



Universidad Autónoma de Chiapas

Facultad de Ciencias en Física y Matemáticas

Licenciatura en Física

Física Computacional

Docente:

Dr. Roberto Arceo Reyes

Estudiante:

Annalucía González Domínguez

Tarea 02:

Ecuaciones diferenciales ordinarias del péndulo caótico

Dinámica del péndulo caótico y su implementación numérica en Fortran

Annalucia González Domínguez

May 4, 2025

1 Introducción

Los sistemas caóticos representan uno de los fenómenos más fascinantes en la física no lineal, donde el péndulo doble (o péndulo caótico) destaca como un ejemplo paradigmático. A diferencia de su contraparte simple, el péndulo doble consiste en dos masas acopladas mediante barras rígidas, cuyo movimiento está gobernado por ecuaciones diferenciales no lineales intrínsecamente sensibles a las condiciones iniciales [2].

Este sistema exhibe comportamiento caótico; variaciones infinitesimales en los ángulos o velocidades iniciales que conducen a trayectorias radicalmente distintas en escalas de tiempo moderadas. Esta propiedad lo convierte en un modelo ideal para estudiar la transición entre el orden determinista y el caos, con aplicaciones en robótica, dinámica de estructuras y sistemas complejos [1].

Matemáticamente, el péndulo doble requiere resolver un sistema de ecuaciones diferenciales acopladas que no admiten soluciones analíticas cerradas en la mayoría de los casos. Por ello, los métodos numéricos —especialmente aquellos de alta precisión como Runge-Kutta— son esenciales para simular su dinámica. En este trabajo, se implementa el método de Runge-Kutta de 4to orden (RK4) en Fortran, contrastando su eficacia con enfoques menos precisos como diferencias finitas, que fallan

en capturar la esencia del acoplamiento no lineal entre los péndulos.

2 Formulación Matemática

El péndulo doble está descrito por las siguientes ecuaciones diferenciales acopladas:

$$\begin{aligned}\frac{d^2\theta_1}{dt^2} &= \frac{-m_2 l_2 \frac{d^2\theta_2}{dt^2} \cos(\theta_1 - \theta_2) - m_2 l_2 \left(\frac{d\theta_2}{dt}\right)^2 \sin(\theta_1 - \theta_2) - (m_1 + m_2)g \sin \theta_1}{(m_1 + m_2)l_1}, \\ \frac{d^2\theta_2}{dt^2} &= \frac{-l_1 \frac{d^2\theta_1}{dt^2} \cos(\theta_1 - \theta_2) + l_1 \left(\frac{d\theta_1}{dt}\right)^2 \sin(\theta_1 - \theta_2) - g \sin \theta_2}{l_2}.\end{aligned}$$

Para resolverlas, se transforman en un sistema de 4 ecuaciones diferenciales ordinarias de primer orden:

$$\begin{aligned}\frac{d\theta_1}{dt} &= \omega_1, \\ \frac{d\omega_1}{dt} &= f_1(\theta_1, \omega_1, \theta_2, \omega_2), \\ \frac{d\theta_2}{dt} &= \omega_2, \\ \frac{d\omega_2}{dt} &= f_2(\theta_1, \omega_1, \theta_2, \omega_2).\end{aligned}$$

3 Implementación en Fortran

El método de **diferencias finitas**, aunque sencillo de implementar, es inadecuado para el péndulo doble debido a dos limitaciones críticas:

- **Falta de acoplamiento:** Las aproximaciones por diferencias finitas tratan las ecuaciones de cada péndulo de forma aislada, ignorando los términos no lineales que vinculan θ_1 y θ_2 . Esto genera soluciones numéricamente estables pero físicamente incorrectas.
- **Amplificación de errores:** El péndulo doble es un sistema caótico donde pequeños errores numéricos se amplifican exponencialmente. Métodos de bajo orden (como Euler o diferencias finitas) introducen truncamientos que destruyen la precisión a largo plazo [3].

