

# RESOLUCIÓN DE PROBLEMAS DE VALOR INICIAL

---

Manuel Carlevaro

Departamento de Ingeniería Mecánica

Grupo de Materiales Granulares - UTN FRLP

manuel.carlevaro@gmail.com

Cálculo Avanzado • 2023

 ·  · 

**Problema de valor inicial:**

$$\begin{cases} y'(t) = f[t, y(t)], & t_0 \leq t \leq T \\ y(t_0) = y_0 \end{cases}$$

## Problema de valor inicial:

$$\begin{cases} y'(t) = f[t, y(t)], & t_0 \leq t \leq T \\ y(t_0) = y_0 \end{cases}$$

### Definición : Condición de Lipschitz.

Una función  $f(t, y)$  satisface una **condición de Lipschitz** en la variable  $y$ , y en un conjunto  $D \in \mathbf{R}^2$  si existe una constante  $L > 0$  tal que

$$|f(t, y_1) - f(t, y_2)| \leq L |y_1 - y_2|$$

siempre que  $(t, y_1)$  y  $(t, y_2)$  estén en  $D$ . La constante  $L$  se llama **constante de Lipschitz** para  $f$ .

### Problema de valor inicial:

$$\begin{cases} y'(t) = f[t, y(t)], & t_0 \leq t \leq T \\ y(t_0) = y_0 \end{cases}$$

#### Definición : Condición de Lipschitz.

Una función  $f(t, y)$  satisface una **condición de Lipschitz** en la variable  $y$ , y en un conjunto  $D \in \mathbf{R}^2$  si existe una constante  $L > 0$  tal que

$$|f(t, y_1) - f(t, y_2)| \leq L |y_1 - y_2|$$

siempre que  $(t, y_1)$  y  $(t, y_2)$  estén en  $D$ . La constante  $L$  se llama **constante de Lipschitz** para  $f$ .

**Ejemplo:** mostrar que  $f(t, y) = t|y|$  satisface una condición de Lipschitz en el intervalo

$$D = \{(t, y) \mid 1 \leq t \leq 2 \text{ y } -3 \leq y \leq 4\}:$$

Para cada par de puntos  $(t, y_1)$  y  $(t, y_2)$  en  $D$ , tenemos:

$$\begin{aligned} |f(t, y_1) - f(t, y_2)| &= |t|y_1| - t|y_2|| \\ &= |t| ||y_1| - |y_2|| \\ &\leq 2 |y_1 - y_2| \end{aligned}$$

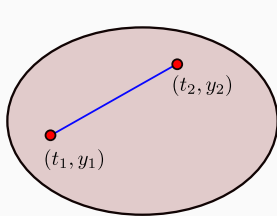
$f$  satisface una condición de Lipschitz sobre  $D$  con  $L = 2$  (menor valor posible). Por ejemplo:

$$|f(2, 1) - f(2, 0)| = |2 - 0| = 2 |1 - 0|$$

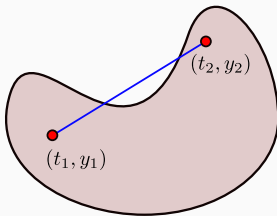
### Definición : Conjunto convexo.

Un conjunto  $D \in \mathbf{R}^2$  se dice que es **convexo** si, dados  $(t_1, y_1)$  y  $(t_2, y_2)$  pertenecientes a  $D$ , entonces

$$[(1 - \lambda)t_1 + \lambda t_2, (1 - \lambda)y_1 + \lambda y_2] \in D \quad \forall \lambda \in [0, 1]$$



Convexo

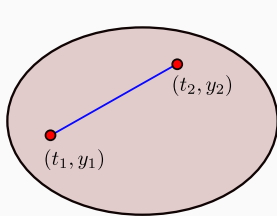


No convexo

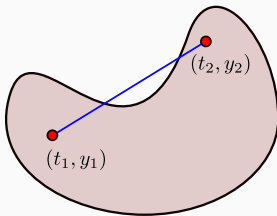
### Definición : Conjunto convexo.

