

RESOLUCIÓN DE PROBLEMAS DE CONTORNO

Manuel Carlevaro

Departamento de Ingeniería Mecánica

Grupo de Materiales Granulares - UTN FRLP

manuel.carlevaro@gmail.com

Cálculo Avanzado • 2023

 ·  · 

Problema de contorno

$$y'' = f(x, y, y'), \quad a \leq x \leq b$$

$$\alpha_1 y(a) + \alpha_2 y'(a) = \alpha$$

$$\beta_1 y(b) + \beta_2 y'(b) = \beta$$

Si f es de la forma

$$f(x, y, y') = p(x)y' + q(x)y + r(x)$$

el problema de contorno se llama **lineal**, de otro modo es **no lineal**.

Problema de contorno

$$y'' = f(x, y, y'), \quad a \leq x \leq b$$

$$\alpha_1 y(a) + \alpha_2 y'(a) = \alpha$$

$$\beta_1 y(b) + \beta_2 y'(b) = \beta$$

Si f es de la forma

$$f(x, y, y') = p(x)y' + q(x)y + r(x)$$

el problema de contorno se llama **lineal**, de otro modo es **no lineal**.

Condiciones de borde o frontera:

- ▶ Dirichlet: $\alpha_2 = \beta_2 = 0$
- ▶ Neumann: $\alpha_1 = \beta_1 = 0$
- ▶ Robin: combinación lineal de valores de la función y sus derivadas en la frontera

Problema de contorno

$$y'' = f(x, y, y'), \quad a \leq x \leq b$$

$$\alpha_1 y(a) + \alpha_2 y'(a) = \alpha$$

$$\beta_1 y(b) + \beta_2 y'(b) = \beta$$

Si f es de la forma

$$f(x, y, y') = p(x)y' + q(x)y + r(x)$$

el problema de contorno se llama **lineal**, de otro modo es **no lineal**.

Condiciones de borde o frontera:

- ▶ Dirichlet: $\alpha_2 = \beta_2 = 0$
- ▶ Neumann: $\alpha_1 = \beta_1 = 0$
- ▶ Robin: combinación lineal de valores de la función y sus derivadas en la frontera

Teorema : Existencia y unicidad.

Sea $f(x, y, y') \in C$ en el conjunto

$$D = \{(x, y, y') \mid a \leq x \leq b, -\infty \leq y \leq \infty, \\ -\infty \leq y' \leq \infty\}$$

y que las derivadas parciales f_y y $f_{y'}$ son también continuas en D . Si

- ▶ $f_y(x, y, y') > 0, \forall (x, y, y') \in D$ y
- ▶ existe una constante M tal que

$$|f_{y'}(x, y, y')| \leq M, \forall (x, y, y') \in D$$

entonces el problema con valores de contorno tiene una solución.

Ejemplo: mostrar que

$$y'' + e^{-xy} + \sin y' = 0, \quad 1 \leq x \leq 2$$
$$y(1) = y(2) = 0$$

tiene una solución única.

Solución: tenemos

$$f(x, y, y') = -e^{-xy} - \sin y'$$

y para todas las x en $[1, 2]$:

$$f_y(x, y, y') = xe^{-xy} > 0 \quad \text{y}$$
$$|f_{y'}(x, y, y')| = |-\cos y'| \leq 1$$

Por lo que el problema tiene solución única.

Ejemplo: mostrar que

$$y'' + e^{-xy} + \operatorname{sen} y' = 0, \quad 1 \leq x \leq 2$$
$$y(1) = y(2) = 0$$

tiene una solución única.

Solución: tenemos

$$f(x, y, y') = -e^{-xy} - \operatorname{sen} y'$$

y para todas las x en $[1, 2]$:

$$f_y(x, y, y') = xe^{-xy} > 0 \quad \text{y}$$
$$|f_{y'}(x, y, y')| = |-\cos y'| \leq 1$$

Por lo que el problema tiene solución única.

Teorema : Problema lineal (corolario).

Si el problema lineal

$$y'' = p(x)y' + q(x)y + r(x), \quad a \leq x \leq b$$
$$y(a) = \alpha, \quad y(b) = \beta$$

satisface

- ▶ $p(x)$, $q(x)$ y $r(x)$ son continuas en $[a, b]$
- ▶ $q(x) > 0$ en $[a, b]$

Entonces el problema con valor en la frontera tiene una única solución.

Ejemplo: mostrar que

$$y'' + e^{-xy} + \operatorname{sen} y' = 0, \quad 1 \leq x \leq 2$$
$$y(1) = y(2) = 0$$

tiene una solución única.