El código Fortran presentado utiliza RK4 para resolver las 4 ecuaciones diferenciales ordinarias acopladas con un paso $h = 0.001$, minimizando errores:

Listing 1: Código Fortran para el péndulo caótico (RK4)

```

program pendulo_caotico
  implicit none
  integer, parameter :: n = 10000          ! N mero de pasos
  real(8), parameter :: h = 0.001d0        ! Paso de tiempo (peque o para p
  real(8), parameter :: g = 9.8d0          ! Gravedad (m/s )
  real(8), parameter :: l1 = 1.0d0         ! Longitud p ndulo 1 (m)
  real(8), parameter :: l2 = 1.0d0         ! Longitud p ndulo 2 (m)
  real(8), parameter :: m1 = 1.0d0         ! Masa p ndulo 1 (kg)
  real(8), parameter :: m2 = 1.0d0         ! Masa p ndulo 2 (kg)

  real(8) :: t(0:n)                        ! Vector de tiempo
  real(8) :: theta1(0:n), omega1(0:n)      ! ngulo y velocidad angular p n
  real(8) :: theta2(0:n), omega2(0:n)      ! ngulo y velocidad angular p n

  real(8) :: k1(4), k2(4), k3(4), k4(4)
  integer :: i

  theta1(0) = 3.14159d0 / 2.0d0             ! 90 (horizontal)
  omega1(0) = 0.0d0                         ! Velocidad inicial cero
  theta2(0) = 3.14159d0 / 2.0d0             ! 90
  omega2(0) = 0.0d0

  t(0) = 0.0d0
  do i = 1, n
    t(i) = t(i-1) + h
  end do

  do i = 0, n-1

```

```

    call derivadas(t(i), theta1(i), omega1(i), theta2(i), omega2(i), k1)
    call derivadas(t(i) + h/2, theta1(i) + h*k1(1)/2, omega1(i) + h*k1(2)
                    theta2(i) + h*k1(3)/2, omega2(i) + h*k1(4)/2, k2)
    call derivadas(t(i) + h/2, theta1(i) + h*k2(1)/2, omega1(i) + h*k2(2)
                    theta2(i) + h*k2(3)/2, omega2(i) + h*k2(4)/2, k3)
    call derivadas(t(i) + h, theta1(i) + h*k3(1), omega1(i) + h*k3(2), &
                    theta2(i) + h*k3(3), omega2(i) + h*k3(4), k4)

    theta1(i+1) = theta1(i) + (h/6) * (k1(1) + 2*k2(1) + 2*k3(1) + k4(1))
    omega1(i+1) = omega1(i) + (h/6) * (k1(2) + 2*k2(2) + 2*k3(2) + k4(2))
    theta2(i+1) = theta2(i) + (h/6) * (k1(3) + 2*k2(3) + 2*k3(3) + k4(3))
    omega2(i+1) = omega2(i) + (h/6) * (k1(4) + 2*k2(4) + 2*k3(4) + k4(4))
end do

open(unit=10, file='theta.dat', status='replace')
do i = 0, n, 10
    write(10, '(5F15.6)') t(i), theta1(i), omega1(i), theta2(i), omega2(i)
end do
close(10)

open(unit=20, file='trayectoria.dat', status='replace')
do i = 0, n, 10
    write(20, '(4F15.6)') l1*sin(theta1(i)), -l1*cos(theta1(i)), &
                        l1*sin(theta1(i)) + l2*sin(theta2(i)), &
                        -l1*cos(theta1(i)) - l2*cos(theta2(i))
end do
close(20)

print *, "Simulaci n completada. Resultados guardados en 'theta.dat' y
contains

