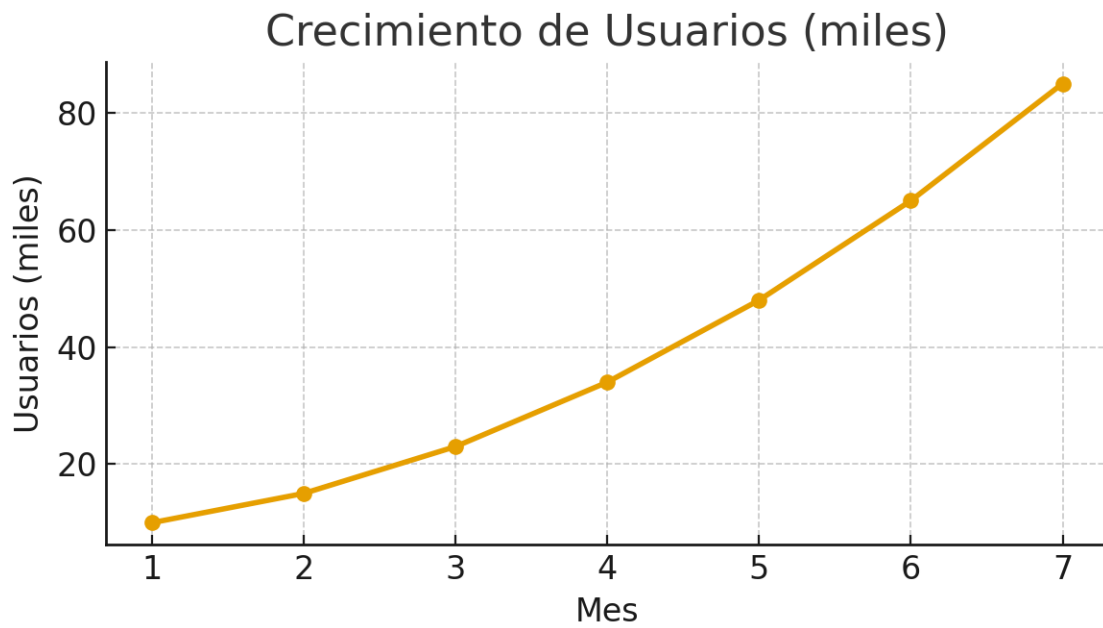
```
print(paste("f(4) =", tasa_mes4, "miles/mes"))
```

```
print(paste("f(1) =", tasa_mes1, "miles/mes"))
```

```
print(paste("f(7) =", tasa_mes7, "miles/mes"))
```

```
print("Segunda derivada:"); print(aceleracion)
```

main='Crecimiento de Usuarios')



## Ejercicio 8.2: Optimización de la función de pérdida

Datos: Épocas: 0,10,20,30,40,50. Loss: 2.45,1.82,1.35,1.08,0.95,0.89. Paso  $h = 10$ .

Tasa de cambio en época 20 (dif. centrada):  $L'(20) = -0.0370$  unidades de loss/época.

Segunda derivada en época 30:  $L''(30) = 0.001400 > 0 \rightarrow$  desaceleración en la disminución del loss (convergencia).

Criterio  $|\Delta L| < 0.01$  se cumple a partir de la época 50.

Estimación  $L(25)$  por linealización alrededor de 20:  $L(25) \approx 1.1650$ .

Fragmento de código (R) usado:

```
tasa_loss_20 <- (loss[4] - loss[2]) / (2*h)
segunda_deriv_30 <- (loss[5] - 2*loss[4] + loss[3]) / h^2
```

Tabla de intervalos  $\Delta \text{Loss}$  por época:

Intervalo	$\Delta \text{Loss/época}$	Magnitud
0-10	-0.0630	0.0630
10-20	-0.0470	0.0470
20-30	-0.0270	0.0270
30-40	-0.0130	0.0130
40-50	-0.0060	0.0060

