MA0501 – Tarea 5

Diego Alberto Vega Víquez - C38367 José Carlos Quintero Cedeño - C26152
Gabriel Valverde Guzmán - C38060

2025-10-20

Tabla de contenidos

Ejercicio 1

Instrucción del ejercicio 1

Implemente en **R** los algoritmos de derivación numérica vistos en clase.

Se deben implementar los siguientes:

- `Der2Puntos(X0, h)`

  (* Calcula la derivada de F usando la fórmula de 2 puntos *)
- `Der3PuntosA(X0, h)`

  (* Calcula la derivada de F usando la fórmula de 3 puntos - fórmula A *)
- `Der3PuntosB(X0, h)`

  (* Calcula la derivada de F usando la fórmula de 3 puntos - fórmula B *)
- `Der5PuntosA(X0, h)`

  (* Calcula la derivada de F usando la fórmula de 5 puntos - fórmula A *)
- `Der5PuntosB(X0, h)`

  (* Calcula la derivada de F usando la fórmula de 5 puntos - fórmula B *)

Solución

```r
Der2Puntos <- function(f, x_0, h) {
  return((f(x_0 + h) - f(x_0)) / h)
}

Der3PuntosA <- function(f, x_0, h) {
  return(((-3 / 2) * f(x_0) + 2 * f(x_0 + h) - (1 / 2) * f(x_0 + 2 * h)) /
           h)
}

Der3PuntosB <- function(f, x_0, h){
  return(((-1/2)*f(x_0 - h) + (1/2)*f(x_0 + h))/h)
}

Der5PuntosA <- function(f, x_0, h){
  return(( (-25/12)*f(x_0) + 4*f(x_0 + h) - 3*f(x_0 + 2*h) +
             (4/3)*f(x_0 + 3*h) - (1/4)*f(x_0 + 4*h) ) / h)
}

Der5PuntosB <- function(f, x_0, h){
```

```r
  return(( f(x_0 - 2*h) - 8*f(x_0 - h) + 8*f(x_0 + h) - f(x_0 + 2*h) ) / (12*h))
}
```

Ejercicio 2

> **i** Instrucción del ejercicio 2
>
> Sea $f(x) = x^3 e^{x^2} - \sin x$.
> Para $h = 0.01$ y $h = 0.001$, aproxime $f'(2.19)$ usando todos los métodos implementados en el ejercicio 1.
> Compare con el valor exacto.
> ¿Cuál es la mejor aproximación?

Solución

```r
f <- function(x) (x^3)*exp(x^2) - sin(x)
df <- function(x) 3*(x^2)*exp(x^2) + 2*(x^4)*exp(x^2)-cos(x)
x <- 2.19
h1 <- 0.01
h2 <- 0.001

print("f'(2.19) con dos puntos")
```

```
[1] "f'(2.19) con dos puntos"
```

```r
Der2Puntos(f, x, h1)
```

```
[1] 7533.961
```

```r
Der2Puntos(f, x, h2)
```

```
[1] 7332.385
```

```r
df(x)
```

```
[1] 7310.45
```

```r
print("f'(2.19) con tres puntos")
```

```
[1] "f'(2.19) con tres puntos"
```

```r
Der3PuntosA(f, x, h1)
```

```
[1] 7300.911
```

```r
Der3PuntosA(f, x, h2)
```

```
[1] 7310.358
```

```r
Der3PuntosB(f, x, h1)
```

```
[1] 7314.993
```

```r
Der3PuntosB(f, x, h2)
```

```
[1] 7310.495
```

```r
print("f'(2.19) con 5 puntos")
```

```
[1] "f'(2.19) con 5 puntos"
```

```r
Der5PuntosA(f, x, h1)
```

```
[1] 7310.423
```

```r
Der5PuntosA(f, x, h2)
```

```
[1] 7310.45
```

```r
Der5PuntosB(f, x, h1)
```

```
[1] 7310.446
```

```r
Der5PuntosB(f, x, h2)
```

```
[1] 7310.45
```

Es claro que entre más pequeño el valor de h escogido, mejor va a ser la aproximación. Lo mismo sucede con la cantidad de nodos usados para estimar la derivada en el punto.

Ejercicio 3

> **i** Instrucción del ejercicio 3
>
> Agregue al archivo anterior una función para la **Extrapolación de Richardson**, que permita trabajar con fórmulas del tipo $N(x_0, h)$.

Solución

```r
Richardson <- function(N, x0, h, n){
  Q <- matrix(NA_real_, nrow = n, ncol = n)
```

```r
  for (i in 1:n) {
    Q[i,1] <- N(x0, h/(2^(i - 1)))
  }
  for (i in 2:n) {
    for (j in 2:i) {
      Q[i,j] <- (4^(j-1)*Q[i,j-1] - Q[i-1,j-1])/(4^(j-1)-1)
    }
  }
  return(Q[n,n])
}
```

Ejercicio 4

Instrucción del ejercicio 4

Se puede probar la siguiente **fórmula progresiva** para calcular derivadas:

$$f'(x_0) = \frac{1}{h}[f(x_0+h) - f(x_0)] - \frac{h}{2}f''(x_0) - \frac{h^2}{2}f'''(x_0) + O(h^3)$$

Aplique **extrapolación de Richardson** para aproximar derivadas con la fórmula anterior. Luego, repita el ejercicio 2 usando este método.

Solución

```r
formula <- function(f, d2f, d3f, x0, h){
  return((f(x0+h)-f(x0))/h - (h/2)*d2f(x0) - (h^2/2)*d3f(x0))
}

f <- function(x) (x^3)*exp(x^2) - sin(x)
df <- function(x) 3*(x^2)*exp(x^2) + 2*(x^4)*exp(x^2)-cos(x)
d2f <- function(x) 6*x*exp(x^2) + 14*x^3*exp(x^2)+4*x^5*exp(x^2)+sin(x)
d3f <- function(x) 6*exp(x^2)+54*x^2*exp(x^2)+48*x^4*exp(x^2)+8*x^6*exp(x^2)+cos(x)

Richardson(function(x, h) formula(f, d2f, d3f, x, h), 2.19, 0.01, 2)
```

```
[1] 7310.437
```

```r
Richardson(function(x, h) formula(f, d2f, d3f, x, h), 2.19, 0.001, 2)
```

```
[1] 7310.45
```

Ejercicio 5

> Instrucción del ejercicio 5
>
> Implemente en **R** los algoritmos de **integración numérica** vistos en clase.
>
> Se deben implementar los siguientes:
>
> - `Trapecio(a, b)`
>   (* Calcula la integral de F de a hasta b, usando la regla del Trapecio *)
> - `Simpson(a, b)`
>   (* Calcula la integral de F de a hasta b, usando la regla de Simpson *)
> - `SimpsonCompuesta(a, b, m)`
>   (* Calcula la integral de F de a hasta b, usando la regla de Simpson Compuesta *)
> - `Romberg(a, b, n)`
>   (* Calcula la integral de F de a hasta b, usando el método de Romberg *)
> - `IntegralDobleSCT(a, b, n, m)`
>   (* Calcula la integral doble de F en regiones de tipo I usando Simpson Compuesto *)