Un conjunto  $D \in \mathbf{R}^2$  se dice que es **convexo** si, dados  $(t_1, y_1)$  y  $(t_2, y_2)$  pertenecientes a  $D$ , entonces

$$[(1 - \lambda)t_1 + \lambda t_2, (1 - \lambda)y_1 + \lambda y_2] \in D \quad \forall \lambda \in [0, 1]$$



Convexo



No convexo

### Teorema : .

Sea  $f(t, y)$  definida en un conjunto convexo  $D \in \mathbf{R}^2$ .

Si existe una constante  $L > 0$  tal que

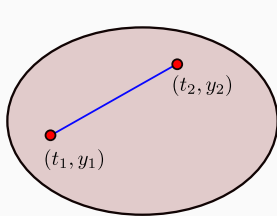
$$\left| \frac{\partial f(t, y)}{\partial y} \right| \leq L, \quad \forall (t, y) \in D$$

entonces  $f$  satisface una condición de Lipschitz sobre  $D$  en la variable  $y$  con constante de Lipschitz  $L$ .

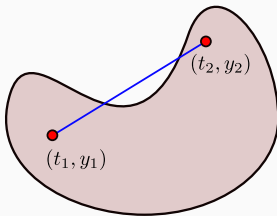
### Definición : Conjunto convexo.

Un conjunto  $D \in \mathbf{R}^2$  se dice que es **convexo** si, dados  $(t_1, y_1)$  y  $(t_2, y_2)$  pertenecientes a  $D$ , entonces

$$[(1 - \lambda)t_1 + \lambda t_2, (1 - \lambda)y_1 + \lambda y_2] \in D \quad \forall \lambda \in [0, 1]$$



Convexo



No convexo

### Teorema : .

Sea  $f(t, y)$  definida en un conjunto convexo  $D \in \mathbf{R}^2$ .

Si existe una constante  $L > 0$  tal que

$$\left| \frac{\partial f(t, y)}{\partial y} \right| \leq L, \quad \forall (t, y) \in D$$

entonces  $f$  satisface una condición de Lipschitz sobre  $D$  en la variable  $y$  con constante de Lipschitz  $L$ .

### Teorema : .

Sea  $D = \{(t, y) \mid t_0 \leq t \leq T \text{ y } -\infty < y < \infty\}$ , y  $f(t, y)$  continua en  $D$ . Si  $f$  satisface una condición de Lipschitz sobre  $D$  en la variable  $y$ , entonces el problema de valor inicial:

$$\begin{cases} y'(t) = f(t, y), & t_0 \leq t \leq T \\ y(t_0) = y_0 \end{cases}$$

tiene una solución única  $y(t)$  para  $t_0 \leq t \leq T$ .

### **Problema bien formulado:**

- ▶ Existencia: el problema tiene al menos una solución.
- ▶ Unicidad: el problema tiene solución única.
- ▶ Estabilidad: pequeñas variaciones en los datos de entrada no generan grandes cambios en la solución.



## Problema bien formulado:

- ▶ Existencia: el problema tiene al menos una solución.
- ▶ Unicidad: el problema tiene solución única.
- ▶ Estabilidad: pequeñas variaciones en los datos de entrada no generan grandes cambios en la solución.

El problema de valor inicial:

$$\begin{cases} y'(t) = f(t, y), & t_0 \leq t \leq T \\ y(t_0) = y_0 \end{cases}$$

está bien formulado si:

- ▶ Existe una solución única  $y(t)$ , y
- ▶ Existen constantes  $\varepsilon_0 > 0$  y  $k > 0$  tales que para cada cualquier  $\varepsilon \in (0, \varepsilon_0)$ , siempre que  $\delta(t)$  sea continua con  $|\delta(t)| < \varepsilon$ ,  $\forall t \in [t_0, T]$ , y cuando  $|\delta_0| < \varepsilon$ , el problema de valor inicial:

$$\begin{cases} z'(t) = f(t, z) + \delta(t), & t_0 \leq t \leq T \\ z(t_0) = y_0 + \delta_0 \end{cases}$$

tiene una solución única que satisface

$$|z(t) - y(t)| < k\varepsilon, \quad \forall t \in [t_0, T]$$

### Teorema : .

Sea  $D = \{(t, y) \mid t_0 \leq t \leq T \text{ y } -\infty < y < \infty\}$ . Si  $f$  es continua y satisface una condición de Lipschitz en la variable  $y$  sobre el conjunto  $D$ , el problema de valor inicial:

$$\begin{cases} y'(t) = f(t, y), & t_0 \leq t \leq T \\ y(t_0) = y_0 \end{cases}$$

está bien formulado.

**Serie de Taylor:**  $f \in C^{(\infty)}$

$$f(t) = f(t_0) + \frac{f'(t_0)}{1!}(t - t_0) + \frac{f''(t_0)}{2!}(t - t_0)^2 \\ + \frac{f^{(3)}(t_0)}{3!}(t - t_0)^3 + \dots$$

**Serie de Taylor:**  $f \in C^{(\infty)}$

$$f(t) = f(t_0) + \frac{f'(t_0)}{1!}(t - t_0) + \frac{f''(t_0)}{2!}(t - t_0)^2 \\ + \frac{f^{(3)}(t_0)}{3!}(t - t_0)^3 + \dots$$

**Teorema : Teorema de Taylor.**

Sea  $k \geq 1$  un entero y  $f : \mathbf{R} \mapsto \mathbf{R}$   $k$  veces diferenciable en  $t_0 \in \mathbf{R}$ . Entonces existe  $h_k : \mathbf{R} \mapsto \mathbf{R}$  tal que:

$$f(t) = f(t_0) + \frac{f'(t_0)}{1!}(t - t_0) + \frac{f''(t_0)}{2!}(t - t_0)^2 \\ + \frac{f^{(3)}(t_0)}{3!}(t - t_0)^3 + \dots + \frac{f^{(k)}(t_0)}{k!}(t - t_0)^k \\ + h_k(t)(t - t_0)^k$$

y

$$\lim_{t \rightarrow t_0} h_k(t) = 0$$

**Serie de Taylor:**  $f \in C^{(\infty)}$

$$f(t) = f(t_0) + \frac{f'(t_0)}{1!}(t - t_0) + \frac{f''(t_0)}{2!}(t - t_0)^2 \\ + \frac{f^{(3)}(t_0)}{3!}(t - t_0)^3 + \dots$$

### Teorema : Teorema de Taylor.

Sea  $k \geq 1$  un entero y  $f : \mathbf{R} \mapsto \mathbf{R}$   $k$  veces diferenciable en  $t_0 \in \mathbf{R}$ . Entonces existe  $h_k : \mathbf{R} \mapsto \mathbf{R}$  tal que:

$$f(t) = f(t_0) + \frac{f'(t_0)}{1!}(t - t_0) + \frac{f''(t_0)}{2!}(t - t_0)^2 \\ + \frac{f^{(3)}(t_0)}{3!}(t - t_0)^3 + \dots + \frac{f^{(k)}(t_0)}{k!}(t - t_0)^k \\ + h_k(t)(t - t_0)^k$$

y

$$\lim_{t \rightarrow t_0} h_k(t) = 0$$

**Forma de Lagrange para el resto:** Sea  $f : \mathbf{R} \rightarrow \mathbf{R}$   $k + 1$  veces diferenciable en  $(t_0, t)$  con  $f^{(k)}$  continua en  $t_0, t$ :

$$R_k(t) = \frac{f^{(k+1)}(\tau)}{(k+1)!}(t - t_0)^{k+1}$$

para  $t_0 \leq \tau \leq t$ .

