Método de Euler

Ecuaciones diferenciales ordinarias: primer orden, segundo orden y sistemas

Profesor: Diego Valencia Enríquez

Universidad Mariana

8 de octubre de 2025

Ruta de la sesión

1. Motivación: aplicaciones en Ingeniería Civil
2. Método de Euler (teoría y error)
3. Euler para: **(i) 1er orden**, **(ii) 2do orden**, **(iii) sistemas acoplados de 1er orden**, **(iv) sistemas de 2do orden**
4. Ejercicios resueltos paso a paso
5. Dos ejemplos aplicados (Hidráulica y Estructuras)
6. Código Python de referencia

Aplicaciones en Ingeniería Civil

- **Hidráulica**: variación temporal del tirante en un tanque, curva de vaciado, tránsito de ondas en canales (linealizado).
- **Transporte**: advección–dispersión 1D (reducción temporal en un punto).
- **Geotecnia**: consolidación unidimensional (modelo lumped), asentamientos dependientes del tiempo.
- **Estructuras**: oscilador SDOF con amortiguamiento (dinámica sísmica simplificada).
- **Ambiental**: decaimiento de contaminantes de primer orden.

La idea clave: aproximar la solución avanzando con la **pendiente local** ($y' = f(t, y)$) usando un paso h.

Método de Euler (idea)

Para el problema de valor inicial (PVI):

$$y'(t) = f(t, y), \qquad y(t_0) = y_0.$$

El método de Euler (explícito) usa:

$$\boxed{y_{n+1} = y_n + h\, f(t_n, y_n)}, \qquad t_{n+1} = t_n + h.$$

- **Orden** local: 2 ($O(h^2)$), **orden global**: 1 ($O(h)$).
- **Estabilidad**: condicionada por h y la rigidez del problema.
- **Interpretación geométrica**: aproximación lineal de $y(t)$ en cada paso.

Error y elección de paso

- Error local de truncamiento: $\tau_n = \dfrac{y(t_{n+1}) - y(t_n)}{h} - f(t_n, y(t_n)) = O(h)$.
- Error global: $E = O(h)$. Disminuye linealmente con h.
- Paso de tiempo típico: $h = \dfrac{t_{\text{fin}} - t_0}{N}$.
- **Regla práctica**: probar h, luego reducir $h/2$ y comparar.

> Si la solución "oscila" o diverge, disminuya h o use un método de mayor estabilidad (e.g., Euler implícito o métodos de Runge–Kutta).

Euler para EDO de primer orden: paso a paso

PVI: $y' = f(t, y), \quad y(t_0) = y_0$.

1. Especificar t_0, y_0, h, N.
2. Para $n = 0, \ldots, N-1$:
$$\begin{aligned} t_{n+1} &= t_n + h, \\ y_{n+1} &= y_n + h\, f(t_n, y_n). \end{aligned}$$
3. Reportar (t_n, y_n).

Ejemplo base: $y' = -ky$, $y(0) = y_0 \Rightarrow y_{n+1} = y_n(1 - kh)$. Estabilidad cuando $0 < h < 2/k$.

Ejercicio 1 (1er orden): decaimiento

PVI: $y' = -0{,}8\,y$, $y(0) = 1$. Paso $h = 0{,}5$, 4 pasos.

$$
\begin{aligned}
y_1 &= 1 + 0{,}5(-0{,}8 \cdot 1) = 0{,}6,\\
y_2 &= 0{,}6 + 0{,}5(-0{,}8 \cdot 0{,}6) = 0{,}36,\\
y_3 &= 0{,}36 + 0{,}5(-0{,}8 \cdot 0{,}36) = 0{,}216,\\
y_4 &= 0{,}216 + 0{,}5(-0{,}8 \cdot 0{,}216) = 0{,}1296.
\end{aligned}
$$

Comparación exacta: $y(t) = e^{-0{,}8t} \Rightarrow y(2) = e^{-1{,}6} \approx 0{,}2019$.
Con $h = 0{,}5$ se *subestima*. Disminuir h mejora el resultado.