Solución

```r
Trapecio <- function(f, a, b){
  return((b-a)*(f(a) + f(b))/2)
}

Simpson <- function(f, a, b){
  return(((b-a)/6)*(f(a) + 4*f((a+b)/2)+f(b)))
}

SimpsonCompuesto <- function(f, a, b, m){
  h <- (b-a)/(2 * m)
  xi0 <- f(a)+f(b)
  xi1 <- 0 #suma de nodos impares
  xi2 <- 0 #suma de nodos pares
  for(j in 1:(2 * m - 1)){
    x <- a+j*h
    if(j%%2 == 0){
      xi2 <- xi2 + f(x)
    } else{
      xi1 <- xi1 + f(x)
```

```r
    }
  }
  return((h/3)*(xi0 + 2*xi2 + 4*xi1))
}

Romberg <- function(f, a, b, n){
  Q <- matrix(NA_real_, nrow = n, ncol = n)
  h <- b-a
  Q[1,1] <- (h/2)*(f(a) + f(b))
  for (i in 2:n){
    suma <- 0
    for (k in 1:(2^(i-2))){
      suma <- suma + f(a + (k - 1/2)*h)
    }
    Q[i,1] <- 0.5*(Q[i-1,1] + h*suma)
    h <- h/2
  }

  for (i in 2:n) {
    for (j in 2:i) {
      Q[i,j] <- (4^(j-1)*Q[i,j-1] - Q[i-1,j-1])/(4^(j-1)-1)
    }
  }
  return(Q[n,n])
}

IntegralDobleSCT <- function(f, g1, g2, a, b, n, m){
  h <- (b-a)/(2*n)
  xi0 <- 0 # inicial + final
  xi1 <- 0 # impares
  xi2 <- 0 # pares
  for (i in 0:(2*n)) {
    xw <- a + i*h
    yi <- SimpsonCompuesto(function(y) f(xw, y), g1(xw), g2(xw), m)
    if(i == 0 || i == 2*n){
      xi0 <- xi0 + yi
```

```r
    } else if(i %% 2 == 0){
      xi2 <- xi2 + yi
    } else {
      xi1 <- xi1 + yi
    }
  }
  return((h/3)*(xi0+4*xi1+2*xi2))
}
```

Ejercicio 6

Instrucción del ejercicio 6

Calcule

$$\int_0^\pi (4 + 2\sin x)\, dx$$

mediante **todos los métodos programados arriba** (*excepto el último método de integración doble*).

Solución

```r
f <- function(x){4 + 2*sin(x)}


Trapecio(f, 0, pi)
```

```
[1] 12.56637
```

```r
Simpson(f, 0, pi)
```

```
[1] 16.75516
```

```r
SimpsonCompuesto(f, 0, pi, 3)
```

```
[1] 16.5681
```

```r
Romberg(f, 0, pi, 3)
```

```
[1] 16.56351
```

Ejercicio 7

> **i** Instrucción del ejercicio 7
>
> Determine el valor de n según las **cotas teóricas del error** para aproximar la integral:
>
> $$\int_0^{\pi} (8 + 5\sin x)\, dx$$
>
> con **4 cifras significativas** mediante la **regla de Simpson Compuesta**.
> Calcule la integral con este valor de n.
> ¿Es realmente el valor de n necesario?

Solución

Note que el error del metodo de simpson compuesto viene dado por:

$$|\frac{(b-a)^5 \cdot f^{(4)}(\mu)}{2880 \cdot n^4}|$$

De manera que si queremos acotarlo por 10^{-4}, entonces hay que despejarlo tomando en cuenta lo siguiente:

Note que $f^{(4)} = 5 \cdot \sin(x) \leq 5$ y ademas $(b-a)^5 = \pi^5$

$$\Rightarrow |\frac{(b-a)^5 \cdot f^{(4)}(\mu)}{2880 \cdot n^4}| \leq |\frac{\pi^5 \cdot 5}{2880 \cdot n^4}| \leq 10^{-4} \Rightarrow n \geq (\frac{\pi^5 \cdot 5}{2880 \cdot 10^{-4}})^{1/4} \approx 8.5375 \Rightarrow n \geq 9$$

Así pues, teóricamente, se requiere que n sea mayor o igual a 9.

Por otra parte, note que con n = 8 se consigue que el error sea menor a 10^{-4}:

```r
integral_real <- integrate(function(x) 8 + 5*sin(x), 0, pi)$value
integral_estimada <- SimpsonCompuesto(function(x) 8 + 5*sin(x), 0, pi, 8)
abs(integral_real- integral_estimada)
```

```
[1] 8.295524e-05
```

Esto no ocurre si n = 7:

```r
integral_real <- integrate(function(x) 8 + 5*sin(x), 0, pi)$value
integral_estimada <- SimpsonCompuesto(function(x) 8 + 5*sin(x), 0, pi, 7)
abs(integral_real- integral_estimada)
```

```
[1] 0.0001417178
```

Ejercicio 8

Instrucción del ejercicio 8

Calcule ln(2) usando:

$$\ln(2) = \int_1^2 \frac{dx}{x}$$

mediante la **regla de Simpson Compuesta**.

¿Cuál es el menor valor de n que garantiza **6 cifras significativas** en el cálculo de ln(2)?

Regla de Simpson compuesta

Sea $f(x) = 1/x$, $[a, b] = [1, 2]$, n par y $h = (b - a)/n = 1/n$. La fórmula es

$$S_n = \frac{h}{3}\Big[f(a) + f(b) + 4\sum_{\substack{j=1 \\ j \text{ impar}}}^{n-1} f(a + jh) + 2\sum_{\substack{j=2 \\ j \text{ par}}}^{n-2} f(a + jh)\Big].$$

Cota de error

Para Simpson compuesta el error satisface

$$|E_n| = \left| -\frac{(b-a)}{180}\, h^4\, f^{(4)}(\xi) \right| \le \frac{(b-a)}{180}\, h^4 \max_{[a,b]} |f^{(4)}|.$$

Con $f(x) = x^{-1}$, $f^{(4)}(x) = 24\, x^{-5}$, decreciente en $[1, 2]$, por lo que

$$\max_{[1,2]} |f^{(4)}| = 24 \quad (\text{en } x = 1).$$

Además $b - a = 1$, luego

$$|E_n| \le \frac{1}{180} \cdot 24\, h^4 = \frac{2}{15}\, h^4.$$

Para **6 cifras significativas** pedimos un error absoluto $\le 0.5 \times 10^{-6}$:

$$\frac{2}{15} h^4 \le 0.5 \times 10^{-6} \implies h \le (3.75 \times 10^{-6})^{1/4} \approx 0.04401.$$

Como $h = 1/n$, obtenemos $n \ge 22.73$. El menor n **par** que cumple esto es

$$\boxed{n = 24}.$$

Valor numérico (R)

```r
# Definimos la función
f <- function(x) 1/x

# Regla de Simpson compuesta
simpson_compuesta <- function(f, a, b, n) {
  stopifnot(n %% 2 == 0)
  h <- (b - a) / n
  x <- seq(a, b, by = h)
  fx <- f(x)
  S <- fx[1] + fx[n + 1] +
       4 * sum(fx[seq(2, n, by = 2)]) +
       2 * sum(fx[seq(3, n - 1, by = 2)])
  (h / 3) * S
}

# Ejemplo: cálculo con n = 24
n <- 24
S24 <- simpson_compuesta(f, 1, 2, n)
err <- abs(S24 - log(2))

c(n = n,
  aproximacion = S24,
  ln2 = log(2),
  error_abs = err)
```

```
           n aproximacion          ln2    error_abs
2.400000e+01 6.931473e-01 6.931472e-01 9.378437e-08
```

El menor número de subintervalos que garantiza 6 cifras significativas es

$$\boxed{n = 24}.$$

Con $n = 24$ se obtiene un error absoluto $< 0.5 \times 10^{-6}$, cumpliendo la exigencia.

Ejercicio 9

> **Instrucción del ejercicio 9**
>
> Use el procedimiento de **integrales dobles** para calcular el área limitada por la parábola $y = 4x - x^2$, el eje X, y la recta $y = -3x + 6$.
>
> Resuélvalo también usando **R** directamente.

Solución

(Determinación de los puntos de intersección, límites de integración y cálculo del área mediante integración doble y verificación en R.)

