ILI-286: Computación Científica II
Laboratorio #3
Simulando Númericamente Sistemas Dinámicos

Claudio Torres Axel Símonsen Martín Villanueva

30 de noviembre 2015

Parte 1: Simulando Sistemas Caóticos

El Atractor de Lorenz

Un sistema dinámico caótico se define como aquel que ante pequeñas perturbaciones iniciales a las variables que lo determinan, produce resultados inesperados o imposibles de predecir. Uno de los primero y más notables ejemplos es el **Atractor de Lorenz**, que fue derivado de un modelo simplificado de convección en la atmósfera terrestre. Este corresponde a un sistema acoplado de tres ecuaciones, de primer orden y no lineales

$$\frac{\mathrm{d}x}{\mathrm{d}t} = \sigma(y - x), \tag{1}$$
$$\frac{\mathrm{d}y}{\mathrm{d}t} = x(\rho - z) - y, \tag{2}$$
$$\frac{\mathrm{d}z}{\mathrm{d}t} = xy - \beta z. \tag{3}$$

que describen la trayectoria de una partícula en el tiempo. Tal sistema está totalmente determinado por las condiciones iniciales (x_0, y_0, z_0). Sin embargo Lorenz descubrió que para ciertas configuraciones de los parámetros (σ, ρ, β) las trayectorias se vuelven caóticas y muestran interesantes propiedades. Una de estas configuraciones (y la que ocuparemos en esta experiencia) es $(\sigma = 10, \rho = 28, \beta = 8/3)$.

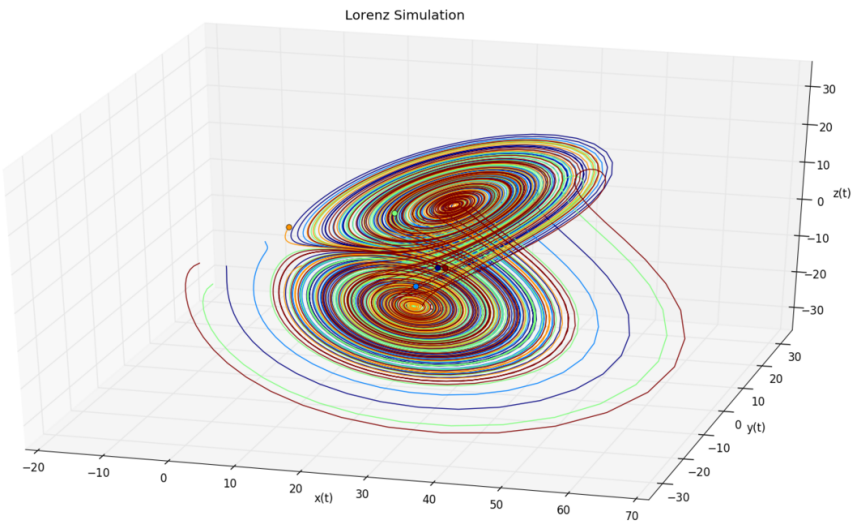

Figure 1: Simulación Númerica del Atractor de Lorenz

No es intuitivo a simple vista, pero con tales configuraciones las trayectorias no forman ciclos acotados ni alcanzan estados estacionarios. Este sistema de Lorenz es totalmente sensible a las condiciones iniciales; Dos estados iniciales no importa cuán cercanos ellos sean tendrán que diverger, usualmente más temprano que tarde. Este resultado hizo concluir a Lorenz que no es posible predecir con mucha anticipación las condiciones climáticas futuras, dadas las condiciones actuales.

Simulación

Como tal sistema no posee solución análitica, los modelaremos numéricamente:

a) Implementar los métodos de Euler, Punto Medio y RK4. Las funciones deben tener (al menos) la misma interfaz que **odeint**: $method(func, y_0, t, h)$. Estas deben soportar sistemas dinámicos, operar de modo vectorizado y retornar una matriz de dimensiones $(\text{len}(t), \text{len}(y))$ con la solución del sistema a lo largo del tiempo.

b) Ejecutar los métodos implementados, con la misma condición inicial (a elección) y mostrar los resultados obtenidos con función *animate_lorenz*() que se provee. Utilizar tiempo final $T_f = 100$ y ancho de malla h el máximo (aproximado) que permita funcionar a todos lo métodos.