Euler para EDO de segundo orden

Idea: reducir a sistema de primer orden.
Dada $y'' = g(t, y, y')$, defina $v = y'$. Entonces:

$$\begin{cases} y' = v, \\ v' = g(t, y, v). \end{cases}$$

Euler:

$$\begin{aligned} y_{n+1} &= y_n + h\, v_n, \\ v_{n+1} &= v_n + h\, g(t_n, y_n, v_n). \end{aligned}$$

Ejercicio 2 (2do orden): oscilador sin amortiguamiento

Modelo: $y'' + \omega^2 y = 0$, $\omega = 2\pi$ (1 Hz).
Condiciones: $y(0) = 1$, $y'(0) = 0$. Paso $h = 0{,}01$.

$$
\begin{aligned}
g(t, y, v) &= -\omega^2 y, \\
y_{n+1} &= y_n + h\, v_n, \\
v_{n+1} &= v_n - h\, \omega^2 y_n.
\end{aligned}
$$

Comentario: Euler explícito introduce *crecimiento/decadencia* artificial en energía. Para dinámica estructural, preferir pasos pequeños o métodos más estables (Euler semi-implícito, Newmark, RK).

Euler explícito: primeras 6 iteraciones (h=0.01)

$\omega = 2\pi$. Comparación con la solución exacta $y(t) = \cos(\omega t)$, $v(t) = -\omega \sin(\omega t)$

n	t_n	y_n (Euler)	v_n (Euler)	$y(t_n)$ exacta	$v(t_n)$ exacta
0	0.00	1.000000	0.000000	1.000000	-0.000000
1	0.01	1.000000	-0.394784	0.998027	-0.394524
2	0.02	0.996052	-0.789568	0.992115	-0.787492
3	0.03	0.988156	-1.182794	0.982287	-1.177352
4	0.04	0.976329	-1.572903	0.968583	-1.562565
5	0.05	0.960600	-1.958342	0.951057	-1.941611

Cuadro: Euler explícito vs. solución exacta en los primeros 6 pasos.

Se observa ligera **sobreestimación de la amplitud** (p. ej., $y_{0,05}$ Euler $= 0{,}960600$ vs. exacta $= 0{,}951057$) y **deriva de fase**. Reducir h o usar esquemas más estables (Euler semi-implícito, RK2/RK4, Newmark) atenúa estos efectos.

Sistemas acoplados de primer orden

Sistema: $\mathbf{y}' = \mathbf{f}(t, \mathbf{y})$, $\mathbf{y} \in \mathbb{R}^m$.
Euler componente a componente:

$$y_{n+1}^{(k)} = y_n^{(k)} + \Delta t\, f_k(t_n, \mathbf{y}_n), \qquad k = 1, \ldots, m.$$

Ejemplo (tanques acoplados): El tanque 1 recibe un caudal de entrada constante Q_{in} (por ejemplo, agua proveniente de una bomba o una tubería principal).
Este tanque descarga hacia el tanque 2 a través de un orificio cuya descarga se modela con la ley de Torricelli ($Q \sim C\sqrt{h}$).
A su vez, el tanque 2 evacúa el fluido hacia el exterior, también con una ley de descarga proporcional a $\sqrt{h_2}$.

$$h_1' = \frac{1}{A_1}(Q_{in} - C\sqrt{h_1}),$$
$$h_2' = \frac{1}{A_2}(C\sqrt{h_1} - C\sqrt{h_2}).$$

Con $A_1 = A_2$, mismo C, y datos $h_1(0), h_2(0)$, se iteran ambas ecuaciones con Euler.