Definimos el Área a calcular como $A = \int\int_R 1dA$. Note que:

- La parábola $y_1 = 4x - x^2$ interseca al eje x en $(0, 0)$ y $(4, 0)$.
- La recta $y_2 = -3x + 6$ interseca al eje x en $(2, 0)$.
- Las ecuaciones se intersecan entre sí en $x = 1$ y $x = 6$.
- En $[0, 1]$ la región tope está dada por y_1 mientras que en $[1, 2]$ la región tope la define y_2.

Calculando A:

$$A = \int_0^1 \int_0^{4x-x^2} dydx + \int_1^2 \int_0^{-3x+6} dydx$$

$$A = \int_0^1 (4x - x^2)dx + \int_1^2 (-3x + 6)dx$$

$$A = \left(2x^2 - \frac{x^3}{3}\right)\Bigg|_{x=0}^{1} + \left(\frac{-3x^2}{2} + 6x\right)\Bigg|_{x=1}^{2} = \frac{19}{6} \approx 3.1\bar{6}$$

Ahora, en R:

```r
y1 <- function(x) 4*x - x^2
y2 <- function(x) -3*x + 6

A1 <- integrate(y1, lower = 0, upper = 1)$value
A2 <- integrate(y2, lower = 1, upper = 2)$value

A  <- A1 + A2
A
```

```
[1] 3.166667
```

Ejercicio 10

Instrucción del ejercicio 10

Usando el procedimiento de integración múltiple, calcule el volumen del sólido limitado por el paraboloide

$$z = 4 - x^2 - 2y^2$$

y el plano xy.

Resuélvalo también usando **R** directamente.

Solución

calculo con la funcion IntegralDobleSCT

```
root_safe <- function(x) pmax(0, 32 - 7*x^2)

volumen_estimado <- IntegralDobleSCT(function(x, y) 4 - x^2 - 2*y^2 - x*y, function(x) { (-x -
volumen_estimado
```

```
[1] 18.99858
```

Calculo directo con R

```
## Integrando: (4 - x^2 - 2y^2 - x*y)
f <- function(x, y) 4 - x^2 - 2*y^2 - x*y

## Límites en x como funciones de y
x_lower <- function(y) (-y - sqrt(16 - 7*y^2)) / 2
x_upper <- function(y) (-y + sqrt(16 - 7*y^2)) / 2

## Borde en y
y_min <- -4 / sqrt(7)
y_max <-  4 / sqrt(7)

## Integral interior en x, para un y dado
inner_x <- function(y) {
  ## Si y está fuera del dominio por redondeo numérico, devuelve 0
  if (7*y^2 > 16) return(0)
  integrate(function(x) f(x, y),
```

```
            lower = x_lower(y), upper = x_upper(y),
            rel.tol = 1e-10, abs.tol = 0)$value
}

## Integral exterior en y
V_dxdy <- integrate(function(y) Vectorize(inner_x)(y),
                    lower = y_min, upper = y_max,
                    rel.tol = 1e-9, abs.tol = 0)

V_dxdy$value
```

```
[1] 18.99857
```

Ejercicio 11

> **i** Instrucción del ejercicio 11
>
> La **Regla del Trapecio Extendida** se presenta en el siguiente teorema:
>
> **Teorema:**
> Sea $f \in C^2[a,b]$, con $h = (b-a)/n$ y $x_j = a + jh$ para cada $j = 0, 1, \dots, n$.
> La regla del trapecio para n subintervalos es:
>
> $$\int_a^b f(x)\,dx = \frac{h}{2}\left[f(a) + f(b) + 2\sum_{j=1}^{n-1} f(x_j)\right] - \frac{(b-a)h^2}{12} f''(\mu), \quad \text{para alguna } \mu \in [a,b].$$
>
> (a) Pruebe el teorema anterior.
>
> (b) Escriba un algoritmo en pseudo-código para la **regla del trapecio extendida**.
>
> (c) Implemente en **R** la regla del trapecio extendida.

Solución

(a) Sea $f \in C^2[a,b]$ y subdividamos el intervalo en n subintervalos de longitud $h = (b-a)/n$.

En cada subintervalo $[x_j, x_{j+1}]$, aplicamos la regla del trapecio simple:

$$\int_{x_j}^{x_{j+1}} f(x)\,dx = \frac{h}{2}\left[f(x_j) + f(x_{j+1})\right] - \frac{h^3}{12} f''(\xi_j), \quad \xi_j \in [x_j, x_{j+1}].$$

Sumando para $j = 0, \dots, n-1$:

$$\int_a^b f(x)\,dx = \sum_{j=0}^{n-1} \int_{x_j}^{x_{j+1}} f(x)\,dx = \frac{h}{2}\Big[f(a) + f(b) + 2\sum_{j=1}^{n-1} f(x_j)\Big] - \frac{h^3}{12}\sum_{j=0}^{n-1} f''(\xi_j).$$

Por el **teorema del valor medio para integrales**, existe $\mu \in [a, b]$ tal que:

$$\sum_{j=0}^{n-1} f''(\xi_j) = n f''(\mu).$$

Sustituyendo $n = \frac{b-a}{h}$, obtenemos el término de error global:

$$E(f) = -\frac{(b-a)h^2}{12} f''(\mu),$$

■

(b) **Pseudo-código de la Regla del Trapecio Extendida**

```
Algoritmo Trapecio_Extendido(f, a, b, n)
  # Calcula _a^b f(x) dx usando la regla del trapecio extendida
  Entrada:
      f(x): función continua
      a, b: límites de integración
      n: número de subintervalos
  Salida:
      Aproximación de la integral

  h ← (b - a) / n
  suma ← f(a) + f(b)

  Para j desde 1 hasta n-1 hacer
      xj ← a + j * h
      suma ← suma + 2 * f(xj)
  Fin Para

  integral ← (h / 2) * suma
  Retornar integral
Fin Algoritmo
```

(c) **Regla del trapecio extendida en R**

```
TrapecioExtendido <- function(f, a, b, n) {
  h <- (b - a) / n
  suma <- f(a) + f(b)

  for (j in 1:(n - 1)) {
    xj <- a + j * h
    suma <- suma + 2 * f(xj)
  }

  I <- (h / 2) * suma
  return(I)
}
```

Ejercicio 12

> **Instrucción del ejercicio 12**
>
> Calcule:
>
> $$\int_0^2 x^2 e^{-x^2}\, dx$$
>
> usando la **regla del trapecio extendida** con $n = 8$.

Solución

(Cálculo numérico de la integral y comparación con el valor de referencia obtenido por software simbólico.)

```
f <- function(x) x^2*exp(-x^2)

t.e <- TrapecioExtendido(f,0,2,8)

s.s <- integrate(f,0,2)$value

print("Aproximación por regla del trapecio extendida:" )
```

```
[1] "Aproximación por regla del trapecio extendida:"
```

```r
t.e
```

```
[1] 0.421582
```

```r
print("Integral según método de R:" )
```

```
[1] "Integral según método de R:"
```

```r
s.s
```

```
[1] 0.4227251
```

```r
print("Error de aproximación:" )
```

```
[1] "Error de aproximación:"
```

```r
s.s - t.e
```

```
[1] 0.001143019
```

Ejercicio 13

Instrucción del ejercicio 13

Sea f definida por:

$$f(x) = \begin{cases} x^3 + 1, & 0 \leq x \leq 2, \\ 2x + 5, & 2 < x \leq 6. \end{cases}$$

(a) Aproxime $\int_0^6 f(x)\,dx$ usando la **regla del trapecio extendida**.
(b) Aproxime $\int_0^6 f(x)\,dx$ usando la **regla de Simpson compuesta**.
(c) Determine cuál resultado es más preciso.