# EJERCICIO 8.2: Optimización de la función de pérdida

```
epocas <- c(0,10,20,30,40,50)
```

```
loss <- c(2.45,1.82,1.35,1.08,0.95,0.89)
```

```
h <- 10
```

```
# Tasa centrada en época 20
```

```
i <- 3
```

```
tasa_loss_20 <- (loss[i+1] - loss[i-1]) / (2*h)
```

```
# Segunda derivada en época 30
```

```
i30 <- 4
```

```
segunda_loss_30 <- (loss[i30+1] - 2*loss[i30] + loss[i30-1]) / (h^2)
```

```

# Intervalos  $\Delta\text{Loss}/\text{época}$ 

deltas <- (loss[-1] - loss[-length(loss)]) / h

intervalos <- data.frame(
  Intervalo = paste(epocas[-length(epocas)], epocas[-1], sep='-'),
  Delta_loss_por_epoca = deltas,
  Magnitud = abs(deltas)
)

# Época donde  $|\Delta L| < 0.01$ 

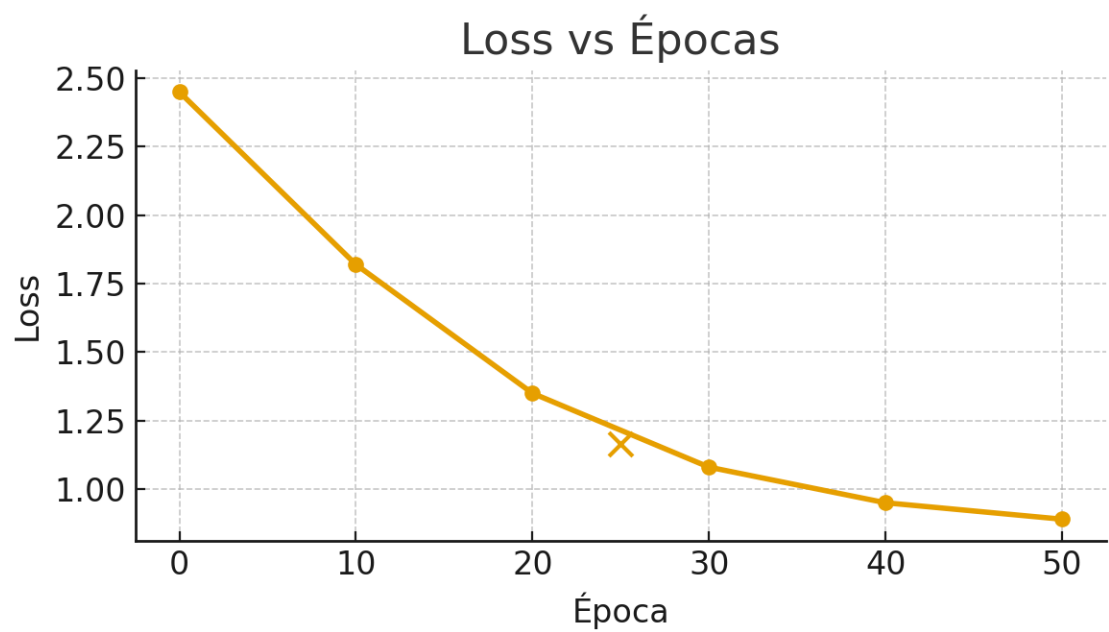
epoca_parada_idx <- which(intervalos$Magnitud < 0.01)[1]

if (!is.na(epoca_parada_idx)) {
  epoca_parada <- epocas[epoca_parada_idx + 1]
} else {
  epoca_parada <- NA
}

# Estimación  $L(25)$ 

loss_25 <- loss[3] + tasa_loss_20 * (25 - 20)

```



### Ejercicio 8.3: Análisis de series temporales de ventas

Datos: Ventas diarias (\$k) para Lun..Dom: 45,52,61,58,73,89,95.  $h = 1$  día.

Velocidad ( $f'$ ) por día (adelante, centrada, atrás):

Día	Velocidad (\$k/día)
Lun	7.0
Mar	8.0
Mie	3.0
Jue	6.0
Vie	15.5
Sab	11.0
Dom	6.0

Día con mayor aceleración: Jue ( $f'' = 18$ ).