Sistemas acoplados de primer orden: tanques en serie

Modelo:

$$h_1' = \frac{1}{A_1}\left(Q_{in} - C\sqrt{h_1}\right),$$
$$h_2' = \frac{1}{A_2}\left(C\sqrt{h_1} - C\sqrt{h_2}\right).$$

Datos (ejemplo numérico): $A_1 = A_2 = 1{,}0,\ C = 0{,}1,\ Q_{in} = 0{,}05,\quad h_1(0) = 0{,}50,\ h_2(0) = 0{,}20.$

Paso temporal: $\Delta t = 0{,}5$ s.

Euler (explícito) componente a componente:

$$\begin{aligned}
h_{1,n+1} &= h_{1,n} + \Delta t\ \frac{1}{A_1}\left(Q_{in} - C\sqrt{h_{1,n}}\right),\\
h_{2,n+1} &= h_{2,n} + \Delta t\ \frac{1}{A_2}\left(C\sqrt{h_{1,n}} - C\sqrt{h_{2,n}}\right),\\
t_{n+1} &= t_n + \Delta t.
\end{aligned}$$

Interpretación: el tanque 1 recibe Q_{in} y descarga a 2; el tanque 2 recibe del 1 y descarga al exterior. El acople es vía $\sqrt{h}$.

Euler explícito: 6 pasos de iteración ($\Delta t = 0{,}5$ s)

$$f_1(h_1) = Q_{in} - C\sqrt{h_1}, \qquad f_2(h_1, h_2) = C\sqrt{h_1} - C\sqrt{h_2}.$$

n	t_n	$h_{1,n}$	$h_{2,n}$	$f_1(h_{1,n})$	$f_2(h_{1,n}, h_{2,n})$	$h_{1,n+1}$	$h_{2,n+1}$
0	0.00	0.500000	0.200000	$-0{,}020711$	$0{,}025989$	0.489645	0.212995
1	0.50	0.489645	0.212995	$-0{,}019975$	$0{,}023823$	0.479657	0.224906
2	1.00	0.479657	0.224906	$-0{,}019257$	$0{,}021833$	0.470029	0.235823
3	1.50	0.470029	0.235823	$-0{,}018559$	$0{,}019997$	0.460749	0.245821
4	2.00	0.460749	0.245821	$-0{,}017879$	$0{,}018298$	0.451810	0.254970
5	2.50	0.451810	0.254970	$-0{,}017217$	$0{,}016722$	0.443202	0.263332

Cuadro: Evolución de alturas h_1, h_2 (m) con Euler explícito.

Paso ilustrativo $n = 0 \to 1$:

$$\begin{aligned}
f_1(h_{1,0}) &= 0{,}05 - 0{,}1\sqrt{0{,}5} = -0{,}020710678,\\
f_2(h_{1,0}, h_{2,0}) &= 0{,}1\sqrt{0{,}5} - 0{,}1\sqrt{0{,}2} = 0{,}025989319,\\
h_{1,1} &= 0{,}5 + 0{,}5(-0{,}020710678) = 0{,}489644661,\\
h_{2,1} &= 0{,}2 + 0{,}5(0{,}025989319) = 0{,}212994659.
\end{aligned}$$

Tendencia: h_1 desciende por descarga hacia 2; h_2 aumenta. Para mayor precisión, reduzca Δt o use un método de mayor orden (RK2/RK4).

Sistemas MDOF de segundo orden

Modelo dinámico general:

$$\mathbf{M}\,\ddot{\mathbf{u}} + \mathbf{C}\,\dot{\mathbf{u}} + \mathbf{K}\,\mathbf{u} = \mathbf{f}(t).$$

- $\mathbf{u}(t)$: desplazamientos nodales (grados de libertad).
- $\mathbf{M}, \mathbf{C}, \mathbf{K}$: matrices de masa, amortiguamiento y rigidez.
- $\mathbf{f}(t)$: vector de cargas externas.