c) Realizar un profiling con `timeit` y `memit` para las tres ejecuciones anteriores pero con $T_f = 1000$. Comente sus resultados.

d) Realizar un análisis de la estabilidad con los gráficos respectivos para los tres métodos (basarse en el material visto en clases: *U3W2_AdvancedApplications.ipynb*). Probar distintos valores de h, analice y comente sus resultados.

e) Realizar una implementación de los tres métodos anteriores, que permita resolver (simular) simultáneamente N sistemas de Lorenz (trayectorias). Las funciones a implementar deben tener como estructura $parallel\_method(func, y_0, t, h)$ donde el único parámetros que cambia es y_0, que es una matriz de condiciones iniciales, de 3 filas (una por cada variable) y $N_{trayectorias}$ columnas. Como resultado debe retornar una matriz de dimensiones $(\text{len}(t), 3, N_{trayectorias})$ con la solución de cada trayectoria en el tiempo.

f) Realizar una simulación (por cada método) de dos trayectorias con condiciones iniciales a elección, pero muy similares: La distancia euclidiana entre ellas debe ser menor a 0.00001. Utilizar tiempo final $T_f = 50$ y ancho de malla h el máximo (aproximado) que permita funcionar a todos lo métodos. Comente y compare sus resultados.

g) Realizar la misma simulación pero para 20 trayectorias, todas ellas con condiciones iniciales que se encuentren dentro de una esfera de radio $r = 0.00001$ con centro a elección. Utilice sólo RK4. Comente sus resultados.

Observación: Para visualizar correctamente las animaciones, no se debe configurar el notebook con `%matplotlib inline`, sino que intentar con las opciones `%matplotlib notebook` o `%matplotlib nbagg` ocupando PYTHON3 de preferencia. En su defecto omitir estas configuraciones, y adjuntar el resultado final de la animación.

Parte 2: BVP de orden superior

En este sección se quiere resolver un BVP de orden superior, por medio del método de *Diferencias Finitas*. La ecuación en cuestión corresponde a la siguiente

$$\begin{aligned} &y^{(4)}(x) = -y(x) \\ &\text{BC} : \{y(0) = 0,\ y(2) = 1,\ y'(2) = -1,\ y''(2) = 1\} \end{aligned} \tag{4}$$

dónde BC corresponden a las cuatro condiciones de frontera, necesarias para resolver el problema.

a) Transformar el BVP en un sistema de IVP's acoplado.

b) Proponer una forma de aproximar la cuarta derivada y discretizar la ecuación 4.

c) Implementar una función que tenga (al menos) como parámetro la cantidad de puntos en la malla a utilizar, y que compute la solución de $y(x)$ por diferencias finitas en esos puntos. **Observación:** *Queda estrictamente prohibido computar inversas. El método debe aprovechar las propiedades del sistema lineal que resulta.*

d) Pruebe su algoritmo para 5 valores de $N \in \{20, 10000\}$ a su elección, de tal modo que se aprecien diferencias. Grafique los resultados obtenidos y comente.

Instrucciones:

(a) La estructura del laboratorio es la siguiente:

  (a) Título, nombre(s) de estudiante(s), email(s) y rol(s).
  (b) Introducción.
  (c) Desarrollo y análisis de resultados.
  (d) Conclusiones.
  (e) Referencias.

(b) El laboratorio debe ser realizado en `IPython` notebook (con `Python2` o `Python3`).

(c) Se evaluará la correcta utilización de librerias `NumPy` y `SciPy`, así como la correcta implementación de algoritmos vectorizados cuando se indique.

(d) El archivo de entrega debe denominarse Lab3-apellido1-apellido2.tar.gz, y debe contener un directorio con todos los archivos necesarios para ejecutar el notebook, junto con un archivo README indicando explícitamente la versión utilizada de Python.

(e) El descuento por día de atraso será de 30 puntos, con un máximo de 1 día de atraso. No se recibirán entregas después de este día.

(f) El trabajo es personal o en grupos de a 2, no se permite compartir código, aunque sí se sugiere discutir aspectos generales con sus compañeros. Copias exactas serán sancionadas con nota 0.

(g) El no seguir estas instrucciones, implica descuentos en su nota obtenida.