**Serie de Taylor:**  $f \in C^{(\infty)}$

$$f(t) = f(t_0) + \frac{f'(t_0)}{1!}(t - t_0) + \frac{f''(t_0)}{2!}(t - t_0)^2 + \frac{f^{(3)}(t_0)}{3!}(t - t_0)^3 + \dots$$

### Teorema : Teorema de Taylor.

Sea  $k \geq 1$  un entero y  $f : \mathbf{R} \mapsto \mathbf{R}$   $k$  veces diferenciable en  $t_0 \in \mathbf{R}$ . Entonces existe  $h_k : \mathbf{R} \mapsto \mathbf{R}$  tal que:

$$f(t) = f(t_0) + \frac{f'(t_0)}{1!}(t - t_0) + \frac{f''(t_0)}{2!}(t - t_0)^2 + \frac{f^{(3)}(t_0)}{3!}(t - t_0)^3 + \dots + \frac{f^{(k)}(t_0)}{k!}(t - t_0)^k + h_k(t)(t - t_0)^k$$

$y$

$$\lim_{t \rightarrow t_0} h_k(t) = 0$$

**Forma de Lagrange para el resto:** Sea  $f : \mathbf{R} \rightarrow \mathbf{R}$   $k + 1$  veces diferenciable en  $(t_0, t)$  con  $f^{(k)}$  continua en  $t_0, t$ :

$$R_k(t) = \frac{f^{(k+1)}(\tau)}{(k+1)!}(t - t_0)^{k+1}$$

para  $t_0 \leq \tau \leq t$ .

**Método de Euler:**  $h = t_1 - t_0, y'(t) = f(t, y)$

$$\begin{aligned} y_1 = y(t_1) &= y(t_0) + y'(t_0)(t_1 - t_0) + \frac{y''(\tau)}{2}(t_1 - t_0)^2 \\ &= y_0 + hf[t_0, y(t_0, y_0)] + y''(\tau) \frac{h^2}{2} \end{aligned}$$

- Aproximación:  $y_1 = y_0 + hf(t_0, y_0)$
- Error local:  $\mathcal{O}(h^2)$
- Error global:  $\mathcal{O}(h)$

**Método de Taylor:**  $y$  tiene  $n + 1$  derivadas continuas en  $[t_0, T]$ , expansión de Taylor alrededor de  $t_i$ :

$$y(t) = y_0 + y'_i(t - t_i) + \frac{y''_i}{2}(t - t_i)^2 + \cdots + \frac{y_i^{(n)}}{n!}(t - t_i)^n + \frac{y^{(n+1)}(\tau)}{(n+1)!}(t - t_i)^{n+1}$$

$$y'_i = f(t_i, y_i)$$

$$y''_i = \left. \frac{d}{dt} f(t, y) \right|_{t=t_i} = \left( \frac{\partial f}{\partial t} + \frac{\partial f}{\partial y} f \right) \Big|_{t=t_i}$$

$$y'''_i = \left. \frac{d^2}{dt^2} f(t, y) \right|_{t=t_i} = \left( \frac{\partial^2 f}{\partial t^2} + 2f \frac{\partial^2 f}{\partial t \partial y} + \frac{\partial^2 f}{\partial y^2} f^2 + \frac{\partial f}{\partial y} \frac{\partial f}{\partial t} + \left( \frac{\partial f}{\partial y} \right)^2 f \right) \Big|_{t=t_i}$$

Evaluando en  $t = t_{i+1}$ , descartando el resto y haciendo  $h = t_{i+1} - t_i$ :

$$y_{i+1} = y_i + hf(t_i, y_i) + \frac{h^2}{2} \left. \frac{d}{dt} f(t, y) \right|_{(t_i, y_i)} + \cdots + \frac{h^n}{n!} \left. \frac{d^{n-1}}{dt^{n-1}} f(t, y) \right|_{(t_i, y_i)}$$

