Práctica 4-
El modelo de Lorenz

Luis Sánchez Izquierdo | miércoles 14 de marzo de 2018

Introducción

En esta práctica voy a analizar el sistema del Atractor de Lorenz cuya respuesta es caótica.

Lo que voy a hacer es:

- Obtener una trayectoria 3D
- Obtener las secciones de Poincaré en los ejes x=0 e y=0 para diferentes condiciones iniciales
- Extra 1: exponente de lyapunov
- Extra 2: el atractor de Lorentz conducido
- Extra 3: animación 3D ¿?
- Extra 4: computación paralela ¿?
- Extra 5: atractores en función de r ¿?
- Extra 6: números aleatorios ¿?

Análisis teórico

1- El modelo de Lorenz

El modelo de Lorenz es un sistema que cuya evolución es caótica. Esto lo que quiere decir es que

Las ecuaciones del modelo de Lorenz surgieron por el estudio de la atmósfera. Lo que hizo era buscar la solución a las ecuaciones de Navier-Stokes para el problema de Rayleich-Bénard. Dicho problema plantea una situación en la que hay un fluido entre dos superficies horizontales infinitas sobre las que se crea un gradiente de temperatura y se pretende estudiar la convección.

El sistema tras ser modificado matemáticamente se acaba describiendo el sistema mediante tres variables:

$x(t) \propto$ Intensidad del movimiento convectivo

$y(t) \propto \Delta T$ entre las corrientes subida y bajada

$z(t) \propto$ distorsión del perfil de temperatura (respecto del caso lineal)

Y se obtiene la dinámica del sistema mediante el siguiente sistema de ecuaciones:

$$\begin{cases} \dfrac{dx}{dt} = \sigma \cdot (y - x) \\ \dfrac{dy}{dt} = r \cdot x - y - x \cdot z \\ \dfrac{dz}{dt} = x \cdot y - b \cdot x \end{cases}$$

En donde hay tres parámetros que son:

$$\sigma = \frac{difusividad\ viscosa}{difusividad\ térmica}$$

$$r = \Delta T = T_{placa\ superior} - T_{placa\ inferior}$$

$$b = Radio\ de\ los\ vórtices\ de\ convección$$

Típicamente se escoge que los valores de los parámetros sean:

$$\sigma = 10, \qquad b = \frac{8}{3}, \qquad r = variable$$

Que r sea variable es porque es el parámetro que usaremos para obtener los distintos regímenes de funcionamiento. Se ha comprobado que los distintos estados vienen dados por:

- $r < 1 \rightarrow estado\ estacionario$
- $1 < r < 4 \rightarrow flujo\ constante$
- $4 < r < 24 \rightarrow region\ intermedia$
- $24 < r \rightarrow flujo\ caótico$

El haber hecho tantas simplificaciones y haber llegado a una dinámica en función de x, y, z hace que no sepamos de forma intuitiva lo que le ocurre al sistema físicamente. Sin embargo, hay un par de consideraciones físicas que se pueden tener en cuenta como que:

$$\text{Si } sign(x) = sign(y) \rightarrow \text{el fluido sube}$$

$$\text{Si } z > 0 \rightarrow \text{Hay un gradiente fuerte cerca bordes}$$

Aun así, lo importante de este sistema es que actúa como un sistema caótico y eso es lo que queremos estudiar computacionalmente. El que sea caótico técnicamente lo que quiere decir es que el sistema es determinista pero impredecible. Se trata de un sistema determinista ya que dado unas condiciones iniciales la solución exacta es única. Sin embargo, es impredecible porque para una ligera variación de las condiciones iniciales se obtendrán variaciones muy significativas a lo largo de la evolución del sistema (al igual que pasa con el péndulo). Y físicamente nunca se va a poder tomar unas condiciones iniciales con perfecta exactitud, lo cual es necesario para el determinismo.

Aquí vale el método numérico de Euler siempre que $\Delta t \rightarrow 0 \approx 10^{-4}$ para que los errores sean menores que la pérdida de energía.

2- Implementación computacional.

En este sistema vale el método numérico de Euler siempre que $\Delta t \rightarrow 0 \approx 10^{-4}$ para que los errores sean menores que la pérdida de energía. Podríamos pensar que al ser un sistema cíclico lo ideal sería usar el método de Euler-Crömer, pero no es posible ya que este método se usa cuando las ecuaciones diferenciales son de segundo orden.