Solución

```r
f <- function(x){
  if(0 <= x & x <= 2){
    return(x^3 + 1)
  } else if(2 < x & x <= 6){
    return(2*x + 5)
  } else{
    return(NULL)
  }
```

```
}
```

a)

```
tr <- TrapecioExtendido(f, 0, 6, 10)
tr
```

```
[1] 57.9456
```

b)

```
si <- SimpsonCompuesto(f, 0, 6, 10)
si
```

```
[1] 58.0076
```

c)

```
valor_real <- integrate(function(x) x^3 + 1, 0, 2)$value + integrate(function(x) 2*x + 5, 2, 6)
valor_real
```

```
[1] 58
```

error absoluto con trapecio:

```
abs(tr - valor_real)
```

```
[1] 0.0544
```

error absoluto con simpson:

```
abs(si - valor_real)
```

```
[1] 0.0076
```

En este caso, el resultado obtenido con el metodo de Simpson Compuesto es más preciso, entonces es mejor.

Ejercicio 14

> **i** Instrucción del ejercicio 14
>
> Escriba un **algoritmo en pseudo-código** para calcular **integrales dobles tipo I** basado en la **Regla del Trapecio Extendida**.
> Luego, agregue este algoritmo al módulo de integración numérica implementado anteriormente.

Solución

```
Algoritmo Trapecio_Extendido_Doble(f, a, b, c, d, nx, ny)
  # Calcula  _{x=a}^{b}  _{y=c}^{d} f(x, y) dy dx
  # usando la Regla del Trapecio Extendida en ambas direcciones

  Entrada:
      f(x, y): función continua en [a,b]×[c,d]
      a, b : límites en x
      c, d : límites en y
      nx   : número de subintervalos en x
      ny   : número de subintervalos en y

  Salida:
      Aproximación de la integral doble

  h_x ← (b - a) / nx
  h_y ← (d - c) / ny
  suma ← 0

  Para i desde 0 hasta nx:
      x_i ← a + i * h_x
      Para j desde 0 hasta ny:
          y_j ← c + j * h_y
          peso ← 1
          Si (i = 0 ó i = nx) → peso ← peso / 2
          Si (j = 0 ó j = ny) → peso ← peso / 2
          suma ← suma + peso * f(x_i, y_j)
      Fin Para
  Fin Para

  integral ← h_x * h_y * suma
  Retornar integral
Fin Algoritmo
```

```
# Regla del Trapecio Extendida para integrales dobles tipo I
trapecio_doble <- function(f, a, b, c, d, nx, ny) {
  hx <- (b - a) / nx
```

```r
  hy <- (d - c) / ny
  suma <- 0

  for (i in 0:nx) {
    x <- a + i * hx
    for (j in 0:ny) {
      y <- c + j * hy
      peso <- 1

      if (i == 0 || i == nx) peso <- peso / 2
      if (j == 0 || j == ny) peso <- peso / 2

      suma <- suma + peso * f(x, y)
    }
  }

  I <- hx * hy * suma
  return(I)
}
```

```r
# Ejemplo de uso:
f <- function(x, y) x^2 + y^2  # función ejemplo
resultado <- trapecio_doble(f, 0, 1, 0, 1, nx = 10, ny = 10)
cat("Aproximación de la integral doble:", resultado, "\n")
```

```
Aproximación de la integral doble: 0.67
```

Ejercicio 15

Instrucción del ejercicio 15

Repita los ejercicios 9 y 10 usando el algoritmo desarrollado en el ejercicio anterior.

Solución

```r
# Algoritmo Trapecio Extendido con límites en y dependientes de x
trapecio_doble_variable <- function(f, a, b, yinf, ysup, nx = 200, ny = 200) {
  hx <- (b - a) / nx
  suma_total <- 0
```

```r
  for (i in 0:nx) {
    x <- a + i * hx
    ly <- yinf(x); uy <- ysup(x)
    hy <- (uy - ly) / ny
    for (j in 0:ny) {
      y <- ly + j * hy
      peso <- 1
      if (i == 0 || i == nx) peso <- peso / 2
      if (j == 0 || j == ny) peso <- peso / 2
      suma_total <- suma_total + peso * f(x, y)
    }
  }
  hx * ( (ysup((a+b)/2) - yinf((a+b)/2)) / ny ) * suma_total
}

## ------------------------------
## Ejercicio 9 (Área)
## Región: y = 4x - x^2 (0<=x<=1) y y = -3x+6 (1<=x<=2), con y>=0
f_area <- function(x, y) 1
A1 <- trapecio_doble_variable(f_area,
                              a = 0, b = 1,
                              yinf = function(x) 0,
                              ysup = function(x) 4*x - x^2,
                              nx = 300, ny = 300)
A2 <- trapecio_doble_variable(f_area,
                              a = 1, b = 2,
                              yinf = function(x) 0,
                              ysup = function(x) -3*x + 6,
                              nx = 300, ny = 300)
A_num <- A1 + A2
A_exacto <- 19/6
cat(sprintf("Área (num): %.8f   |   Área (exacta): %.8f   |   Error: %.2e\n",
            A_num, A_exacto, A_num - A_exacto))
```

```
Área (num): 3.25000000   |   Área (exacta): 3.16666667   |   Error: 8.33e-02
```

```r
## ------------------------------
## Ejercicio 10 (Volumen)
## z = 4 - x^2 - 2y^2 sobre x^2 + 2y^2 <= 4  (z>=0)
f_vol <- function(x, y) 4 - x^2 - 2*y^2
V_num <- trapecio_doble_variable(f_vol,
                                a = -sqrt(2), b =  sqrt(2),
                                yinf = function(x) -sqrt((4 - x^2)/2),
                                ysup = function(x)  sqrt((4 - x^2)/2),
                                nx = 300, ny = 300)
V_exacto <- 4*pi*sqrt(2)
cat(sprintf("Volumen (num): %.8f   |   Volumen (exacto): %.8f   |   Error: %.2e\n",
            V_num, V_exacto, V_num - V_exacto))
```

```
Volumen (num): 17.77750124   |   Volumen (exacto): 17.77153175   |   Error: 5.97e-03
```

Ejercicio 16

Instrucción del ejercicio 16

Sea

$$f(x) = \int_0^x \cos(t^2)\, dt.$$

(a) Interpole $f(x)$ con nodos $x_0 = 0$, $x_1 = \frac{\pi}{2}$, $x_2 = \pi$,
$x_3 = \frac{3\pi}{2}$, $x_4 = 2\pi$, $x_5 = \frac{5\pi}{2}$, $x_6 = 3\pi$,
usando alguno de los métodos vistos en el curso.

Sugerencia: Recuerde que $\int \cos(t^2)\, dt$ no puede calcularse en términos de funciones elementales.

(b) Grafique el polinomio $p(x)$ y la función $f(x)$ en un mismo plano.

(c) Dé dos razones por las cuales **a priori** se sabe que la interpolación anterior para $f(x)$ es "mala".