Reducción a sistema de primer orden:

$$\begin{cases} \dot{\mathbf{u}} = \mathbf{v}, \\ \dot{\mathbf{v}} = \mathbf{M}^{-1}\big(\mathbf{f}(t) - \mathbf{C}\mathbf{v} - \mathbf{K}\mathbf{u}\big). \end{cases}$$

Un sistema MDOF representa, por ejemplo, un edificio de n pisos, donde cada piso se modela como un grado de libertad horizontal.

Sistemas de segundo orden (p.ej. MDOF)

Estructuras MDOF: $\mathbf{M}\,\ddot{\mathbf{u}} + \mathbf{C}\,\dot{\mathbf{u}} + \mathbf{K}\,\mathbf{u} = \mathbf{f}(t)$.
Defina $\mathbf{v} = \dot{\mathbf{u}}$. Sistema de 1er orden:

$$\begin{cases} \dot{\mathbf{u}} = \mathbf{v}, \\ \dot{\mathbf{v}} = \mathbf{M}^{-1}\big(\mathbf{f}(t) - \mathbf{C}\mathbf{v} - \mathbf{K}\mathbf{u}\big). \end{cases}$$

Euler:

$$\mathbf{u}_{n+1} = \mathbf{u}_n + h\,\mathbf{v}_n,$$
$$\mathbf{v}_{n+1} = \mathbf{v}_n + h\,\mathbf{M}^{-1}\big(\mathbf{f}(t_n) - \mathbf{C}\mathbf{v}_n - \mathbf{K}\mathbf{u}_n\big).$$

Nota: estabilidad puede exigir h muy pequeño; para análisis sísmico, suele preferirse Newmark-β o métodos explícitos condicionalmente estables con Δt crítico.

Euler explícito para MDOF

Esquema de Euler:

$$\mathbf{u}_{n+1} = \mathbf{u}_n + h\,\mathbf{v}_n,$$
$$\mathbf{v}_{n+1} = \mathbf{v}_n + h\,\mathbf{M}^{-1}\big(\mathbf{f}(t_n) - \mathbf{C}\mathbf{v}_n - \mathbf{K}\mathbf{u}_n\big).$$

Datos ejemplo: 2 GDL

$$\mathbf{M} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad \mathbf{C} = \begin{bmatrix} 0{,}1 & 0 \\ 0 & 0{,}1 \end{bmatrix}, \quad \mathbf{K} = \begin{bmatrix} 20 & -10 \\ -10 & 10 \end{bmatrix}.$$

Condiciones iniciales: $\mathbf{u}(0) = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$, $\mathbf{v}(0) = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$, $\mathbf{f}(t) = \mathbf{0}$.

Paso: $h = 0{,}01$, iterar 6 pasos.

Euler explícito: 6 pasos de iteración

n	t_n	$u_n^{(1)}$	$u_n^{(2)}$	Observación
0	0.00	1.0000	0.0000	Estado inicial
1	0.01	1.0000	0.0000	$v = 0$, aún no cambia
2	0.02	0.9980	0.0001	Empieza a descender $u^{(1)}$
3	0.03	0.9941	0.0004	Acoplamiento transfiere energía a $u^{(2)}$
4	0.04	0.9883	0.0010	Oscilación inicia
5	0.05	0.9807	0.0018	
6	0.06	0.9712	0.0029	Crece respuesta en el 2° GDL

Cuadro: Primeras 6 iteraciones con Euler explícito en un sistema de 2 GDL.

Se observa cómo el primer grado de libertad comienza a oscilar y el segundo responde por **acoplamiento dinámico**.

Conclusión del ejemplo MDOF

- El método de Euler permite aproximar la respuesta de sistemas MDOF, pero introduce **errores de amplitud y fase**.
- Para estabilidad, el paso h debe ser muy pequeño.
- En práctica estructural se prefieren métodos como **Newmark**-β, **Wilson**-θ o Runge–Kutta.
- El ejemplo muestra el principio básico: un edificio de varios grados de libertad puede ser modelado como un sistema dinámico matricial y resuelto numéricamente.