Consecuentemente, para resolver este sistema basta con definir la función de Lorenz:

```
14 def lorenz(x, y, z, s=10, r=26, b=8/3):
15     vx = s*(y - x)
16     vy = r*x - y - x*z
17     vz = x*y - b*z
18     return vx, vy, vz
```

Y realizar un bucle para calcular las posiciones a lo largo del tiempo, tras haber establecido unas condiciones iniciales:

```
# Crear los valores iniciales
x[0], y[0], z[0] = (1.,0.,0.)

# Obtener la evolución en el tiempo
for i in range(N):
    vx, vy, vz = lorenz(x[i], y[i], z[i])
    x[i+1] = x[i] + vx*dt
    y[i+1] = y[i] + vy*dt
    z[i+1] = z[i] + vz*dt
    t[i+1] = t[i] + dt
```

Resultados computacionales:

1- Representación de la trayectoria $\vec{r}(t) = \big(x(t), y(t), z(t)\big)$

Este apartado básicamente consiste en calcular la evolución temporal y representarla en $3D$. Para ello ejecuto el código anterior y lo represento en $3D$ con la librería:

```
from pylab import legend
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
```

Y el código:

```
fig = plt.figure(1)
ax = fig.gca(projection='3d')
ax.plot(x, y, z, lw=0.5, \
        label=("Condiciones iniciales: r=(%d, %d, %d)" % (x[0], y[0], z[0])))
ax.set_xlabel("X Axis")
ax.set_ylabel("Y Axis")
ax.set_zlabel("Z Axis")
ax.set_title("Lorenz Attractor")
plt.show()
legend()
```

Y lo que se obtiene es:

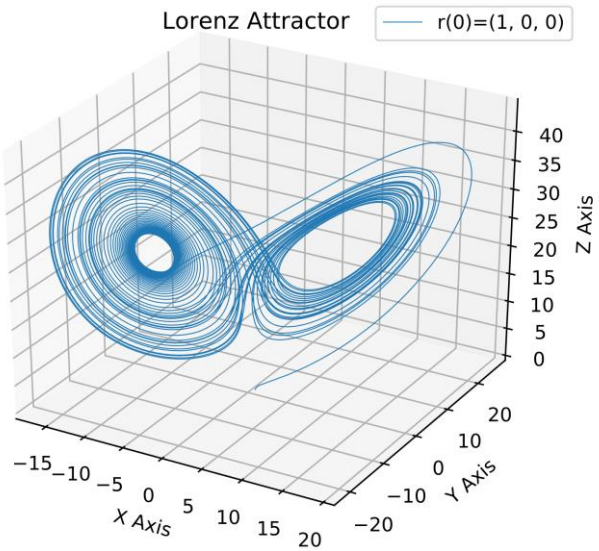


Figura 1- Trayectoria

2- Obtención del atractor

Para analizar el atractor lo que hay que hacer es calcular las secciones de Poincaré y analizar el comportamiento para distintas condiciones iniciales.

La sección de Poincaré se calcula en el espacio de las fases tomando un valor constante. En este caso el espacio de las fases viene dado por $x(y,z)$, $y(x,z)$ o bien $z(x,y)$. En este caso tomaremos las secciones de Poincaré en las que:

$$x = 0 \text{ ó bien } y = 0$$

Además, tomaré estas secciones de Poincaré para varios valores de condiciones iniciales distintas. Computacionalmente basta con, una vez calculadas las trayectorias, seleccionar aquellos datos que están próximos a cero:

```
dt = 1e-4
tol = 2*dt
```

```
###### en el eje x=0
yPoincare_x0 = []
zPoincare_x0 = []
for i in range(N):
    if abs(x[i]) < tol:
        yPoincare_x0.append(y[i])
        zPoincare_x0.append(z[i])
```

Tras ejecutar el código se obtienen los dos atractores para las distintas condiciones iniciales (ver las condiciones iniciales en la leyenda):