Solución

a y b)

```r
library(tidyverse)

neville <- function(nodos, valores, x) {
  stopifnot(is.numeric(nodos),
```

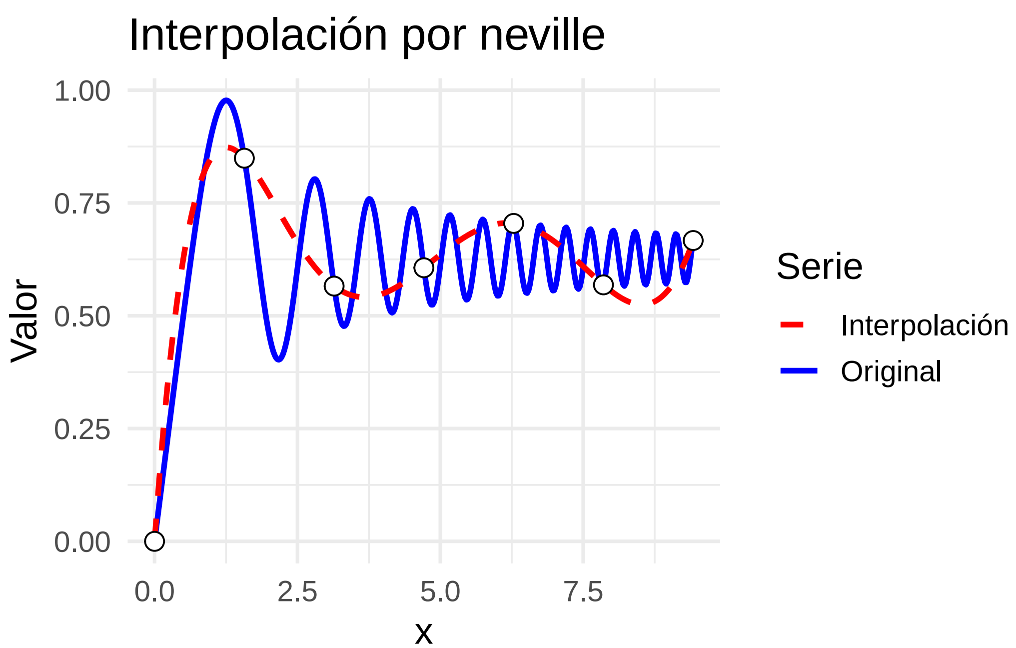
```r
            is.numeric(valores),
            length(nodos) == length(valores))
  n <- length(nodos)
  Q <- matrix(NA_real_, nrow = n, ncol = n)

  # Columna inicial con valores de Y
  Q[, 1] <-  valores

  # Construccion de la tabla de Neville
  for (i in 2:n) {
    for (j in 2:i) {
      numerador <- ((x - nodos[i - j + 1]) * Q[i, j - 1] - (x - nodos[i]) * Q[i -
                                                                                1, j - 1])
      denominador <- nodos[i] - nodos[i - j + 1]
      Q[i, j] <-  numerador / denominador
    }
  }
  return(list(valor = Q[n, n], tabla = Q))
}

graficar.polinomio <- function(nodos,
                               a,
                               b,
                               metodo,
                               f = NULL,
                               valores = NULL,
                               df = NULL,
                               derivadas.clamped = NULL) {
  stopifnot(is.numeric(nodos), length(nodos) >= 2)
  # Validación: exactamente uno de f o valores
  if (is.null(f) == is.null(valores)) {
    stop("Debe proveer exactamente uno: 'f' (función) o 'valores' (numérico).")
  }
  # Obtener valores en nodos según el caso
  if (!is.null(f)) {
    stopifnot(is.function(f))
```

```
    valores_nodos <- f(nodos)
  } else {
    stopifnot(is.numeric(valores), length(valores) == length(nodos))
    valores_nodos <- valores
  }

  # Derivadas (opcional). Acepta función o vector numérico.
  derivadas_nodos <- NULL
  if (!is.null(df)) {
    if (is.function(df)) {
      derivadas_nodos <- df(nodos)
    } else if (is.numeric(df)) {
      stopifnot(length(df) == length(nodos))
      derivadas_nodos <- df
    } else {
      stop("`df` debe ser función o vector numérico de derivadas en los nodos.")
    }
  }

  # Wrapper vectorizado para el método (con o sin derivadas)
  if (!is.null(derivadas_nodos)) {
    H <- function(x)
      vapply(x, function(xx)
        metodo(nodos, valores_nodos, derivadas_nodos, xx)$valor, numeric(1))
  } else {
    if(is.null(derivadas.clamped)){
      H <- function(x)
        vapply(x, function(xx)
          metodo(nodos, valores_nodos, xx)$valor, numeric(1))
    } else{
      H <- function(x)
        vapply(x, function(xx)
          spline.sujeto(nodos, valores_nodos, derivadas.clamped, xx)$valor, numeric(1))
    }
  }
```

```
# Malla y data frames
xi <- seq(a, b, length.out = 400)
df_plot <- data.frame(x  = xi,
                      Hx = H(xi),
                      fx = if (!is.null(f))
                        f(xi)
                      else
                        NA_real_)
df_nodos <- data.frame(x = nodos, y = valores_nodos)

# Gráfico: con f (dos curvas) o solo interpolación
p <- ggplot(df_plot, aes(x = x))
if (!is.null(f)) {
  p <- p +
    geom_line(aes(y = fx, color = "Original"), linewidth = 1) +
    geom_line(aes(y = Hx, color = "Interpolación"),
              linewidth = 1,
              linetype = "dashed") +
    scale_color_manual(values = c(
      "Original" = "blue",
      "Interpolación" = "red"
    ))
} else {
  p <- p +
    geom_line(aes(y = Hx, color = "Interpolación"), linewidth = 1) +
    scale_color_manual(values = c("Interpolación" = "red"))
}
p +
  geom_point(
    data = df_nodos,
    aes(x = x, y = y),
    shape = 21,
    size = 3,
    fill = "white"
  ) +
  labs(title = paste0("Interpolación por ", deparse(substitute(metodo))), y = "Valor", color
```

```
    theme_minimal(base_size = 14)
}
```

```
nodos <- c(0, pi/2, pi, 3*pi/2, 2*pi, 5*pi/2, 3*pi)
imagenes <- numeric(6)
for (i in 1:7) {
  imagenes[i] <- integrate(function(x) cos(x^2), 0, nodos[i])$value
}

f_scalar <- function(x) integrate(function(y) cos(y^2), lower = 0, upper = x)$value
f <- Vectorize(f_scalar)

graficar.polinomio(nodos, 0, 3*pi, neville, f = f)
```



c)

Se puede saber desde antes de hacer la gráfica que la interpolación no va a ser buena por dos motivos.

Primero, los puntos que se eligieron están bastante separados entre sí dentro del intervalo. Eso hace que el polinomio tenga que "rellenar" grandes espacios sin información, y al hacerlo termina alejándose bastante de la forma real de la función.

Segundo, la función original cambia de forma muy rápido, sube y baja muchas veces. Como el polinomio tiene que pasar por todos los puntos, trata de imitar esas variaciones y termina haciendo curvas exageradas que no representan bien el comportamiento verdadero.

Ejercicio 17

> **Instrucción del ejercicio 17**
>
> Determine a, b, c en la fórmula de cuadratura:
>
> $$\int_{-1}^{1} f(x)\,dx = af(-\alpha) + bf(0) + cf(\alpha) + \mathcal{E}(f)$$
>
> en función de α, tal que tenga una **exactitud algebraica igual a 3**, por lo menos.

Solución

Por simetría del intervalo y de los nodos $(-\alpha, 0, \alpha)$, debe cumplirse que los pesos de los puntos simétricos sean iguales:

$$a = c.$$

Así, la fórmula queda:

$$\int_{-1}^{1} f(x)\,dx = a[f(-\alpha) + f(\alpha)] + bf(0) + \mathcal{E}(f).$$

Queremos que la fórmula sea **exacta para todos los polinomios de grado** ≤ 3.

1. Para $f(x) = 1$:

   $$\int_{-1}^{1} 1\,dx = 2, \qquad a(1+1) + b(1) = 2a + b = 2.$$

   Por tanto,

   $$2a + b = 2 \tag{1}$$

2. Para $f(x) = x$:

   $$\int_{-1}^{1} x\,dx = 0, \qquad a(-\alpha + \alpha) + b(0) = 0,$$

   lo cual se cumple automáticamente, como se esperaba por simetría.