Fragmento de código (R) usado:

```
# EJERCICIO 8.3: Análisis de Series Temporales de Ventas
```

```
dias <- 1:7
```

```
ventas <- c(45,52,61,58,73,89,95)
```

```
h <- 1
```

```
n <- length(ventas)
```

```
nombres_dias <- c('Lun','Mar','Mie','Jue','Vie','Sab','Dom')
```

```
# Derivadas
```

```
f1_adelante <- (ventas[2] - ventas[1]) / h
```

```
f1_central <- numeric(n-2)
```

```
for (i in 2:(n-1)) {
```

```
  f1_central[i-1] <- (ventas[i+1] - ventas[i-1]) / (2*h)
```

```
}
```

```
f1_atras <- (ventas[n] - ventas[n-1]) / h
```

```
velocidad <- c(f1_adelante, f1_central, f1_atras)
```

```
# Segunda derivada
```



```

aceleracion <- numeric(n-2)

for (i in 2:(n-1)) {
  aceleracion[i-1] <- (ventas[i+1] - 2*ventas[i] + ventas[i-1]) / (h^2)
}

# Día con mayor aceleración
max_acel_idx <- which.max(aceleracion) + 1
dia_max_acel <- nombres_dias[max_acel_idx]
print(paste("Día con mayor aceleración:", dia_max_acel))

```



## Ejercicio 8.4: Gradiente de función de activación (sigmoide)

Datos:  $x = -3, -2, -1, 0, 1, 2, 3$ .  $\sigma(x)$  given. Paso  $h = 1$  (grande).

Derivada numérica centrada en 0:  $\sigma'(0) \approx 0.23110$  (analítica = 0.25000).

Derivadas numéricas en -2 y 2: 0.11075, 0.11075 (analítica  $\approx 0.10499$ , 0.10499).

Conclusión:  $h=1$  es demasiado grande; usar  $h \approx 0.01$  o 0.001 para mayor precisión.

Tabla comparativa ( $x$ ,  $\sigma(x)$ , numérico, analítico):

$x$	$\sigma(x)$	Numérico (aprox)	Analítico
-3.0	0.0474		0.04515
-2.0	0.1192	0.11075	0.10499
-1.0	0.2689		0.19659
0.0	0.5000	0.23110	0.25000
1.0	0.7311		0.19659
2.0	0.8808	0.11075	0.10499
3.0	0.9526		0.04515

```
# EJERCICIO 8.4: Gradiente de función sigmoide
```

```
x <- c(-3,-2,-1,0,1,2,3)
```

```
sigma_x <- c(0.0474,0.1192,0.2689,0.5,0.7311,0.8808,0.9526)
```

```
h <- 1
```

```
# Derivada numérica centrada
```

```
grad_num <- rep(NA, length(x))
```

```
for (i in 2:(length(x)-1)) {
```

```
  grad_num[i] <- (sigma_x[i+1] - sigma_x[i-1]) / (2*h)
```

```
}
```

```
# Derivada analítica
```

```
grad_analit <- sigma_x * (1 - sigma_x)
```

```
data.frame(x=x, sigma=sigma_x, Numerico=grad_num, Analitico=grad_analit)
```

## Ejercicio 8.5: Detección de anomalías en métricas de sistema (latencia)

Datos: Horas 0..7, Latencias (ms): 120,125,128,135,280,290,275,155.  $h = 1$ .

Tasa de cambio ( $f'$ ) por hora:

Hora	Tasa (ms/hora)
0	5.0
1	4.0
2	5.0
3	76.0
4	77.5
5	-2.5
6	-67.5
7	-120.0

Pico (cambio sign de  $f'$  de + a -) ocurre en la hora: 4.

Magnitud del salto (hora 3->4) usando derivada hacia adelante: 145 ms/hora.