Ejemplo aplicado A (Hidráulica): vaciado de tanque

Modelo (Torricelli): $\dfrac{dh}{dt} = -\dfrac{a}{A}\sqrt{2gh},\ h(0) = h_0.$
Euler con paso h_t:

$$h_{n+1} = h_n - h_t \frac{a}{A}\sqrt{2g\, h_n}, \qquad h_n \geq 0.$$

Datos ejemplo: $h_0 = 2\ \mathrm{m},\ a = 1\ \mathrm{cm}^2,\ A = 1\ \mathrm{m}^2,\ g = 9{,}81.$
Cuidados: imponer $h_{n+1} = \max(h_{n+1}, 0)$.
Comparar con solución analítica $\left(\sqrt{h(t)} = \sqrt{h_0} - \frac{a}{2A}\sqrt{2g}\, t\right)$.

Ejemplo aplicado B (Estructuras): SDOF amortiguado

Modelo: $m\ddot{u} + c\dot{u} + ku = F_0 \sin(\omega t)$.
Reducción: $u' = v, \quad v' = \frac{1}{m}(F_0 \sin \omega t - cv - ku)$.

$$
\begin{aligned}
u_{n+1} &= u_n + h\, v_n, \\
v_{n+1} &= v_n + h\, \frac{1}{m}\big(F_0 \sin(\omega t_n) - c v_n - k u_n\big).
\end{aligned}
$$

Datos ejemplo: $m = 1,\ c = 0{,}05,\ k = (2\pi)^2,\ F_0 = 1,\ \omega = 2\pi,\ u(0) = 0,\ v(0) = 0,\ h = 0{,}001.$

Ejercicios propuestos (para clase/taller)

1. **1er orden**: $y' = y - t^2 + 1,\ y(0) = 0{,}5,\ h = 0{,}2$ hasta $t = 1$. Compare con exacta.
2. **2do orden**: $y'' + 4y = 0,\ y(0) = 0,\ y'(0) = 1,\ h = 0{,}05$. Grafique $y(t)$.
3. **Sistema 1er orden**: depósitos en serie (ver diapositiva "Sistemas 1er orden"), simule 0–300 s con $h = 0{,}5$.
4. **Sistema 2do orden**: MDOF de 2 grados con $\mathbf{M} = \mathrm{diag}(m_1, m_2)$, $\mathbf{K}$ acoplada; use cargas nulas y condiciones iniciales no triviales.

Código Python: Método de Euler (implementación básica)

```python
import numpy as np
import matplotlib.pyplot as plt

def euler(f, t0, y0, h, N):
    """
    Método de Euler para resolver y' = f(t, y)

    Parámetros:
    f: función f(t, y)
    t0: tiempo inicial
    y0: condición inicial y(t0)
    h: paso de tiempo
    N: número de pasos

    Retorna:
    t: array de tiempos
    y: array de soluciones
    """
    # Inicializar arrays
    t = np.zeros(N+1)
    y = np.zeros(N+1, dtype=float)

    # Condición inicial
    t[0] = t0
    y[0] = y0

    # Iteración del método de Euler
    for n in range(N):
        y[n+1] = y[n] + h * f(t[n], y[n])
        t[n+1] = t[n] + h

    return t, y
```

Listing 1: Implementación del método de Euler para EDOs escalares

Código Python: Método de Euler para sistemas de EDOs

```python
def euler_system(F, t0, Y0, h, N):
    """
    Método de Euler para sistemas de EDOs
    Resuelve Y' = F(t, Y) donde Y es un vector

    Parámetros:
    F: función vectorial F(t, Y)
    t0: tiempo inicial
    Y0: condición inicial Y(t0)
    h: paso de tiempo
    N: número de pasos

    Retorna:
    t: array de tiempos
    Y: matriz de soluciones (cada fila es un estado)
    """
    # Inicializar arrays
    t = np.zeros(N+1)
    Y = np.zeros((N+1, len(Y0)))

    # Condición inicial
    t[0] = t0
    Y[0] = np.array(Y0, dtype=float)

    # Iteración del método de Euler
    for n in range(N):
        Y[n+1] = Y[n] + h * F(t[n], Y[n])
        t[n+1] = t[n] + h

    return t, Y
```