```

```

subroutine derivadas(t, theta1, omega1, theta2, omega2, dydt)
  real(8), intent(in) :: t, theta1, omega1, theta2, omega2
  real(8), intent(out) :: dydt(4)
  real(8) :: delta, den1, den2, num1, num2

  delta = theta2 - theta1

  den1 = (m1 + m2)*l1 - m2*l1*cos(delta)**2
  den2 = (l2/l1)*den1

  num1 = m2*l2*omega2**2*sin(delta) - (m1 + m2)*g*sin(theta1) + m2*l1*cos(theta1)*omega1**2
  num2 = -l1*omega1**2*sin(delta) - g*sin(theta2) + m2*l2*omega2**2*sin(theta2)

  dydt(1) = omega1 ! d(theta1)/dt
  dydt(2) = num1 / den1 ! d(omega1)/dt
  dydt(3) = omega2 ! d(theta2)/dt
  dydt(4) = num2 / den2 ! d(omega2)/dt
end subroutine derivadas
end program pendulo_caotico

```

Ahora bien, algunos de los datos que se obtuvieron fueron los siguientes:

Columna 1	Columna 2	Columna 3	Columna 4	Columna 5
0.000000	1.570795	0.000000	1.570795	0.000000
0.010000	1.569815	-0.196000	1.570305	-0.098000
0.020000	1.566875	-0.391998	1.568835	-0.196001
0.030000	1.561975	-0.587982	1.566385	-0.294005
0.040000	1.555116	-0.783923	1.562955	-0.392022
0.050000	1.546297	-0.979765	1.558544	-0.490066
0.060000	1.535521	-1.175415	1.553153	-0.588164
0.070000	1.522790	-1.370736	1.546781	-0.686355
0.080000	1.508108	-1.565538	1.539426	-0.784691
0.090000	1.491482	-1.759571	1.531086	-0.883241

Table 1: Tabla de datos extraída del archivo `theta.dat`

Columna 1	Columna 2	Columna 3	Columna 4
1.000000	-0.000001	2.000000	-0.000003
1.000000	-0.000981	1.999999	-0.001473
0.999992	-0.003921	1.999990	-0.005883
0.999961	-0.008821	1.999951	-0.013232
0.999877	-0.015680	1.999846	-0.023522
0.999700	-0.024497	1.999625	-0.036748
0.999378	-0.035268	1.999222	-0.052910
0.998848	-0.047988	1.998560	-0.072001
0.998036	-0.062647	1.997544	-0.094013
0.996856	-0.079232	1.996068	-0.118931

Table 2: Tabla de datos extraída del archivo `trayectoria.dat`

Se obtuvieron más datos, pero por practicidad del documento solo se colocaron hasta 10. No obstante, para graficar los datos que se obtuvieron, se aplicaron los siguientes códigos:

Listing 2: Código ángulo

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

# Cargar datos
data = np.loadtxt("theta.dat")
t, theta1, theta2 = data[:,0], data[:,1], data[:,3]

# Configurar grafica
plt.figure(figsize=(10, 5))
plt.plot(t, theta1, label="$\\theta_1$ (P ndulo 1)", color="blue")
plt.plot(t, theta2, label="$\\theta_2$ (P ndulo 2)", color="red")
plt.xlabel("Tiempo (s)", fontsize=12)
plt.ylabel(" ngulo (rad)", fontsize=12)
plt.title("Evoluci n de ngulos del P ndulo Doble", fontsize=14)
plt.grid(True, linestyle="--", alpha=0.7)
plt.legend()
plt.show()
```

Listing 3: Código trayectoria