3. Para $f(x) = x^2$:

   $$\int_{-1}^{1} x^2\,dx = \frac{2}{3}, \qquad a(\alpha^2 + \alpha^2) + b(0^2) = 2a\alpha^2.$$

   Por tanto,

   $$2a\alpha^2 = \frac{2}{3} \quad \Rightarrow \quad a = \frac{1}{3\alpha^2}. \tag{2}$$

4. Para $f(x) = x^3$:
$$\int_{-1}^{1} x^3\,dx = 0, \qquad a(-\alpha^3 + \alpha^3) + b(0^3) = 0,$$
lo cual también se cumple automáticamente.
Por tanto, la regla será exacta hasta grado 3.

5. Sustituyendo (2) en (1):
$$2\left(\frac{1}{3\alpha^2}\right) + b = 2 \quad \Rightarrow \quad b = 2 - \frac{2}{3\alpha^2}.$$

Los coeficientes son:
$$\boxed{a = c = \frac{1}{3\alpha^2}, \qquad b = 2 - \frac{2}{3\alpha^2}.}$$

Esta regla tiene **exactitud algebraica igual a 3** para todo $\alpha > 0$ y se reduce a una familia de cuadraturas simétricas de tres puntos en $[-1, 1]$.

Ejercicio 18

> **Instrucción del ejercicio 18**
>
> Repita el ejercicio anterior pero con **exactitud algebraica igual a 5**.

Solución

(Derivación del sistema extendido para los polinomios de grado hasta 5 y cálculo de los coeficientes.)

Continuando desde las condiciones del ejercicio anterior, al aplicar la cuarta iteración, $f(x) = x^4$ obtenemos que:

$$\int_{-1}^{1} x^4\,dx = \frac{2}{5}, \qquad a(\alpha^4 + \alpha^4) + b(0^2) = 2a\alpha^4.$$

En la iteración 5, $f(x) = x^5$, volvemos a obtener el resultado de $f(x) = x$ y $f(x) = x^3$. De la ecuación anterior, utilizando (2) tenemos que:

$$\frac{2}{5} = 2\left(\frac{1}{3\alpha^2}\right)\alpha^4$$

$$\Leftrightarrow \alpha = \sqrt{\frac{3}{5}}$$

$$\Rightarrow a = \frac{5}{9} = c \quad \wedge \quad b = \frac{8}{9}$$

Ejercicio 19

Instrucción del ejercicio 19

Demuestre la **regla de Simpson** a partir de la fórmula de cuadratura

$$\int_{x_0}^{x_2} f(x)\,dx = a_0 f(x_0) + a_1 f(x_1) + a_2 f(x_2) + \mathcal{E}(f)$$

usando el hecho de que la fórmula de Simpson tiene **exactitud algebraica igual a 3**.

Solución

(Derivación paso a paso de los coeficientes a_0, a_1, a_2 mediante el ajuste de los momentos exactos de monomios de grado 0 a 3.)

Queremos determinar los coeficientes a_0, a_1, a_2 de la cuadratura tal que sean los que establece la regla de simpson, asumiendo exactitud algebraica 3.

Tome nodos igualmente espaciados: $x_1 = \dfrac{x_1 + x_2}{2}$ y longitud total $2h = x_2 - x_0$

Queremos que la formula sea exacta en $f(x) = 1, x, x^2, x^3$

1- Para $f(x) = 1$

$$\int_{x_0}^{x_2} 1dx = 2h = a_0 + a_1 + a_2$$

2- Para $f(x) = x$

$$\int_{x_0}^{x_2} xdx = \frac{{x_2}^2 - {x_0}^2}{2} = 2x_1 h = a_0 x_0 + a_1 x_1 + a_2 x_2$$

$$\Rightarrow 2x_1 h = x_1(a_0 + a_1 + a_2) + h(a_2 - a_0)$$

Como ya sabemos que $a_0 + a_1 + a_2 = 2h$, entonces debe cumplirse que $a_2 - a_0 = 0 \Rightarrow a_2 = a_1$

3- Para $f(x) = x^2$

$$\int_{x_0}^{x_2} x^2 dx = \frac{{x_2}^3 - {x_0}^3}{3} = \frac{2h}{3}(3{x_1}^2 + h^2)$$

y

$$a_0x_0^2 + a_1x_1^2 + a_2x_2^2 = a_0(x_1-h)^2 + a_1x_1^2 + a_2(x_1+h)^2$$

Reemplazando $a_0 = a_2$

$$= a_0[(x_1-h)^2 + (x_1+h)^2] + a_1x_1^2 = 2a_0(x_1^2+h^2) + a_1x_1^2$$

Igualando y usando $a_0 + a_1 + a_2 = 2h \Rightarrow a_1 = 2h - 2a_0$:

$$\frac{2h}{3}(3x_1^2+h^2) = 2a_0(x_1^2+h^2) + (2h-2a_0)x_1^2$$

Simplificando:

$$\frac{2h}{3}(3x_1^2+h^2) = 2hx_1^2 - 2a_0h^2$$

Despejando a_0:

$$a_0 = \frac{h}{3} \Rightarrow a_1 = 2h - 2a_0 = \frac{4h}{3}$$

Asi pues, podemos ver que

$$\int_{x_0}^{x_1} f(x) \approx \frac{h}{3}(f(x_0) + 4f(x_1) + f(x_2))$$

lo cual es justamente la regla de simpson.

■

Ejercicio 20

> **Instrucción del ejercicio 20**
>
> Demuestre que existe una constante positiva c tal que la fórmula de cuadratura
>
> $$\int_{-1/2}^{1/2} f(x)\,dx = \frac{1}{3}\,[f(-c) + f(0) + f(c)] + \mathcal{E}(f)$$
>
> tiene **exactitud algebraica igual a 3**.

Solución

Buscamos $c > 0$ tal que la regla

$$Q[f] = \tfrac{1}{3}\,[\, f(-c) + f(0) + f(c)\,]$$

reproduzca exactamente los momentos de $I[f] = \displaystyle\int_{-1/2}^{1/2} f(x)\,dx$ para todos los polinomios de grado ≤ 3.

Por simetría del intervalo y de los nodos $(-c, 0, c)$:

- Para $p(x) = 1$,
$$I[p] = \int_{-1/2}^{1/2} 1\,dx = 1, \qquad Q[p] = \tfrac{1}{3}(1+1+1) = 1.$$

- Para $p(x) = x$ y $p(x) = x^3$,
$$I[p] = 0 \quad \text{(funciones impares)}, \qquad Q[p] = \tfrac{1}{3}((-c)^k + 0 + c^k) = 0 \quad (k = 1, 3).$$

- Para $p(x) = x^2$,
$$I[p] = \int_{-1/2}^{1/2} x^2\,dx = \left.\frac{x^3}{3}\right|_{-1/2}^{1/2} = \frac{1}{12},$$
  mientras que
$$Q[p] = \tfrac{1}{3}(c^2 + 0 + c^2) = \frac{2c^2}{3}.$$
  Exigiendo $Q[p] = I[p]$ se obtiene
$$\frac{2c^2}{3} = \frac{1}{12} \quad\Longrightarrow\quad c^2 = \frac{1}{8} \quad\Rightarrow\quad c = \frac{1}{2\sqrt{2}}\ (> 0).$$

Con este valor de c, la regla es exacta para $1, x, x^2, x^3$, por lo que su **exactitud algebraica es 3**. En consecuencia, el término de error $\mathcal{E}(f)$ se anula para todo polinomio de grado ≤ 3 (y, para $f \in C^4$, es de orden $O(\|f^{(4)}\|)$ por el teorema de Peano para reglas de cuadratura). ■

Ejercicio 21

Instrucción del ejercicio 21

Determine las constantes a y b tal que la fórmula de cuadratura

$$\int_0^\infty e^{-x} f(x)\,dx = \frac{1}{4}[af(b) + bf(a)] + \mathcal{E}(f)$$

tenga **exactitud algebraica igual a 3**, por lo menos.