Solución: tenemos

$$f(x, y, y') = -e^{-xy} - \operatorname{sen} y'$$

y para todas las x en $[1, 2]$:

$$f_y(x, y, y') = xe^{-xy} > 0 \quad \text{y}$$
$$|f_{y'}(x, y, y')| = |-\cos y'| \leq 1$$

Por lo que el problema tiene solución única.

Teorema : Problema lineal (corolario).

Si el problema lineal

$$y'' = p(x)y' + q(x)y + r(x), \quad a \leq x \leq b$$
$$y(a) = \alpha, \quad y(b) = \beta$$

satisface

- ▶ $p(x)$, $q(x)$ y $r(x)$ son continuas en $[a, b]$
- ▶ $q(x) > 0$ en $[a, b]$

Entonces el problema con valor en la frontera tiene una única solución.

Métodos de solución:

- ▶ Método de disparo
- ▶ Diferencias finitas

Método de disparo: transformamos el problema de condiciones de contorno (Dirichlet):

$$\begin{aligned}y'' &= f(x, y, y'), \quad a \leq x \leq b \\ y(a) &= \alpha, \quad y(b) = \beta\end{aligned}$$

en un problema de valor inicial:

$$\begin{aligned}y'' &= f(x, y, y'), \quad a \leq x \leq b \\ y(a) &= \alpha, \quad y'(a) = r_0\end{aligned}$$

donde r_0 es un parámetro inicial: $y(b) \mapsto y(b, r_0)$

Resolvemos el problema de valor inicial con algún método conocido (por ejemplo Runge-Kutta de cuarto orden).

Si $y(b, r_0)$ no está cerca de β , corregimos la aproximación con r_1, r_2, \dots hasta que

$$|y(b, r_k) - \beta| < \varepsilon$$

Si $f(x, y, y')$ satisface las condiciones de existencia y unicidad, el problema:

$$y(b, r) - \beta = 0$$

es una ecuación no lineal en la variable r .

Método de disparo: transformamos el problema de condiciones de contorno (Dirichlet):

$$y'' = f(x, y, y'), \quad a \leq x \leq b$$
$$y(a) = \alpha, \quad y(b) = \beta$$

en un problema de valor inicial:

$$y'' = f(x, y, y'), \quad a \leq x \leq b$$
$$y(a) = \alpha, \quad y'(a) = r_0$$

donde r_0 es un parámetro inicial: $y(b) \mapsto y(b, r_0)$

Resolvemos el problema de valor inicial con algún método conocido (por ejemplo Runge-Kutta de cuarto orden).

Si $y(b, r_0)$ no está cerca de β , corregimos la aproximación con r_1, r_2, \dots hasta que

$$|y(b, r_k) - \beta| < \varepsilon$$

Si $f(x, y, y')$ satisface las condiciones de existencia y unicidad, el problema:

$$y(b, r) - \beta = 0$$

es una ecuación no lineal en la variable r .

Procedimiento:

- ▶ Seleccionamos aproximaciones iniciales r_0 y r_1 que **encierren** la solución:

$$y(b, r_0) < \beta < y(b, r_1)$$

- ▶ Calculamos la raíz r^* de

$$f_{\text{residuo}}(r) = y(b, r) - \beta$$

con el método de bisección.

- ▶ Resolvemos el problema de valor inicial con

$$y(a) = \alpha, \quad y'(a) = r^*$$

Ejemplo:

$$y'' = 12x - 4y, \quad 0 \leq x \leq 1$$

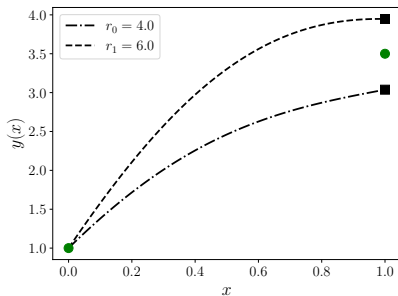
$$y(0) = 1, \quad y(1) = 3.5$$

Exploramos la solución para los valores

$$y'(0) = r_1 = 4 \text{ y } y'(0) = r_2 = 6:$$

```
9 def f(x, y):  
10     f = np.zeros(2)  
11     f[0] = y[1]  
12     f[1] = 12 * x - 4 * y[0]  
13     return f
```

```
22 x_a, x_b = 0.0, 1.0  
23 n_points = 100  
24 beta = 3.5  
25 x = np.linspace(x_a, x_b, n_points)  
26  
27 sol_1 = solve_ivp(f, [x_a, x_b], [1.0, 4.0], dense_output=True)  
28 y_1 = sol_1.sol(x)  
29 sol_2 = solve_ivp(f, [x_a, x_b], [1.0, 6.0], dense_output=True)  
30 y_2 = sol_2.sol(x)  
31 plt.plot(x, y_1.T[:, 0], '-.k', label=r"$r_0 = 4.0$")  
32 plt.plot(x, y_2.T[:, 0], '--k', label=r"$r_1 = 6.0$")  
33 plt.plot([0, 1], [1.0, 3.5], 'go')
```



```
15 def residuo(r, a, b, beta):
16     x = np.linspace(a, b, 100)
17     sol = solve_ivp(f, [a, b], [1, r], dense_output=True)
18     z = sol.sol(x)
19     resid = z.T[-1, 0] - beta
20     return resid
```

```
36 r_opt = bisect(residuo, 4.0, 6.0, args=(x_a, x_b, beta),
37     full_output=True)
38 print(r_opt[1])
39 sol = solve_ivp(f, [x_a, x_b], [1.0, r_opt[0]],
40     dense_output=True)
41 y = sol.sol(x)
42 plt.plot(x, y.T[:, 0], '-r', label=r"$r^* = 5.01665$")
```

```
$ ./disparo.py
    converged: True
      flag: 'converged'
function_calls: 42
  iterations: 40
      root: 5.016654027140248
```

```

15 def residuo(r, a, b, beta):
16     x = np.linspace(a, b, 100)
17     sol = solve_ivp(f, [a, b], [1, r], dense_output=True)
18     z = sol.sol(x)
19     resid = z.T[-1, 0] - beta
20     return resid

```

```

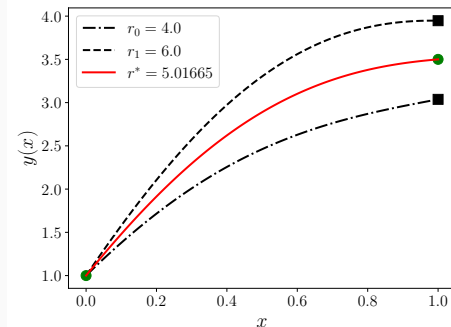
36 r_opt = bisect(residuo, 4.0, 6.0, args=(x_a, x_b, beta),
37     full_output=True)
38 print(r_opt[1])
39 sol = solve_ivp(f, [x_a, x_b], [1.0, r_opt[0]],
40     dense_output=True)
41 y = sol.sol(x)
42 plt.plot(x, y.T[:, 0], '-r', label=r"$r^* = 5.01665$")

```

```

$ ./disparo.py
    converged: True
      flag: 'converged'
function_calls: 42
  iterations: 40
      root: 5.016654027140248

```



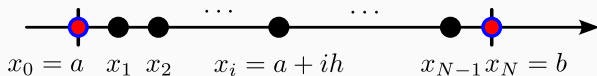
Diferencias finitas: aproximación de derivadas por diferencias finitas. Problema lineal con valores de contorno:

$$y'' = p(x)y' + q(x)y + r(x), \quad a \leq x \leq b, \quad y(a) = \alpha, \quad y(b) = \beta \quad (1)$$

Diferencias finitas: aproximación de derivadas por diferencias finitas. Problema lineal con valores de contorno:

$$y'' = p(x)y' + q(x)y + r(x), \quad a \leq x \leq b, \quad y(a) = \alpha, \quad y(b) = \beta \quad (1)$$

Malla: dividimos $[a, b]$ en N subintervalos:

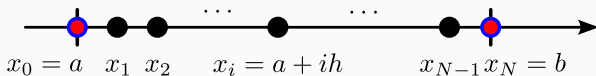


con $h = (b - a)/N$, $x_i = a + ih$, $i = 0, 1, \dots, N - 1, N$.

Diferencias finitas: aproximación de derivadas por diferencias finitas. Problema lineal con valores de contorno:

$$y'' = p(x)y' + q(x)y + r(x), \quad a \leq x \leq b, \quad y(a) = \alpha, \quad y(b) = \beta \quad (1)$$

Malla: dividimos $[a, b]$ en N subintervalos:



con $h = (b - a)/N$, $x_i = a + ih$, $i = 0, 1, \dots, N - 1, N$.