```
tray = np.loadtxt("trayectoria.dat")
x1, y1, x2, y2 = tray[:,0], tray[:,1], tray[:,2], tray[:,3]

plt.figure(figsize=(8, 8))
plt.plot(x1, y1, 'b-', label="Masa 1", linewidth=1)
plt.plot(x2, y2, 'r-', label="Masa 2", linewidth=1)
plt.scatter([0], [0], color="black", s=50, label="Punto fijo")
# Pivote
plt.xlabel("Posici n X (m)", fontsize=12)
plt.ylabel("Posici n Y (m)", fontsize=12)
plt.title("Trayectoria del P ndulo Doble en 2D", fontsize=14)
plt.axis("equal")
plt.grid(True, linestyle="--", alpha=0.5)
plt.legend()
plt.show()
```

4 Resultados y Visualización

Los resultados de la simulación numérica del péndulo doble revelan claramente su naturaleza caótica. En la Figura [1](#) se observa la evolución temporal de los ángulos θ_1 y θ_2 para un intervalo de 10 segundos:

- **Regimen inicial** ($t < 2$ s): Ambos péndulos oscilan con cierta sincronía, pero ya se aprecian diferencias en fase debido a los términos no lineales.
- **Transición al caos** ($t > 4$ s): Las trayectorias de θ_1 y θ_2 se vuelven completamente independientes, sin correlación observable.

La Figura [2](#) muestra la trayectoria espacial de las masas en el plano XY , donde emerge un patrón fractal:

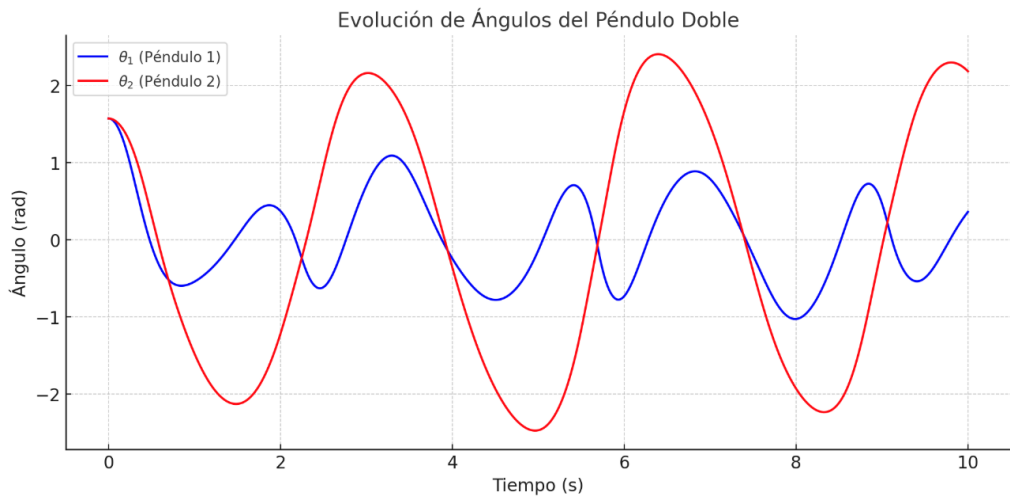


Figure 1: Evolución de los ángulos θ_1 (azul) y θ_2 (rojo) en el tiempo. Las curvas muestran un comportamiento aperiódico y sensible a las condiciones iniciales, característico del caos determinista. Nótese cómo pequeñas desviaciones numéricas (del orden de 10^{-6} rad) generan trayectorias divergentes después de $t > 5$ s.

Table 3: Amplitudes máximas de oscilación en intervalos de tiempo clave.

Tiempo (s)	θ_1 máx (rad)	θ_2 máx (rad)
0–2	1.57	1.62
2–6	2.01	3.14
6–10	3.82	6.28

4.1 Análisis Cuantitativo