Listing 2: Extensión del método para sistemas de ecuaciones

Código Python: Ejemplos y definición de problemas

```python
# Ejemplo 1: Decaimiento exponencial
# EDO: y' = -k y, y(0) = 1
# Solución analítica: y(t) = exp(-k t)
k = 0.8
f1 = lambda t, y: -k * y

# Resolver con método de Euler
t1, y1 = euler(f1, t0=0.0, y0=1.0, h=0.05, N=200)

# Ejemplo 2: Oscilador armónico
# EDO de 2do orden: y'' + w^2 y = 0
# Convertido a sistema de 1er orden:
# y' = v
# v' = -w^2 y
w = 2 * np.pi  # Frecuencia angular

def F_oscillator(t, Y):
    """
    Función vectorial para el oscilador armónico
    Y[0] = y (posición)
    Y[1] = v (velocidad)
    """
    dydt = Y[1]           # y' = v
    dvdt = -w**2 * Y[0]   # v' = -w^2 y
    return np.array([dydt, dvdt])

# Condiciones iniciales: y(0)=1, v(0)=0
t2, Y2 = euler_system(F_oscillator,
                      t0=0.0,
                      Y0=[1.0, 0.0],
                      h=0.001,
                      N=5000)
```

Listing 3: Definición de los problemas a resolver

Código Python: Visualización de resultados

```python
# Crear figura con subplots
plt.figure(figsize=(12, 8))

# Subplot 1: Decaimiento exponencial
plt.subplot(2, 2, 1)
plt.plot(t1, y1, 'b-', linewidth=2,
         label="Euler numérico")
# Solución analítica para comparación
y_analitica = np.exp(-k * t1)
plt.plot(t1, y_analitica, 'r--',
         label="Solución analítica")
plt.xlabel('Tiempo (t)')
plt.ylabel('y(t)')
plt.legend()
plt.grid(True, alpha=0.3)
plt.title('Decaimiento Exponencial')

# Subplot 2: Posición del oscilador
plt.subplot(2, 2, 2)
plt.plot(t2, Y2[:, 0], 'r-',
         label="Posición (y)")
plt.xlabel('Tiempo (t)')
plt.ylabel('y(t)')
plt.legend()
plt.grid(True, alpha=0.3)
plt.title('Oscilador: Posición')
```

Listing 4: Código: Subplots 1 y 2

```python
# Subplot 3: Velocidad del oscilador
plt.subplot(2, 2, 3)
plt.plot(t2, Y2[:, 1], 'g-',
         label="Velocidad (v)")
plt.xlabel('Tiempo (t)')
plt.ylabel('v(t)')
plt.legend()
plt.grid(True, alpha=0.3)
plt.title('Oscilador: Velocidad')

# Subplot 4: Espacio de fases
plt.subplot(2, 2, 4)
plt.plot(Y2[:, 0], Y2[:, 1], 'purple',
         label="Trayectoria")
plt.xlabel('Posición (y)')
plt.ylabel('Velocidad (v)')
plt.legend()
plt.grid(True, alpha=0.3)
plt.title('Espacio de Fases')

# Ajustar layout y mostrar
plt.tight_layout()
plt.show()
```

Listing 5: Código: Subplots 3 y 4

Comentarios finales y recomendaciones

- Euler es sencillo y útil para **introducción y prototipado**, pero su error global es $O(h)$.
- En sistemas oscilatorios/estáticos, verifique estabilidad: pasos pequeños o métodos alternativos (Euler implícito, RK2/RK4, Newmark).
- Siempre haga **pruebas de convergencia**: compare $h, h/2, h/4$.