Solución

(Empleo de la función peso e^{-x}, desarrollo de los momentos de Laguerre y resolución del sistema de condiciones para a y b.)

Planteando las condiciones tenemos que:

- $$f(x) = 1 \Rightarrow \int_0^{\infty} e^{-x}\,dx = 1 = \frac{1}{4}\,(a+b) \;\Rightarrow\; a+b=4.$$

- $$f(x) = x \Rightarrow \int_0^{\infty} x\,e^{-x}\,dx = 1 = \frac{1}{4}\,(ab+ab) = \frac{1}{2}\,ab \;\Rightarrow\; ab=2.$$

- $$f(x) = x^2 \Rightarrow \int_0^{\infty} x^2 e^{-x}\,dx = 2 = \frac{1}{4}\,(ab^2+ba^2) = \frac{1}{4}\,ab(a+b) = \frac{1}{4}\cdot 2\cdot 4 = 2$$

$$f(x) = x^3 \Rightarrow \int_0^{\infty} x^3 e^{-x}\,dx = 6 = \frac{1}{4}\,(ab^3+ba^3) = \frac{1}{4}\,ab(a^2+b^2).$$

Note que $a^2+b^2=(a+b)^2-2ab=16-4=12$, por lo que la cuarta condición también es cierta $\left(\frac{1}{4}\cdot 2\cdot 12 = 6\right)$.

Ahora, resolviendo la siguiente ecuación tenemos que:

$$t^2-(a+b)t+ab = t^2-4t+2=0 \Rightarrow a = 2-\sqrt{2} \;\wedge\; b = 2+\sqrt{2}$$

Obteniendo las constantes buscadas.

Ejercicio 22

Instrucción del ejercicio 22

Demuestre que existen constantes $c_1 \in [a,b]$ y $c_2 \in [a,b]$ tal que la fórmula de cuadratura

$$\int_a^b f(x)\,dx = \frac{b-a}{2}\,[f(c_1)+f(c_2)] + \mathcal{E}(f)$$

tiene **exactitud algebraica igual a 3**, por lo menos.

Solución

Buscamos constantes c_1 y c_2 en $[a, b]$ tales que la fórmula de cuadratura

$$\int_a^b f(x)\,dx = \frac{b-a}{2}\left[f(c_1) + f(c_2)\right]$$

tenga exactitud algebraica igual a 3, es decir, que sea exacta para los polinomios $f(x) = 1$, $f(x) = x$, $f(x) = x^2$ y $f(x) = x^3$.

1. Caso $f(x) = 1$

$$\int_a^b 1\,dx = b - a = \frac{b-a}{2}(1+1)$$

Por lo tanto, la fórmula es exacta para funciones constantes.

2. Caso $f(x) = x$

$$\int_a^b x\,dx = \frac{b^2 - a^2}{2} = \frac{b-a}{2}(c_1 + c_2)$$

De aquí se obtiene la relación

$$c_1 + c_2 = a + b$$

3. Caso $f(x) = x^2$

$$\int_a^b x^2\,dx = \frac{b^3 - a^3}{3} = \frac{b-a}{2}(c_1^2 + c_2^2)$$

Simplificando,

$$c_1^2 + c_2^2 = \frac{2}{3}(a^2 + ab + b^2)$$

Definimos las sumas simétricas

$$S_1 = c_1 + c_2 = a + b, \quad S_2 = c_1 c_2$$

Entonces,

$$S_1^2 - 2S_2 = \frac{2}{3}(a^2 + ab + b^2)$$

De donde

$$S_2 = \frac{a^2 + 4ab + b^2}{6}$$

4. Cálculo de los puntos c_1 y c_2

Después de desarrollar un poco el sistema de ecuaciones dado por los casos 2 y 3, se nota que los puntos c_1 y c_2 son las raíces del polinomio cuadrático

$$t^2 - S_1 t + S_2 = 0$$

es decir,

$$c_{1,2} = \frac{S_1 \pm \sqrt{S_1^2 - 4S_2}}{2}$$

Sustituyendo los valores de S_1 y S_2,

$$S_1^2 - 4S_2 = (a+b)^2 - \frac{4(a^2 + 4ab + b^2)}{6} = \frac{(b-a)^2}{3}$$

Por tanto,

$$\boxed{c_1 = \frac{a+b}{2} - \frac{b-a}{2\sqrt{3}}, \quad c_2 = \frac{a+b}{2} + \frac{b-a}{2\sqrt{3}}}$$

Como $0 < 1/\sqrt{3} < 1$, se cumple que c_1 y c_2 pertenecen al intervalo $[a, b]$.

5. Caso $f(x) = x^3$

Usando la identidad

$$c_1^3 + c_2^3 = S_1^3 - 3S_1 S_2$$

y sustituyendo los valores de S_1 y S_2, se verifica que la igualdad también se cumple para $f(x) = x^3$.

Por tanto, existen c_1 y c_2 en $[a, b]$ tales que

$$\int_a^b f(x)\,dx = \frac{b-a}{2}[f(c_1)+f(c_2)] + E(f)$$

y la fórmula tiene exactitud algebraica igual a 3.

Ejercicio 23

Instrucción del ejercicio 23

Usando **integración numérica**, escriba un procedimiento en **R** que permita obtener para una función dada $f(x)$ su **polinomio trigonométrico** $S_n(x)$,
es decir, una **aproximación a su Serie de Fourier**.

Instrucción del inciso (a)

Si se definen:

$$\phi_0(x) = \frac{1}{\sqrt{2\pi}},$$

$$\phi_k(x) = \frac{1}{\sqrt{\pi}}\cos(kx), \quad k = 1, 2, \dots, n,$$

$$\phi_{n+k}(x) = \frac{1}{\sqrt{\pi}}\sin(kx), \quad k = 1, 2, \dots, n-1.$$

Pruebe que el conjunto

$$B = \{\phi_0, \phi_1, \dots, \phi_{2n-1}\}$$

es **ortogonal** con el producto interno integral en el intervalo $[-\pi, \pi]$.
Dada $f \in C[-\pi, \pi]$, su polinomio trigonométrico $S_n(x)$ se define como:

$$S_n(x) := \sum_{k=0}^{2n-1} a_k\,\phi_k(x),$$

donde

$$a_k = \int_{-\pi}^{\pi} f(x)\phi_k(x)\,dx, \quad k = 0, 1, \dots, 2n-1.$$

El límite:

$$\lim_{n\to\infty} S_n(x) = \lim_{n\to\infty} \sum_{k=0}^{2n-1} a_k \phi_k(x)$$

se denomina **Serie de Fourier** de f.

Solución (a)

Usaremos, para $r, s \in \mathbb{Z}_{\geq 0}$,

$\cos(rx)\cos(sx) = \frac{1}{2}\big[\cos((r-s)x)+\cos((r+s)x)\big]$, $\sin(rx)\sin(sx) = \frac{1}{2}\big[\cos((r-s)x)-\cos((r+s)x)\big]$, $\sin(rx)\cos(sx) =$

Además, para $m \in \mathbb{Z} \setminus \{0\}$ se cumple

$$\int_{-\pi}^{\pi} \cos(mx)\, dx = 0, \qquad \int_{-\pi}^{\pi} \sin(mx)\, dx = 0,$$

y $\int_{-\pi}^{\pi} 1\, dx = 2\pi$.