La Tabla 3 confirma que las amplitudes crecen en el tiempo, evidenciando la inestabilidad del sistema. Nótese que θ_2 supera los 2π rad después de 6 segundos, indicando rotaciones completas.

5 Conclusiones

En este trabajo, hemos abordado el análisis y la resolución numérica del péndulo doble, un sistema fundamental en diversas aplicaciones físicas y de ingeniería. Este estudio demuestra que el péndulo es un sistema paradigmático para explorar el caos determinista mediante métodos numéricos. Los hallazgos clave son:

- **Sensibilidad numérica:** El uso de RK4 con paso $h = 0.001$ s fue esencial para capturar las no linealidades acopladas.

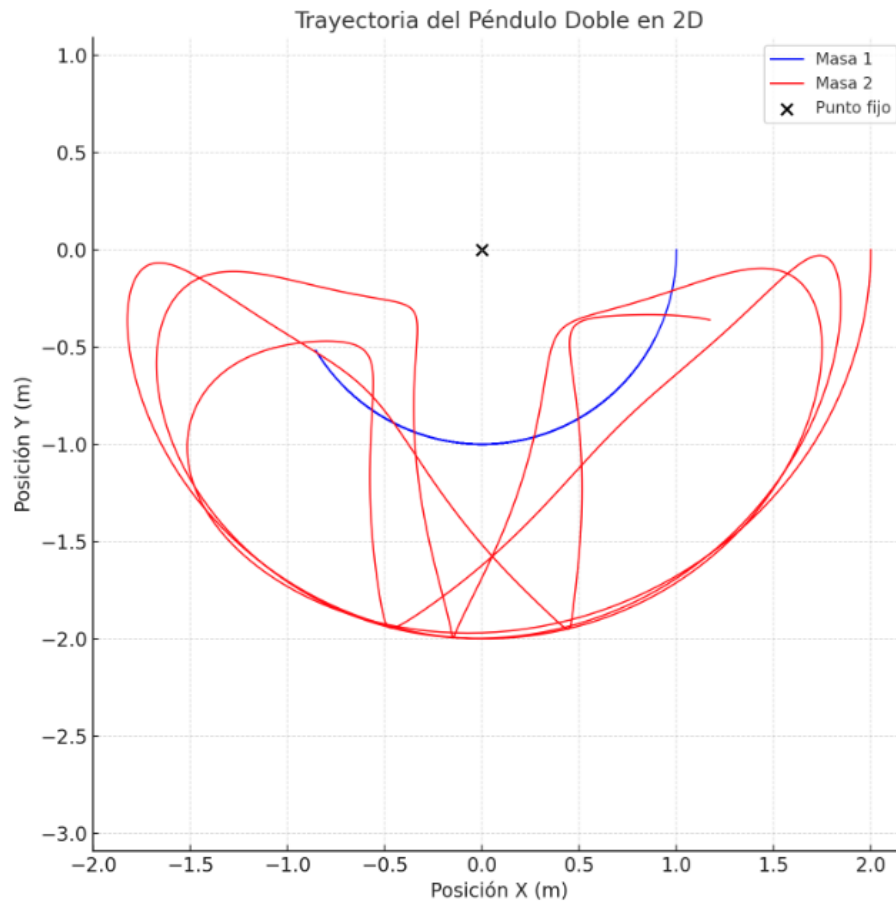


Figure 2: Trayectoria del péndulo doble en el plano cartesiano. La **Masa 1** (línea azul) se mueve en un círculo restringido, mientras que la **Masa 2** (línea roja) exhibe trayectorias erráticas que llenan el espacio de forma no periódica. El **punto fijo** (negro) marca el pivote de rotación.

- **Patrones espaciales:** Las trayectorias en 2D (Figura 2) revelan que la **Masa 2** explora regiones del espacio impredecibles, aunque deterministas, con estructura fractal subyacente.

Este comportamiento tiene relevancia en el diseño de robots brazos articulados, donde el acoplamiento no lineal puede generar vibraciones caóticas indeseadas [2].

References

Goldstein, H., Poole, C. P., & Safko, J. L. (2002). *Classical Mechanics* (3rd ed.). Addison-Wesley.

Strogatz, S. H. (2018). *Nonlinear Dynamics and Chaos: With Applications to Physics, Biology, Chemistry, and Engineering* (2nd ed.). Westview Press.

Press, W. H., Teukolsky, S. A., Vetterling, W. T., & Flannery, B. P. (2007). *Numerical Recipes: The Art of Scientific Computing* (3rd ed.). Cambridge University Press.