Expansión de y : polinomio de Taylor, alrededor de x_i evaluando en x_{i+1} y x_{i-1} , con $y \in C^4[x_{i-1}, x_{i+1}]$:

$$y(x_i + h) = y(x_i) + hy'(x_i) + \frac{h^2}{2}y''(x_i) + \frac{h^3}{6}y'''(x_i) + \frac{h^4}{24}y^{(4)}(\xi_i^+)$$

$$y(x_i - h) = y(x_i) - hy'(x_i) + \frac{h^2}{2}y''(x_i) - \frac{h^3}{6}y'''(x_i) + \frac{h^4}{24}y^{(4)}(\xi_i^-)$$

con $\xi_i^+(x_i, x_i + h)$ y $\xi_i^- \in (x_i - h, x_i)$. Restando y sumando:

$$y'(x_i) = \frac{1}{2h}[y(x_{i+1}) - y(x_{i-1})] - \frac{h^2}{6}y'''(\eta_i)$$

$$y''(x_i) = \frac{1}{h^2}[y(x_{i+1}) - 2y(x_i) + y(x_{i-1}))] - \frac{h^2}{12}y^{(4)}(\xi_i)$$

(2)

con $\eta_i, \xi_i \in (x_{i-1}, x_{i+1})$.

Notación: $y(x_i) \mapsto y_i$,

$f(x_i) \mapsto f_i$, $f = p, q, r$.

Reemplazando (2) en (1):

$$\frac{y_{i+1} - 2y_i + y_{i-1}}{h^2} = p_i \left[\frac{y_{i+1} - y_{i-1}}{2h} \right] + q_i y_i + r_i - \frac{h^2}{12} [2p_i y'''(\eta_i) - y^{(4)}(\xi_i)]$$

Reemplazando (2) en (1):

$$\frac{y_{i+1} - 2y_i + y_{i-1}}{h^2} = p_i \left[\frac{y_{i+1} - y_{i-1}}{2h} \right] + q_i y_i + r_i - \frac{h^2}{12} [2p_i y'''(\eta_i) - y^{(4)}(\xi_i)]$$

Condiciones de borde $y_0 = \alpha$, $y_{N+1} = \beta$, error de truncamiento $\mathcal{O}(h^2) \mapsto$ sistema de ecuaciones lineales:

$$\left(\frac{y_{i+1} - 2y_i + y_{i-1}}{h^2} \right) + p(x_i) \left(\frac{y_{i+1} - y_{i-1}}{2h} \right) + q(x_i)y_i = -r(x_i)$$

para los puntos interiores de la malla $i = 1, 2, \dots, N - 1$. Reordenando:

$$\left(-1 - \frac{h}{2}p_i \right) y_{i-1} + (2 + h^2 q_i) y_i + \left(-1 + \frac{h}{2}p_i \right) y_{i+1} = -h^2 r(x_i)$$

Sistema con matriz tridiagonal $(N - 1) \times (N - 1)$:

$$\mathbf{A}\mathbf{y} = \mathbf{b}$$

Para $i = 1$:

$$y_{i-1} = y_0 = y(a) = \alpha$$

Para $i = N - 1$:

$$y_{i+1} = y_N = y(b) = \beta$$

$$\mathbf{A} = \begin{bmatrix} 2 + h^2 q_1 & -1 + \frac{h}{2} p_1 & & & & 0 \\ -1 - \frac{h}{2} p_2 & 2 + h^2 q_2 & -1 + \frac{h}{2} p_2 & & & \\ -1 - \frac{h}{2} p_3 & 2 + h^2 q_3 & -1 + \frac{h}{2} p_3 & & & \\ & & \ddots & & & \\ & & & -1 - \frac{h}{2} p_{N-2} & 2 + h^2 q_{N-2} & -1 + \frac{h}{2} p_{N-2} \\ 0 & & & & -1 - \frac{h}{2} p_{N-1} & 2 + h^2 q_{N-1} \end{bmatrix}$$

$$\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_{N-2} \\ y_{N-1} \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} -h^2 r_1 + \left(1 + \frac{h}{2} p_1\right) \alpha \\ -h^2 r_2 \\ \vdots \\ -h^2 r_{N-2} \\ -h^2 r_{N-1} + \left(1 - \frac{h}{2} p_{N-1}\right) \beta \end{bmatrix}$$