**Nota:** el método de Euler es el de Taylor con  $n = 1$ .

## Runge-Kutta de segundo orden

$$\begin{cases} y' = f(t, y), & t_0 \leq t \leq T \\ y(t_0) = y_0 \end{cases}$$

$$y_{n+1} = y_n + \int_{t_n}^{t_{n+1}} f(t, y) dt$$

Regla del trapecio:

$$\int_{t_n}^{t_{n+1}} f(t, y) dt \simeq \frac{h}{2} [f(t_n, y_n) + f(t_{n+1}, y_{n+1})]$$

Aproximamos:

$$y_{n+1} \approx \bar{y}_{n+1} = y_n + hf(t_n, y_n)$$

$$y_{n+1} = y_n + \frac{h}{2} [f(t_n, y_n) + f(t_{n+1}, \bar{y}_{n+1})]$$

Forma canónica:

$$k_1 = hf(t_n, y_n)$$

$$k_2 = hf(t_{n+1}, y_n + k_1)$$

$$y_{n+1} = y_n + \frac{1}{2}(k_1 + k_2)$$

Errores:

► Error local:  $\mathcal{O}(h^3)$

► Error global  $\mathcal{O}(h^2)$

RK2 óptimo (minimiza coeficiente de error):

$$k_1 = \frac{2}{3}hf(t_i, y_i)$$

$$k_2 = \frac{3}{4}hf(t_i + \frac{2h}{3}, k_1)$$

$$y_{n+1} = y_n + \frac{1}{4} \left( \frac{3}{2}k_1 + 4k_2 \right)$$

## Runge-Kutta de cuarto orden

**Nota:** Si  $f(t, y) = f(t)$ : integración de 1/3 de Simpson

$$\begin{cases} y' = f(t, y), & t_0 \leq t \leq T \\ y(t_0) = y_0 \end{cases}$$

$$y_{n+1} = y_n + \frac{h}{6} [f(x_n) + 4f(x_n + \frac{h}{2}) + f(t_{n+1})]$$

$$y_{n+1} = y_n + \frac{1}{6}(k_0 + 2k_1 + 2k_2 + k_3)$$

con

$$k_0 = hf(t_n, y_n)$$

$$k_1 = hf\left(t_n + \frac{h}{2}, y_n + \frac{k_0}{2}\right)$$

$$k_2 = hf\left(t_n + \frac{h}{2}, y_n + \frac{k_1}{2}\right)$$

$$k_3 = hf(t_n + h, y_n + k_2)$$

Errores:

- ▶ Error local:  $\mathcal{O}(h^5)$
- ▶ Error global  $\mathcal{O}(h^4)$



## Runge-Kutta de cuarto orden

$$\begin{cases} y' = f(t, y), & t_0 \leq t \leq T \\ y(t_0) = y_0 \end{cases}$$

$$y_{n+1} = y_n + \frac{1}{6}(k_0 + 2k_1 + 2k_2 + k_3)$$

con

$$k_0 = hf(t_n, y_n)$$

$$k_1 = hf\left(t_n + \frac{h}{2}, y_n + \frac{k_0}{2}\right)$$

$$k_2 = hf\left(t_n + \frac{h}{2}, y_n + \frac{k_1}{2}\right)$$

$$k_3 = hf(t_n + h, y_n + k_2)$$

Errores:

- ▶ Error local:  $\mathcal{O}(h^5)$
- ▶ Error global  $\mathcal{O}(h^4)$

**Nota:** Si  $f(t, y) = f(t)$ : integración de 1/3 de Simpson

$$y_{n+1} = y_n + \frac{h}{6} [f(x_n) + 4f(x_n + \frac{h}{2}) + f(t_{n+1})]$$

**Sistema de ecuaciones:**

$$\begin{cases} \mathbf{y}'(t) = \mathbf{f}(t, \mathbf{y}(t)) \\ \mathbf{y}(t_0) = \mathbf{y}_0 \end{cases}$$