Momentos de anomalía ( $|f'| > 50$  ms/hora): 3, 4, 6, 7

Código r:

```
# EJERCICIO 8.5: Detección de anomalías en latencia
```

```
horas <- 0:7
```

```
latencia <- c(120,125,128,135,280,290,275,155)
```

```
h <- 1
```

```
n <- length(latencia)
```

```
# Tasa de cambio
```

```
f1_adelante <- (latencia[2] - latencia[1]) / h
```

```
f1_central <- numeric(n-2)
```

```
for (i in 2:(n-1)) {
```

```
  f1_central[i-1] <- (latencia[i+1] - latencia[i-1]) / (2*h)
```

```
}
```

```
f1_atras <- (latencia[n] - latencia[n-1]) / h
```

```
velocidad <- c(f1_adelante, f1_central, f1_atras)
```

```
# Aceleración
```

```
aceleracion <- numeric(n-2)
```

```
for (i in 2:(n-1)) {
```

```
  aceleracion[i-1] <- (latencia[i+1] - 2*latencia[i] + latencia[i-1]) / (h^2)
```

```
}
```

```
# Pico de anomalía
```

```
pico_hora <- NA
```

```
for (i in 1:(length(aceleracion)-1)) {
```

```
  if (aceleracion[i] > 0 && aceleracion[i+1] < 0) {
```

```
    pico_hora <- horas[i+2]
```

```
  }
```

```
}
```

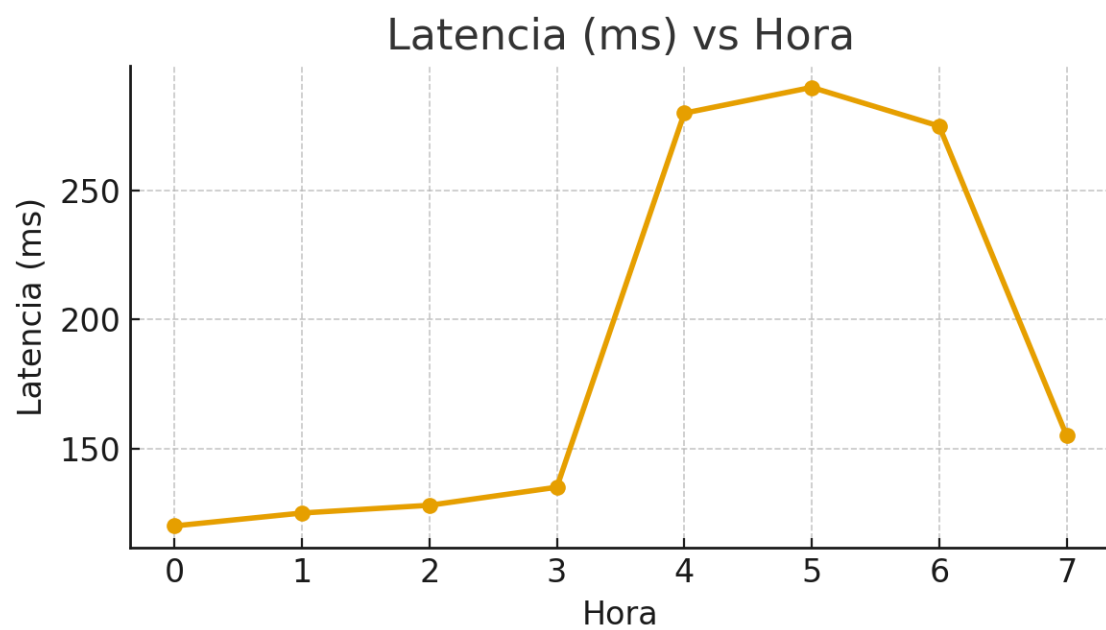
```
# Momentos con  $|f'| > 50$ 
```

```
horas_anom <- horas[abs(velocidad) > 50]
```

```
list(velocidad=velocidad, aceleracion=aceleracion,
```

```
     pico_hora=pico_hora, horas_anom=horas_anom)
```

Gráfica de latencia:



## Ejercicio 8.6: Análisis de tasa de conversión

Datos: Gasto (\$k): 0,5,10,15,20,25. Conversión (%): 2.1,3.8,5.2,6.1,6.7,7.0.  $h = 5$ .