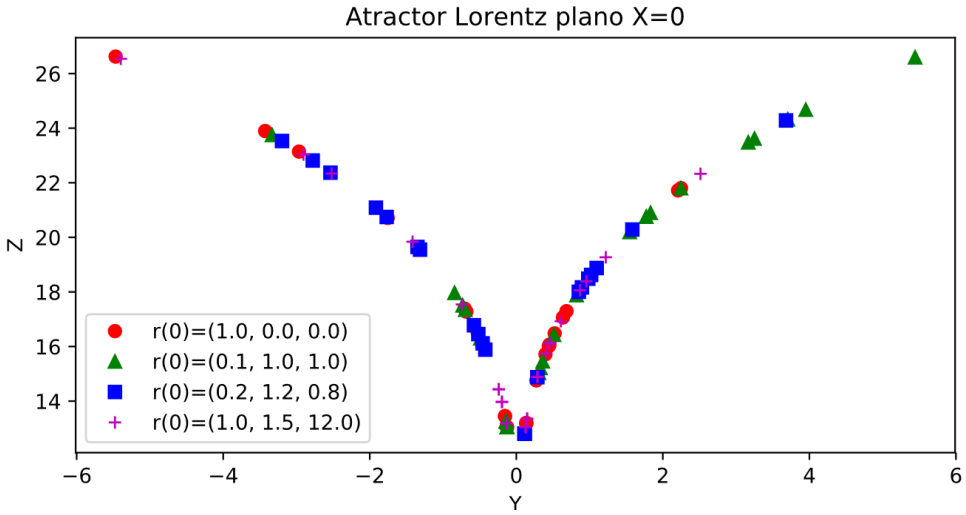


Figura 2- Atractor extraño en el plano x=0

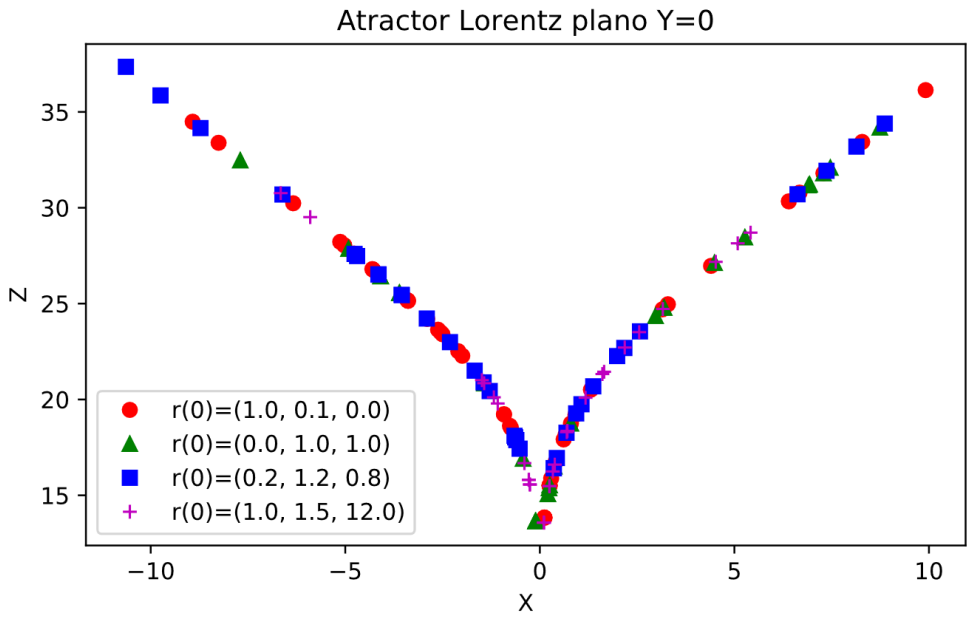


Figura 3- atractor extraño en el plano y=0

Como se puede observar, independientemente de las condiciones iniciales es sistema evolucionará y hará que siempre pase por la misma zona en el plano $x = 0$ o en el plano $y = 0$. Esto es lo que definimos como atractor extraño.

3- Extra 1: Análisis del Caos. Exponente del Lyapunov

En este apartado lo que quiero es analizar el caos del sistema. Este caos consiste básicamente en que bajo el establecimiento de condiciones iniciales ligeramente diferentes los efectos a largo plazo serán trayectorias totalmente diferentes. Esta es la componente de impredecibilidad que comentaba en el análisis teórico.

Para analizar esto es estudio el exponente de Lyapunov: representar $abs(x_1(t) - x_2(t))$ o $abs(y_1(t) - y_2(t))$ o $abs(z_1(t) - z_2(t))$en función del tiempo t en una gráfica logarítmica. La envolvente de la gráfica nos dará el valor de dicho exponente.

