

# Cálculo Numérico de las Constantes de Feigenbaum en la Función $f(x) = \lambda x(1 - x^2)$

Brandon Miguel López Díaz

May 3, 2025

## Abstract

Este trabajo presenta un análisis numérico para obtener las constantes de Feigenbaum asociadas a la bifurcación de periodo en el sistema dinámico definido por la función  $f(x) = \lambda x(1 - x^2)$ . Utilizando un código en Fortran, se calculan las constantes delta y alfa de Feigenbaum a partir de una secuencia de valores críticos del parámetro  $\lambda$  donde ocurren bifurcaciones sucesivas. Se incluye una explicación teórica del fenómeno de bifurcación, los fundamentos del método de cálculo, un análisis de los resultados obtenidos y una discusión sobre la precisión numérica.

## 1 Introducción

Los sistemas dinámicos no lineales muestran comportamientos complejos que pueden incluir caos determinista. Una de las rutas más estudiadas hacia el caos es la duplicación de periodo, en la cual una sucesión de bifurcaciones lleva a una dinámica caótica. El físico Mitchell Feigenbaum descubrió que estas bifurcaciones ocurren a valores específicos del parámetro de control  $\lambda$ , y que las razones entre distancias sucesivas entre estos valores tienden a una constante universal: la delta de Feigenbaum.

El presente trabajo tiene como objetivo implementar un método numérico en Fortran para calcular estas constantes, aplicándolo a la función cúbica  $f(x) = \lambda x(1 - x^2)$ .

## 2 Fundamentos Teóricos

### 2.1 Bifurcaciones y Caos

Una bifurcación es un cambio cualitativo en la dinámica de un sistema al variar un parámetro. En ciertos sistemas, como las iteraciones de funciones no lineales, al aumentar el parámetro  $\lambda$ , el punto fijo se vuelve inestable y da paso a un ciclo de periodo 2, luego a uno de periodo 4, y así sucesivamente.

## 2.2 Constantes de Feigenbaum

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

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

Asimismo, las razones de las distancias en el espacio de estado convergen a otra constante universal:

$$\alpha = \lim_{n \rightarrow \infty} \frac{d_n}{d_{n+1}} \quad (2)$$

## 3 Metodología y Código

Se implementó un algoritmo en Fortran que:

- Estima los valores críticos de  $\lambda_n$  donde ocurren bifurcaciones.
- Calcula la posición de los puntos fijos de alta iteración usando el método de bisección.
- Calcula las derivadas y verifica estabilidad local.
- Obtiene las constantes  $\delta_n$  y  $\alpha_n$  a partir de diferencias finitas.

La función iterada es:

$$f(x) = \lambda x(1 - x^2) \quad (3)$$

Se inicia con un punto cercano a  $x = \frac{1}{\sqrt{3}}$  y se observa su evolución bajo  $2^n$  iteraciones.

## 4 Implementación Numérica en Fortran

### Programa Principal

```
1      implicit real*8(a-h,o-z)
2      common /lambda/al,eta,n
3      dimension alv(20),d(20),xs(20)
4      open(1,file='feigenbaum.out',status='unknown')
5      alv(1) = 1.0d0
6      alv(2) = 2.0d0
7      d(1) = 1.0d0 / sqrt(3.0d0)
8      d(2) = 0.233929985d0
9      eta = 0.1d0 * 2.5029d0
10     dif = 0.01d0
11     do 10 n = 1, 13
12         eta = eta / 2.5029d0
13         al = alv(n+1) + dif
14         do 2 j = 1, 100
15             x = ffp(dum)
16             der = dn(x)
17             t = 1.0d0 + der
18             if (t .le. 0.0d0) goto 11
19             al = al + dif
20             t1 = t
21             all = al - dif
22     continue
23     11 tu = t
24     alu = al
25     12 al = 0.5d0 * (all + alu)
26     x = ffp(dum)
27     der = dn(x)
28     t = 1.0d0 + der
29     if (t * t1 .gt. 0.0d0) then
30         all = al
31         t1 = t
32     else
33         alu = al
34         tu = t
35     endif
36     if (alu - all .gt. 1.0d-14) goto 12
37     alv(n+2) = al
38     d(n+2) = abs(x - (1.0d0 / sqrt(3.0d0)))
39     alp = (d(n+1) - d(n)) / (d(n+2) - d(n+1))
40     del = (alv(n+1) - alv(n)) / (alv(n+2) - alv(n+1))
41     dif = 0.1d0 * (alv(n+1) - alv(n)) / del
42     write (1, 22) n, del, alp, al, x
43     print 22, n, del, alp, al, x
44     22 format(i3, 2f15.12, 2f21.18)
45     10 continue
46     end
```