** Productos con ϕ_0 **

- Con sí misma:
$$\langle \phi_0, \phi_0 \rangle = \int_{-\pi}^{\pi} \frac{1}{2\pi}\, dx = \frac{1}{2\pi} \cdot 2\pi = 1.$$

- Con $\phi_k = \frac{1}{\sqrt{\pi}} \cos(kx)$ $(k \geq 1)$:
$$\langle \phi_0, \phi_k \rangle = \frac{1}{\sqrt{2\pi\, \pi}} \int_{-\pi}^{\pi} \cos(kx)\, dx = 0.$$

- Con $\phi_{n+k} = \frac{1}{\sqrt{\pi}} \sin(kx)$ $(k \geq 1)$:
$$\langle \phi_0, \phi_{n+k} \rangle = \frac{1}{\sqrt{2\pi\, \pi}} \int_{-\pi}^{\pi} \sin(kx)\, dx = 0.$$

Por lo tanto, ϕ_0 es ortogonal a todas las demás y tiene norma 1.

Coseno con coseno

Para $k, \ell \geq 1$,

$$\langle \phi_k, \phi_\ell \rangle = \frac{1}{\pi} \int_{-\pi}^{\pi} \cos(kx)\cos(\ell x)\, dx = \frac{1}{2\pi} \int_{-\pi}^{\pi} \big[\cos((k-\ell)x) + \cos((k+\ell)x)\big]\, dx.$$

- Si $k \neq \ell$, ambas integrales son 0, así que $\langle \phi_k, \phi_\ell \rangle = 0$.

- Si $k = \ell$, queda

$$\langle \phi_k, \phi_k \rangle = \frac{1}{2\pi} \int_{-\pi}^{\pi} [1 + \cos(2kx)]\, dx = \frac{1}{2\pi}[2\pi + 0] = 1.$$

Seno con seno

Para $k, \ell \geq 1$,

$$\langle \phi_{n+k}, \phi_{n+\ell} \rangle = \frac{1}{\pi} \int_{-\pi}^{\pi} \sin(kx)\sin(\ell x)\, dx = \frac{1}{2\pi} \int_{-\pi}^{\pi} \left[\cos((k-\ell)x) - \cos((k+\ell)x)\right] dx.$$

- Si $k \neq \ell$, el valor es 0.
- Si $k = \ell$, entonces

$$\langle \phi_{n+k}, \phi_{n+k} \rangle = \frac{1}{2\pi} \int_{-\pi}^{\pi} [1 - \cos(2kx)]\, dx = \frac{1}{2\pi}[2\pi - 0] = 1.$$

Seno con coseno

Para $k, \ell \geq 1$,

$$\langle \phi_{n+k}, \phi_{\ell} \rangle = \frac{1}{\pi} \int_{-\pi}^{\pi} \sin(kx)\cos(\ell x)\, dx = \frac{1}{2\pi} \int_{-\pi}^{\pi} \left[\sin((k+\ell)x) + \sin((k-\ell)x)\right] dx = 0.$$

De los productos con ϕ_0 y el seno con coseno se sigue que

$$\langle \phi_i, \phi_j \rangle = \begin{cases} 1, & i = j, \\ 0, & i \neq j, \end{cases}$$

por lo que $B = \{\phi_0, \phi_1, \dots, \phi_{2n-1}\}$ es no solo **ortogonal** sino **ortonormal** en $L^2([-\pi, \pi])$ con el producto interno dado. ■

> **Instrucción del inciso (b)**
>
> Usando alguno de los métodos de integración numérica programados anteriormente, escriba una función en **R** para calcular:
>
> $$a_k = \int_{-\pi}^{\pi} f(x)\phi_k(x)\, dx, \quad \text{para cada } k = 0, 1, \dots, 2n-1.$$

Solución (b)

```r
coef.fourier <- function(f, k, tipo = "coseno") {

  integrando <- switch(
    tipo,
    "constante" = function(x) f(x),
    "coseno"    = function(x) f(x) * cos(k * x),
    "seno"      = function(x) f(x) * sin(k * x),
    stop("El parámetro 'tipo' debe ser 'constante', 'coseno' o 'seno'.")
  )

  # Cálculo del coeficiente según el tipo
  if (tipo == "constante") {
    return((1 / (2 * pi)) * SimpsonCompuesto(integrando, -pi, pi, 200))
  } else {
    return((1 / pi) * SimpsonCompuesto(integrando, -pi, pi, 200))
  }
}
```

Instrucción del inciso (c)

Programe una función en **R** para calcular la suma:

$$S(n,x) = \sum_{k=0}^{2n-1} a_k \phi_k(x).$$

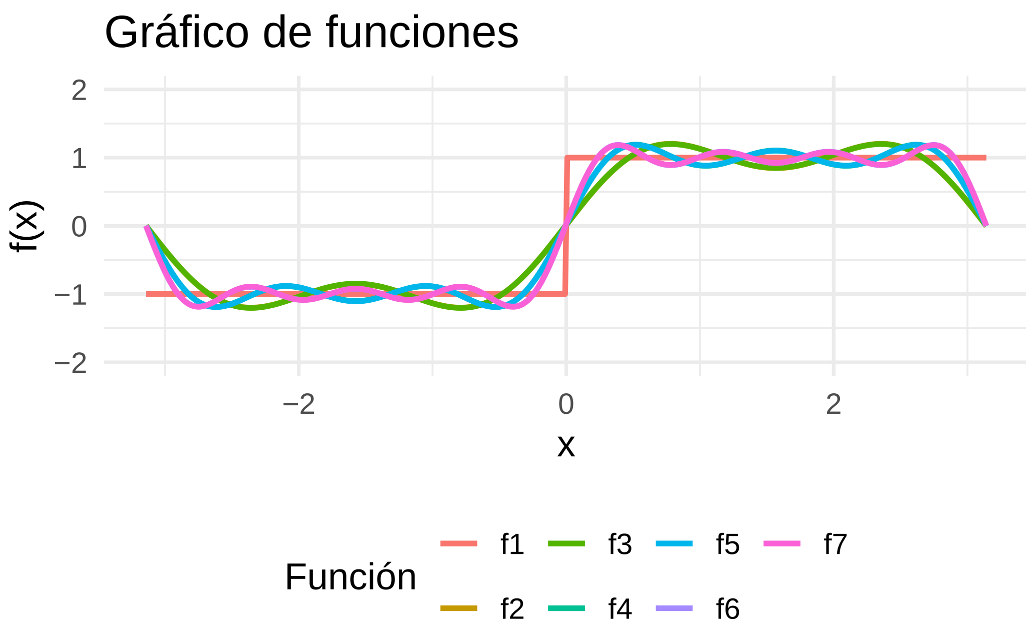
Solución (c)

```r
serie.fourier <- function(f, n, x) {
  a0 <- coef.fourier(f, 0, tipo = "constante")
  suma <- a0
  for (k in 1:n) {
    a_k <- coef.fourier(f, k, tipo = "coseno")
    b_k <- coef.fourier(f, k, tipo = "seno")
    suma <- suma + a_k * cos(k * x) + b_k * sin(k * x)
  }
  return(suma)
}
```

> **Instrucción del inciso (d)**
>
> Con el programa anterior, encontrar $S_n(x)$ para
>
> $n = 3, 4, 5, 6, 7, 8$
>
> para la función:
>
> $$f(x) = \begin{cases} -1, & -\pi < x < 0, \\ 1, & 0 < x < \pi. \end{cases}$$
>
> Luego grafique $f(x)$ y $S_n(x)$ en un mismo plano para $n = 3, 4, 5, 6, 7, 8$.

Solución (d)

```r
funcion.prueba <- function(x) {
  if (-pi <= x && x < 0) {
    return(-1)
  } else if (0 <= x && x <= pi) {
    return(1)
  } else {
    return(NA)
  }
}

funcion.prueba <- Vectorize(funcion.prueba)

graficar(list(function(x) funcion.prueba(x), function(x) serie.fourier(funcion.prueba, 3, x), f
```



En este grafico, f1 representa la funcion original, y de f2 a f7 son las series de fourier incrementando n de 3 a 8