Las funciones $\vec{r}_1(t) = (x_1(t), y_1(t), z_1(t))$ y $\vec{r}_2(t) = (x_2(t), y_2(t), z_2(t))$ tienen condiciones iniciales dadas por:

$$\vec{r}_1(t = 0) = (1, 0, 0)$$

$$\vec{r}_2(t = 0) = \vec{r}_1(t = 0) + \vec{u}_{random}$$

En donde $|\vec{u}_{random}|$ es muy pequeño.

```
# Condiciones iniciales del sistema 1
x[0] = 1.
y[0] = 0.
z[0] = 0.

size2 = 0.05
# Condiciones iniciales del sistema 2
x1[0] = x[0] + size2 * (random.random()*2 - 1)
y1[0] = y[0] + size2 * (random.random()*2 - 1)
z1[0] = z[0] + size2 * (random.random()*2 - 1)
```

Por lo que se ve en el código, el módulo del vector es del orden de $|\vec{u}_{random}| \approx 0.05$

Tras ejecutar el código y representar los resultados se obtienen resultados parecidos al siguiente:

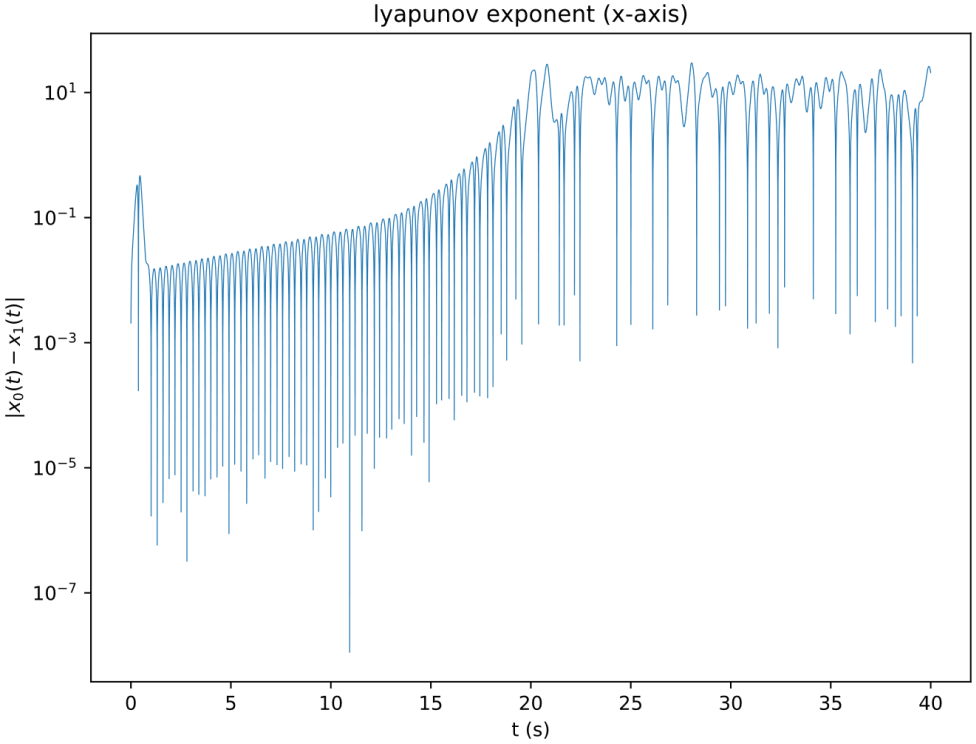


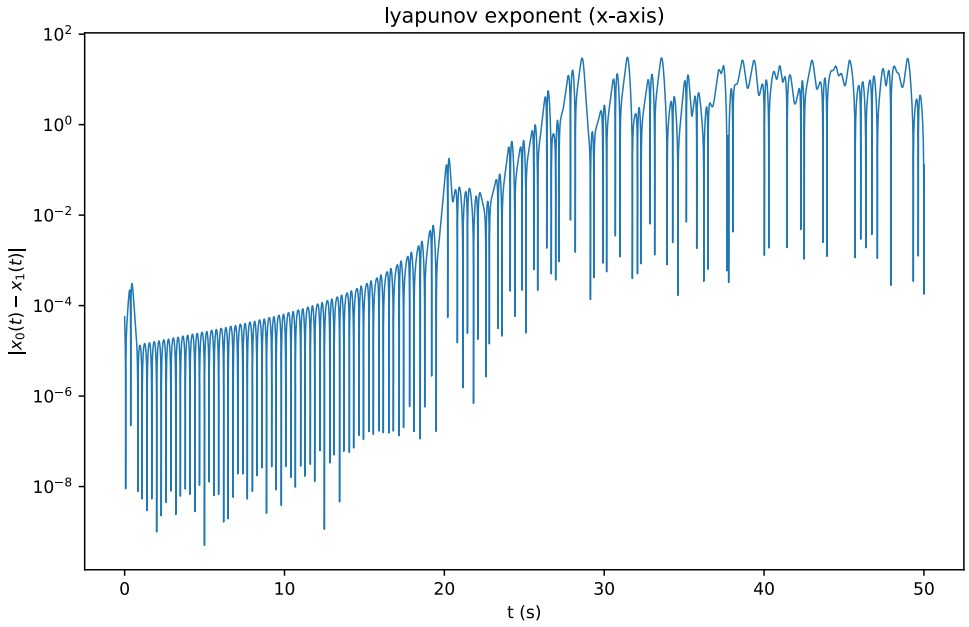
Figura 4- Exponente de Lyapunov. Movimiento impredecible