Euler:

$$\mathbf{y}_{n+1} = \mathbf{y}_n + h\mathbf{f}(t_n, \mathbf{y}_n)$$

RK4:

$$\mathbf{k}_0 = h\mathbf{f}(t_n, \mathbf{y}_n)$$

$$\mathbf{k}_1 = h\mathbf{f}\left(t_n + \frac{h}{2}, \mathbf{y}_n + \frac{\mathbf{k}_0}{2}\right)$$

$$\mathbf{k}_2 = h\mathbf{f}\left(t_n + \frac{h}{2}, \mathbf{y}_n + \frac{\mathbf{k}_1}{2}\right)$$

$$\mathbf{k}_3 = h\mathbf{f}(t_n + h, \mathbf{y}_n + \mathbf{k}_2)$$

$$\mathbf{y}_{n+1} = \mathbf{y}_n + \frac{1}{6}(\mathbf{k}_0 + 2\mathbf{k}_1 + 2\mathbf{k}_2 + \mathbf{k}_3)$$

---

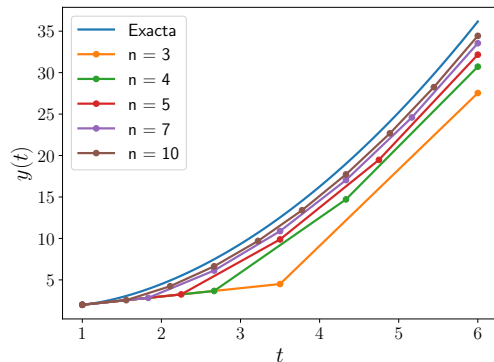
```

2 import numpy as np
3 import matplotlib.pyplot as plt
4 plt.style.use('../utils/classes.mplstyle')
5
6 def f(t, y):
7     return 3 * t - y / t
8
9 def y_exacta(t):
10     return t**2 + 1 / t
11
12 def euler(f, t_0, T, n, y_0):
13     h = (T - t_0) / (n - 1)
14     ts = t_0 + np.arange(n) * h
15     ys = np.zeros(n)
16     y = y_0
17     for n, t in enumerate(ts):
18         ys[n] = y
19         y += h * f(t, y)
20     return ts, ys
21
22 t_e = np.linspace(1, 6, 100)
23 plt.plot(t_e, y_exacta(t_e), label='Exacta')
24 for n in [3, 4, 5, 7, 10]:
25     t, y = euler(f, 1, 6, n, 2)
26     plt.plot(t, y, '-.', label=f"n = {n}")

```

---

Euler hacia adelante



---

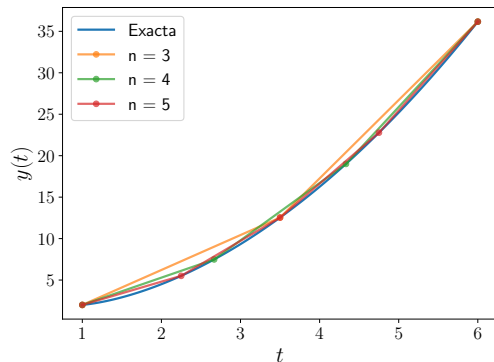
```

12 def rk4(f, t_0, T, n, y_0):
13     h = (T - t_0) / (n - 1)
14     ts = t_0 + np.arange(n) * h
15     ys = np.zeros(n)
16     y = y_0
17     for n, t in enumerate(ts):
18         ys[n] = y
19         k0 = h * f(t, y)
20         k1 = h * f(t + h/2, y + k0 / 2)
21         k2 = h * f(t + h/2, y + k1 / 2)
22         k3 = h * f(t + h, y + k2)
23         y += (k0 + 2 * k1 + 2 * k2 + k3) / 6
24     return ts, ys
25
26 t_e = np.linspace(1, 6, 100)
27 plt.plot(t_e, y_exacta(t_e), label='Exacta')
28 for n in [3, 4, 5]:
29     t, y = rk4(f, 1, 6, n, 2)
30     plt.plot(t, y, '-.', label=f"n = {n}", alpha=0.7)

