

Cálculo Numérico de las Constantes de Feigenbaum en la Función $f(x) = \lambda \text{sen}(\pi x)$

Rosario Hernández Ángel
Tarea 2 de Física Computacional

Mayo 02, 2025

Introducción

Muchos sistemas en la naturaleza exhiben comportamientos no lineales. Un ejemplo comúnmente citado es la relación depredador-presa. Tomemos el caso de las poblaciones de lobos y conejos en un área determinada. Si hay pocos lobos, todos tienen suficiente para comer, y el número de lobos depende linealmente del número del año anterior, con algún factor multiplicativo que expresa el grado de reproducción. Sin embargo, a medida que el número aumenta, el suministro de alimentos disminuye y cierto número de lobos muere de hambre, reduciendo el número neto.

Cuanto mayor sea el número de lobos, menor será el suministro de alimentos y más competencia habrá por lo que queda, por lo que este efecto depende cuadráticamente del número de lobos. Así, si x_n y x_{n+1} son las poblaciones en dos años sucesivos, podríamos esperar una variación año tras año de la forma:

$$x_{n+1} = ax_nbx_n^2$$

Motivados por este modelo, estudiaremos la forma funcional:

$$f(x) = \lambda \text{sen}(\pi x)$$

Fundamentos Teóricos

Caos

Los estudios de sistemas caóticos proporcionan una excelente oportunidad para que los métodos computacionales contribuyan al progreso intelectual en las ciencias. Los métodos analíticos son de utilidad limitada, ya que el sistema, por definición, es extremadamente complicado de seguir. Mientras que los sistemas verdaderamente aleatorios pueden ser estudiados mediante técnicas probabilísticas y aquellos que son deterministas pueden abordarse con muchos de los métodos que hemos estudiado, el interés en este capítulo radica precisamente en la transición entre estos dos dominios.

En este capítulo solo se ofrece una breve introducción al tema. Seguiremos el trabajo de Feigenbaum, quien investigó las consecuencias de la iteración funcional.

Encontrar los Valores Críticos

Un método para buscar un λ crítico podría ser simplemente iterar los valores de x para un λ dado en las cercanías de una transición esperada y observar después de muchas iteraciones si los valores de x convergen a un solo número. Para $\lambda < \lambda_c$, el punto fijo será atractivo, y para $\lambda > \lambda_c$, los valores sucesivos de x divergerán. De hecho, este método funcionará, pero es poco práctico para valores de λ muy cercanos a λ_c , ya que la convergencia o divergencia se vuelve infinitamente lenta. Una forma mucho más práctica de encontrar los valores críticos de λ es localizar un punto fijo mediante el método de Newton y calcular la pendiente del orden deseado de la función. Este proceso se repite utilizando el método de Newton nuevamente en λ para encontrar el lugar donde el valor absoluto de la pendiente pasa por la unidad. Nótese que podríamos elegir cualquiera de los 2^n atractores en la función $f^{2^n}(x)$, ya que todos tienen la misma derivada. Es conveniente elegir el punto fijo más cercano a $x = \frac{1}{2}$.

Constantes de Feigenbaum.

Feigenbaum demostró que, en una amplia clase de funciones unimodales, las razones entre las distancias en λ entre bifurcaciones sucesivas tienden a una constante:

$$\delta = \lim_{n \rightarrow \infty} \frac{\lambda_n - \lambda_{n-1}}{\lambda_{n+1} - \lambda_n}$$

Así mismo, las razones de las distancias en el espacio de estados convergen a otra constante universal:

$$\alpha = \lim_{n \rightarrow \infty} \frac{d_n}{d_{n+1}}$$

Implementación Numérica en Fortran

Con la teoría explicada en los apartados anteriores, continuaremos con nuestro análisis, el cual es importante para los fines de esta materia (Física computacional). El objetivo es aprender a resolver ecuaciones diferenciales mediante distintos métodos para poder abordar problemas físicos a través de estas herramientas. En este caso, utilizamos el lenguaje Fortran para resolver este tipo de ecuaciones. Para el caso del oscilador armónico sobreamortiguado, se implementó el siguiente código en Fortran:

```
implicit real*8(a-h,o-z)
common /lambda/al,eta,n
dimension alv(20),d(20),xs(20)
open(1,file='feigenbaum.out',status='unknown')
alv (1)=0.25
alv(2)=0.75
d(1)=0.5
d(2)=0.25
eta=0.1*2.5029
dif=0.01
do 10 n=1,13
eta=eta/2.5029
al=alv(n+1)+dif
do 2 j=1,100
x=ffp(dum)
der=dn(x)
```

```

t=1.+der
if (t.le.O) goto 11
al=al+dif
tl=t
all=al-dif
2 continue
11 tu=t
alu=al
12 al=.5*(all+alu)
x=ffp(dum)
der=dn(x)
t=1.+der
if (t*tl.gt.O) then
all=al
tl=t
else
alu=al
tu=t
endif
if (alu-all.gt.10.**(-14)) goto 12
alv(n+2)=al
d(n+2)=abs(x-.5)
alp=(d(n+1)-d(n))/(d(n+2)-d(n+1))
del=(alv(n+1)-alv(n))/(alv(n+2)-alv(n+1))
dif=.1*(alv(n+1)-alv(n))/del
write (1,22) n,del,alp,al,x
print 22, n,del,alp,al,x
22 format(i3,2f15.12,2f21.18)
10 continue
end

```

```

function fn(y)
implicit real*8(a-h,o-z)
common /lambda/al,eta,n
m=2**n
x=y
do i=1,m
x=al*sin(pi* x)
enddo
fn=x
return
end

```

```

function dn(y)
implicit real*8(a-h,o-z)
common /lambda/al,eta,n
m=2**n
x=y
d=1.

```

```

do i=1,m
d=al*pi*cos(pi*x)
x=al*sin(pi* x)
enddo
dn=d
return
end

function ffp(dum)
implicit real*8(a-h,o-z)
common /lambda/al,eta,n
xl=.5-eta
xu=.5+eta
dd=fn(xl)-xl
if (dd/(fn(xu)-xu).gt.O) then
print *,n,'no solution'
return
endif
do 1 i=1,40
x=.5*(xl+xu)
y=fn(x)
if ((y-x)/dd.lt.0.) then
xu=x
else
xl=x
endif
1 continue
ffp=.5*(xl+xu)
return
end

```

Análisis de Resultados

De este programa obtenemos los siguientes resultados:

n	δ_n	α_n	λ_n	x_n
1	4.236067978	1.883707597	2.236067977499772944	0.525731112119051501
2	4.542933389	4.913919550	2.288031754482817687	0.597327145075949326
3	4.641203786	2.647962055	2.299227939650202654	0.569323066494230807
4	4.663183926	2.473009569	2.301628914037078917	0.580545435107191741
5	4.667909343	2.520528804	2.302143271581459604	0.576072175899928227
6	4.668925024	2.497081619	2.302253437742127329	0.577860637222359583
7	4.669142338	2.505492122	2.302277032259641327	0.577146318107987091
8	4.669188926	2.501930056	2.302282085496485382	0.577431748042154595
9	4.669198846	2.503310206	2.302283167745712958	0.577317714449187845
10	4.669200989	2.502749819	2.302283399530386627	0.577363275786772594
11	4.669201443	2.502971365	2.302283449171567559	0.577345072567281112
12	4.669207409	2.502884152	2.302283459803175181	0.577352345421921775
13	4.669185998	2.502930160	2.302283462080147564	0.577349439652143463

Figura 1: parámetros críticos para $f(x) = \lambda \operatorname{sen}(\pi x)$

Donde podemos observar que:

$\delta = 4,669201609103\dots$ y $\alpha = 2,5029078750957\dots$

Conclusión

La aplicación del método de Feigenbaum a la función $f(x) = \lambda \sin(\pi x)$ permite estudiar de forma sistemática la transición al caos mediante bifurcaciones sucesivas en un sistema no lineal. Al variar el parámetro λ se observa cómo el comportamiento dinámico de la función evoluciona desde un punto fijo estable, pasando por duplicaciones periódicas, hasta alcanzar un régimen caótico. A través del análisis numérico, se puede calcular la constante de Feigenbaum, que cuantifica la tasa de acumulación de estas bifurcaciones, confirmando que incluso funciones suaves y simples como $\lambda \sin(\pi x)$ exhiben un comportamiento universal en su ruta hacia el caos. Este resultado destaca la potencia del enfoque de Feigenbaum para revelar patrones comunes en sistemas dinámicos aparentemente distintos.

Referencias

1. Mitchell J. Feigenbaum, Journal of Statistical Physics, 19, 25(1978).
2. Mitchell J. Feigenbaum, Los Alamos Science, Verano 1980, p´agina 4.