La explicación de esta gráfica es que inicialmente la variación en distancias a lo largo del $eje\ x$ podría ser de hasta 0.05 unidades, es decir, del orden de 10^{-2}. A medida que avanza el tiempo se la situación la diferencia va aumentando hasta que llega a un valor estable. La pendiente con la que va aumentando es el exponente de Lyapunov, que en este caso mide:

$$\gamma \approx \frac{10^{1} - 10^{-2}}{20 - 0} \approx 0.5 > 0$$

Aun así, se ve también que el aumento no es homogéneo, sino que la pendiente varía. El momento en que se estabiliza en un valor en torno a 10^{1} es porque ha llegado al máximo nivel de desorden. El movimiento de $\vec{r}(t)$ está acotado entre ciertos valores. De forma aproximada con la Figura 1 se ve que dichos valores son $x \in (-20,20)$ $y \in (-20,20)$ $z \in (0,40)$ por tanto la máxima distancia de separación que se puede obtener es del orden de 10^{1}, que es valor estable que se alcanza.

En el caso en el que es tamaño de $\vec{u}_{random}$ sea más pequeño se obtiene un comportamiento similar. Para $|\vec{u}_{random}| \approx 0.0001$ se obtiene la siguiente gráfica:



Ahora el exponente de Lyapunov será:

$$\gamma \approx \frac{10^1 - 10^{-4}}{30 - 0} \approx 0.3 > 0$$

Que es prácticamente igual que en el caso anterior, lo que es normal de esperar ya que el sistema se desordena prácticamente igual de rápido.

4- Extra 2: El atractor de Lorenz guiado.

En este apartado extra analizo una propiedad curiosa del Atractor de Lorenz: el atractor de Lorenz guiado. Se trata de ver cómo evolucionan dos sistemas con condiciones iniciales diferentes, pero con el mismo valor de la coordenada x:

$$\vec{r}_1(t) = \big(x_1(t), y_1(t), z_1(t)\big)$$

$$\vec{r}_2(t) = \big(x_1(t), y_2(t), z_2(t)\big)$$

El primer sistema evoluciona libremente pero el segundo sistema está condicionado a que $x_2(t) = x_1(t)$. Computacionalmente esto se implementa en el momento en el que se calcula la evolución temporal:

```
54 for i in range(N):
55     vx1, vy1, vz1 = lorenz(x1[i], y1[i], z1[i])
56     x1[i+1] = x1[i] + vx1*dt
57     y1[i+1] = y1[i] + vy1*dt
58     z1[i+1] = z1[i] + vz1*dt
59 
60     vx2, vy2, vz2 = lorenz(x2[i], y2[i], z2[i])
61     x2[i+1] = x1[i+1]
62     y2[i+1] = y2[i] + vy2*dt
63     z2[i+1] = z2[i] + vz2*dt
64 
65     t[i+1] = t[i] + dt
```

En azul está subrayado lo más importante: $x_2(t) = x_1(t)$

Los resultados se pueden representar en la gráfica que muestre la proyección de los puntos (y, z) ya que la información de los puntos (x) nos da igual porque sabemos que por definición serán iguales. Para la simulación he puesto que las condiciones iniciales son aleatorias entre un rango de $(-20, 20)$.