Listing 1: Cálculo de las constantes de Feigenbaum

## Función fn(y)

```
1      function fn(y)
2      implicit real*8(a-h,o-z)
3      common /lambda/al,eta,n
4      m = 2**n
5      x = y
6      do i = 1, m
7          x = al * x * (1.0d0 - x**2)
8      enddo
9      fn = x
10     return
11     end
```

Listing 2: Función fn(y) - Iteración de f(x)

## Función dn(y)

```
1      function dn(y)
2      implicit real*8(a-h,o-z)
3      common /lambda/al,eta,n
4      m = 2**n
5      x = y
6      d = 1.0d0
7      do i = 1, m
8          d = d * al * (1.0d0 - 3.0d0 * x**2)
9          x = al * x * (1.0d0 - x**2)
10     enddo
11     dn = d
12     return
13     end
```

Listing 3: Función dn(y) - Derivada total de la iteración

## Función ffp(dum)

```

1  function ffp(dum)
2  implicit real*8(a-h,o-z)
3  common /lambda/al,eta,n
4  x_target = 1.0d0 / sqrt(3.0d0)
5  xl = x_target - eta
6  xu = x_target + eta
7  dd = fn(xl) - xl
8  if (dd * (fn(xu) - xu) .gt. 0.0d0) then
9      print *, n, 'no solution, usando x=', x_target
10     ffp = x_target
11     return
12 endif
13 do 1 i = 1, 40
14     x = 0.5d0 * (xl + xu)
15     y = fn(x)
16     if ((y - x) * dd .lt. 0.0d0) then
17         xu = x
18     else
19         xl = x
20     endif
21 1 continue
22 ffp = 0.5d0 * (xl + xu)
23 return
24 end

```

Listing 4: Función ffp - Punto fijo por bisección

## 5 Análisis de Resultados

Table 1: Constantes de Feigenbaum calculadas numéricamente

$n$	$\delta_n$	$\alpha_n$	$\lambda_n$	$x_n$
1	4.236067978	1.883707556	2.2360679775	0.5257311121
2	4.542933389	5.761620879	2.2880317545	0.5973271451
3	4.641203786	2.647962055	2.2992279397	0.5693230665
4	4.663183926	2.473009569	2.3016289140	0.5805454351
5	4.667909343	2.520528804	2.3021432716	0.5760721759
6	4.668925024	2.497081619	2.3022534377	0.5778606372
7	4.669142336	2.505492122	2.3022770323	0.5771463181
8	4.669188930	2.501930057	2.3022820855	0.5774317480
9	4.669198851	2.503310197	2.3022831677	0.5773177144
10	4.669201089	2.502749730	2.3022833995	0.5773632758
11	4.669202588	2.502970791	2.3022834492	0.5773450726
12	4.669199134	2.502883376	2.3022834598	0.5773523454
13	4.669152951	2.502950608	2.3022834621	0.5773494396

Los resultados del código muestran una convergencia clara de las razones  $\delta_n$  hacia el valor conocido de  $\delta \approx 4.6692$ , y de  $\alpha_n$  hacia  $\alpha \approx 2.5029$ . Las primeras iteraciones presentan errores más notorios debido a la cercanía de los valores y a la sensibilidad numérica, pero a partir de  $n = 5$  la aproximación es notablemente estable.

## 6 Conclusiones

El método numérico implementado permite obtener con buena precisión las constantes universales de Feigenbaum para la función  $f(x) = \lambda x(1 - x^2)$ . Este trabajo evidencia la universalidad de estas constantes y su aparición en diferentes familias de funciones no lineales. Además, resalta la importancia del análisis numérico cuidadoso en sistemas caóticos.

## 7 Referencias

- M. J. Feigenbaum, "Quantitative Universality for a Class of Nonlinear Transformations," Journal of Statistical Physics, vol. 19, no. 1, pp. 25–52, 1978.
- H. G. Schuster, Deterministic Chaos, Wiley-VCH, 1988.
- S. H. Strogatz, Nonlinear Dynamics and Chaos, Westview Press, 2001.
- J. C. Sprott, Chaos and Time-Series Analysis, Oxford University Press, 2003.