Ejemplo:

$$y'' = 12x - 4y, 0 \leq x \leq 1$$

$$y(0) = 1, y(1) = 3.5$$

```
7 def p(x):
8     return 0
9
10 def q(x):
11     return -4
12
13 def r(x):
14     return 12 * x
15
16 def solve_DF(N):
17     a, b = 0, 1
18     alfa, beta = 1, 3.5
19     h = (b - a) / N
20     x = np.linspace(a, b, N + 1)
21     print("x = ", x, "h = ", h)
```

```
22 # Matriz A
23 A = np.zeros((N-1, N-1))
24 A[0, 0] = 2 + h**2 * q(x[0])
25 A[0, 1] = -1 + h / 2 * p(x[0])
26 A[N-2, N - 3] = -1 - h / 2 * p(x[N-1])
27 A[N-2, N-2] = 2 + h**2 * q(x[N-1])
28 for i in range(1, N-2):
29     A[i, i - 1] = -1 - h / 2 * p(x[i])
30     A[i, i] = 2 + h**2 * q(x[i])
31     A[i, i + 1] = -1 + h / 2 * p(x[i])
32 print(A)
33 # Vector b
34 b = np.zeros(N-1)
35 b[0] = -h**2 * r(x[0]) + (1 + h / 2 * p(x[0])) * alfa
36 b[1 : -1] = - h**2 * r(x[1 : -3])
37 b[-1] = -h**2 * r(x[N-1]) + (1 - h / 2 * p(x[N-1])) * beta
38 print("b = ", b)
39 # Resolvemos para y
40 y = np.zeros(N + 1)
41 y[0] = alfa
42 y[1:-1] = np.linalg.solve(A, b)
43 y[-1] = beta
44 return x, y
```

```

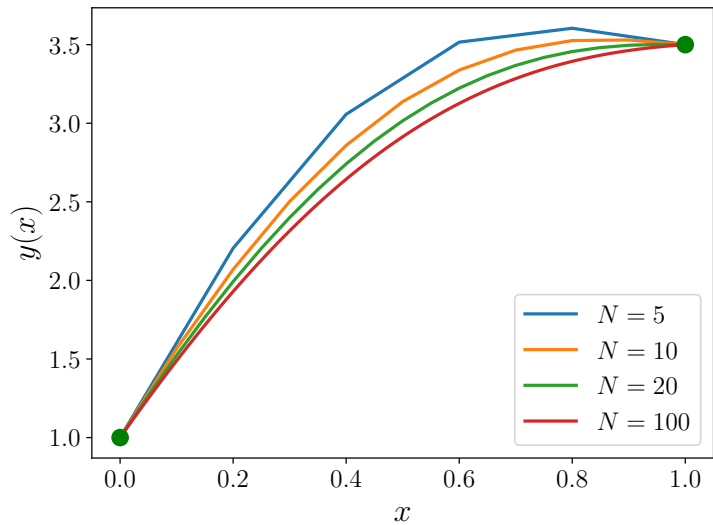
46 # Graficamos las soluciones
47 for N in [5, 10, 20, 100]:
48     print(f"N = {N}")
49     x, y = solve_DF(N)
50     plt.plot(x, y, label=f"$N = {N}$")
51
52 plt.plot([0, 1], [1.0, 3.5], 'go')
53 plt.xlabel(r"$x$")
54 plt.ylabel(r"$y(x)$")
55 plt.legend(loc=4)
56 plt.tight_layout()
57 plt.savefig("dif-finitas.pdf")

```

```

N = 10
x = [0.  0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1. ] h =  0.1
[[ 1.96 -1.    0.    0.    0.    0.    0.    0.    0.    0. ]
 [-1.    1.96 -1.    0.    0.    0.    0.    0.    0.    0. ]
 [ 0.   -1.    1.96 -1.    0.    0.    0.    0.    0.    0. ]
 [ 0.    0.   -1.    1.96 -1.    0.    0.    0.    0.    0. ]
 [ 0.    0.    0.   -1.    1.96 -1.    0.    0.    0.    0. ]
 [ 0.    0.    0.    0.   -1.    1.96 -1.    0.    0.    0. ]
 [ 0.    0.    0.    0.    0.   -1.    1.96 -1.    0.    0. ]
 [ 0.    0.    0.    0.    0.    0.   -1.    1.96 -1.    0. ]
 [ 0.    0.    0.    0.    0.    0.    0.   -1.    1.96 ]]
b = [ 1.   -0.012 -0.024 -0.036 -0.048 -0.06  -0.072 -0.084  3.392]

```



- ▶ R.L. Burden, D.J. Faires y A.M. Burden. ***Análisis numérico***. 10.^a ed. Mexico: Cengage Learning, 2017. Capítulo 11.
- ▶ B. Bradie. ***A Friendly Introduction to Numerical Analysis***. New Jersey, United States: Pearson Education Inc., 2006. Capítulo 8.
- ▶ J. Kiusalaas. ***Numerical Methods in Engineering with Python 3***. Cambridge, United Kingdom: Cambridge University Press, 2013. Capítulo 8.
- ▶ J.H. Mathews y K.D. Fink. ***Numerical methods using MATLAB***. New Jersey, United States: Pearson Education Inc., 2004. Capítulo 9.