Los resultados son los siguientes:

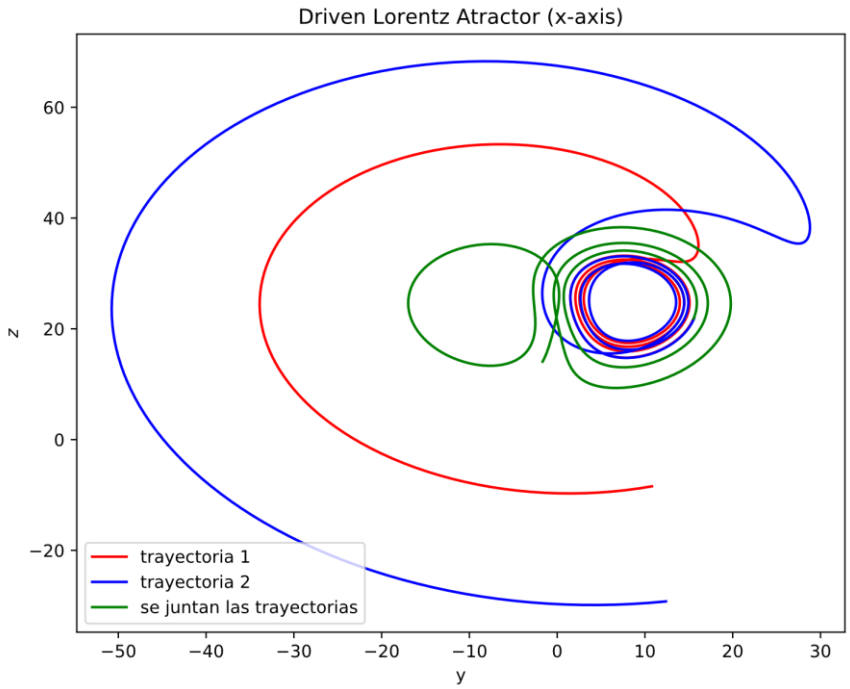


Figura 5- Modelo de Lorenz guiado (se muestra los primeros pasos de la trayectoria)

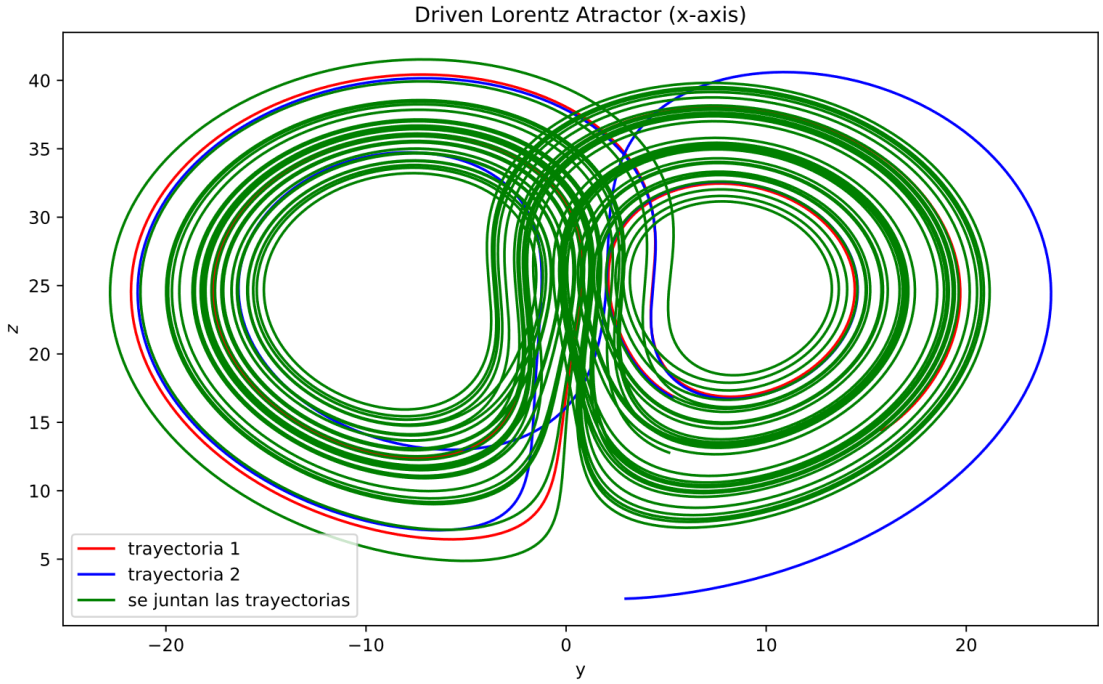


Figura 6- Modelo de Lorenz guiado (se muestran más pasos)