ROI marginal en cada punto (aprox):

Gasto (\$k)	ROI marginal (% per k)
0	0.340
5	0.310
10	0.230
15	0.150
20	0.090
25	0.060

Gastos con ROI marginal  $> 0.2$ : 0, 5, 10

Segunda derivada en \$15k: -0.0120 (negativa  $\rightarrow$  rendimientos decrecientes).

Recomendación: No aumentar gasto más allá de \$25k por ROI marginal muy bajo.

# EJERCICIO 8.6: Análisis de tasa de conversión

```
gasto <- c(0,5,10,15,20,25)
```

```
conversion <- c(2.1,3.8,5.2,6.1,6.7,7.0)
```

```
h <- 5
```

```
n <- length(gasto)
```

```
# ROI marginal
```

```
f1_adel <- (conversion[2] - conversion[1]) / h
```

```
f1_central <- numeric(n-2)
```

```
for (i in 2:(n-1)) {
```

```
  f1_central[i-1] <- (conversion[i+1] - conversion[i-1]) / (2*h)
```

```
}
```

```
f1_atras <- (conversion[n] - conversion[n-1]) / h
```

```
roi_marginal <- c(f1_adel, f1_central, f1_atras)
```

```
# Rango con ROI marginal  $> 0.2$ 
```

```
gasto_ideal <- gasto[roi_marginal > 0.2]
```

```
# Segunda derivada en 15k
```

```
i15 <- which(gasto==15)
```

```
fpp_15k <- (conversion[i15+1] - 2*conversion[i15] + conversion[i15-1]) / (h^2)
```

```
list(roi_marginal=roi_marginal, gasto_ideal=gasto_ideal, fpp_15k=fpp_15k)
```

## Ejercicio 8.7: Feature Engineering con derivadas

Datos: Tiempo (s) 0..7, Temp (°C): 20.1,20.3,20.8,21.5,22.6,24.2,26.1,28.5.  $h = 1$ .

Se crearon features: Velocidad (f') y Aceleración (f''). Normalización Min-Max aplicada.

Momentos de alerta (Velocidad > 0.8 °C/s): 3, 4, 5, 6, 7

Tabla de temperaturas y features (resumen):

Tiempo (s)	Temp (°C)	Velocidad (°C/s)	Aceleración (°C/s²)
0	20.1	0.20	
1	20.3	0.35	0.30
2	20.8	0.60	0.20
3	21.5	0.90	0.40
4	22.6	1.35	0.50
5	24.2	1.75	0.30
6	26.1	2.15	0.50
7	28.5	2.40	

Codigo r:

```
# EJERCICIO 8.7: Feature Engineering con derivadas
```

```
tiempo <- 0:7
```

```
temp <- c(20.1,20.3,20.8,21.5,22.6,24.2,26.1,28.5)
```

```
h <- 1
```

```
n <- length(temp)
```

```
# Velocidad (1ª derivada)
```

```
velocidad <- numeric(n)
```

```
velocidad[1] <- (temp[2] - temp[1]) / h
```

```
for (i in 2:(n-1)) {
```

```
  velocidad[i] <- (temp[i+1] - temp[i-1]) / (2*h)
```

```
}
```

```
velocidad[n] <- (temp[n] - temp[n-1]) / h
```



```
# Aceleración (2ª derivada)

aceleracion <- rep(NA, n)

for (i in 2:(n-1)) {
  aceleracion[i] <- (temp[i+1] - 2*temp[i] + temp[i-1]) / (h^2)
}
```

```
# Detección de picos (alerta)

umbral_alerta <- 0.8

alerta <- velocidad > umbral_alerta

momentos_alerta <- tiempo[alerta]
```

```
# Resultados

data.frame(Tiempo=tiempo, Temperatura=temp,
  Velocidad=round(velocidad,3),
  Aceleracion=round(aceleracion,3),
  Alerta=alerta)
```

Gráfica de temperatura:

Temperatura vs Tiempo