```

---

## Runge-Kutta de cuarto orden



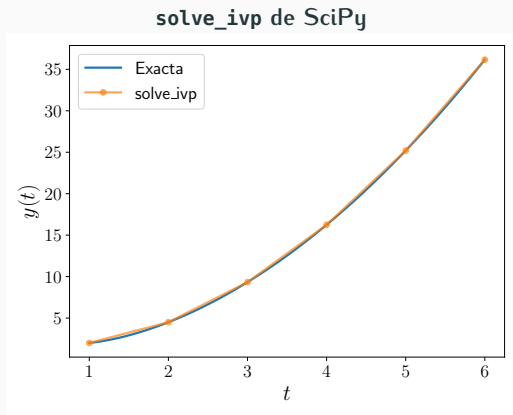
---

```

12 t_e = np.linspace(1, 6, 100)
13 plt.plot(t_e, y_exacta(t_e), label='Exacta')
14
15 from scipy.integrate import solve_ivp
16
17 sol = solve_ivp(f, [1, 6], [2], method='RK45',
18     t_eval=[1, 2, 3, 4, 5, 6])
19
20 plt.plot(sol.t, sol.y[0], '.-', label="solve_ivp",
21     alpha=0.7)
22
23 plt.xlabel(r"$t$")
24 plt.ylabel(r"$y(t)$")
25 plt.legend()
26 plt.tight_layout()
27 plt.savefig("solve_ivp.pdf")

```

---



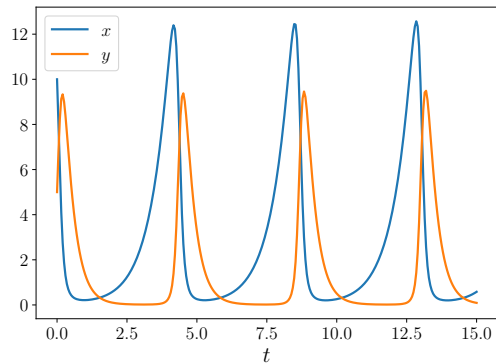
## Sistema Lotka-Volterra

$$\frac{dx}{dt} = \alpha x - \beta xy$$

$$\frac{dy}{dt} = \delta xy - \gamma y$$

$$x(0) = x_0, y(0) = y_0$$

```
1 #!/usr/bin/env python3
2 import numpy as np
3 import matplotlib.pyplot as plt
4 from scipy.integrate import solve_ivp
5 plt.style.use('../utils/classes.mplstyle')
6
7 def LV(t, z, a, b, d, g):
8     x, y = z
9     return [a * x - b * x * y, d * x * y - g * y]
10
11 sol = solve_ivp(LV, [0, 15], [10, 5],
12     args=(1.5, 1, 1, 3), dense_output=True)
13
14 t = np.linspace(0, 15, 300)
15 z = sol.sol(t)
16 plt.plot(t, z.T[:,0], label=r'$x$')
17 plt.plot(t, z.T[:,1], label=r'$y$')
18 plt.xlabel(r"$t$")
19 plt.legend()
20 plt.tight_layout()
21 plt.savefig("lv.pdf")
```



- ▶ R.L. Burden, D.J. Faires y A.M. Burden. *Análisis numérico*. 10.<sup>a</sup> ed. Mexico: Cengage Learning, 2017. Capítulo 5.
- ▶ B. Bradie. *A Friendly Introduction to Numerical Analysis*. New Jersey, United States: Pearson Education Inc., 2006. Capítulo 7.
- ▶ A. Gezerlis. *Numerical Methods in Physics With Python*. Cambridge, United Kingdom: Cambridge University Press, 2020. doi: [10.1017/9781108772310](https://doi.org/10.1017/9781108772310). Capítulo 8.
- ▶ J. Kiusalaas. *Numerical Methods in Engineering with Python 3*. Cambridge, United Kingdom: Cambridge University Press, 2013. Capítulo 7.