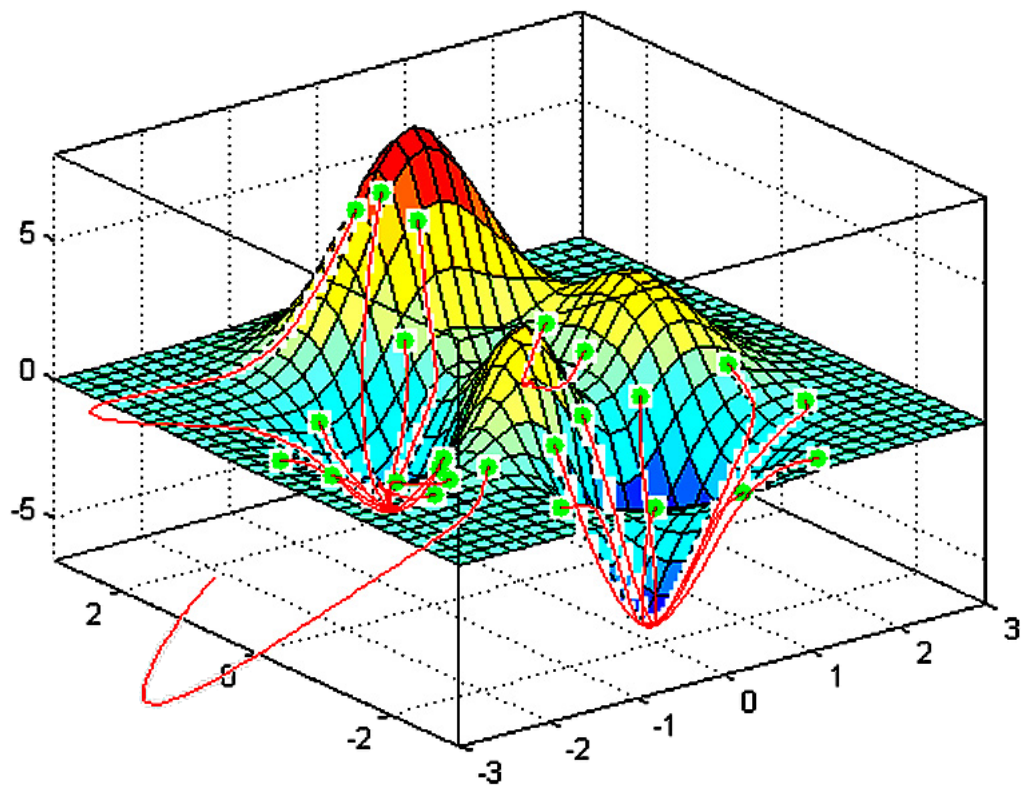


# Algoritmos de Optimización

Maximiliano Cartes, Jan Seiffert, Diego Zapata  
mcartes2017@alu.uct.cl, jseiffert2016@alu.uct.cl, dzapata2017@alu.uct.cl  
Calculo Avanzado  
MAT1189  
Semestre I-2021

Abril 2021

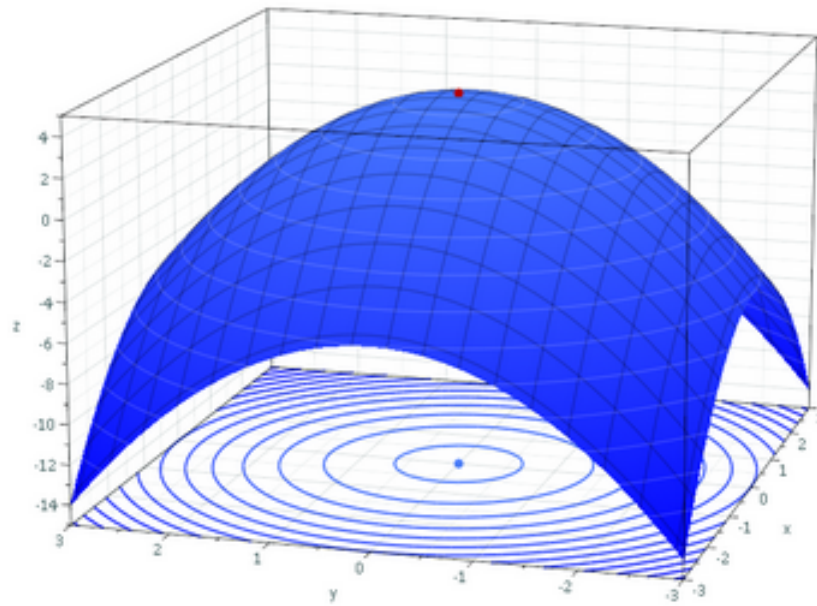


## Índice

1. Introducción	3
2. Optimización en ajuste de curvas	4
3. Métodos de optimización	5
4. Contexto de ámbito profesional en donde se deberá implementar distintos tipos de optimización	7
5. Otros ejemplos de CurveFitting, Interpolaciones y optimización en Python	8
6. Código para graficar funciones	11
7. Referencias	13

## 1. Introducción

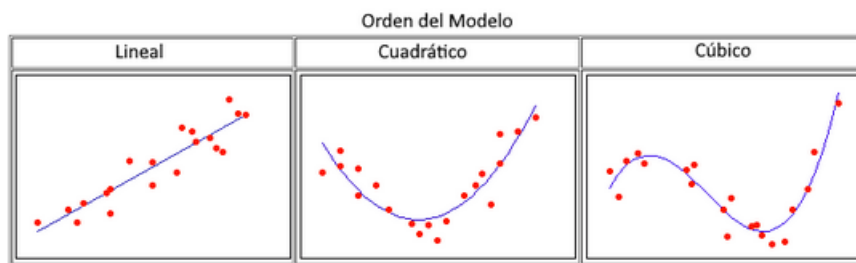
Un modelo es la abstracción de un problema real, en el cual se aplican ciertas consideraciones matemáticas permitiendo obtener resultados óptimos. Estos modelos son muy importantes para las tomas de decisiones dentro de empresas y organizaciones. Cuando ya se establecieron las mejores a partir del modelo realizado, los valores que son óptimos pueden diferir de la realidad debido a factores externos.



En otras palabras, Un modelo de optimización es la representación matemática a un problema real, en el cual tenemos conocimiento del impacto de cada una de las existentes variables, de las cuales intentamos encontrar el mínimo o máximo valor posible dentro de una función objetivo.

## 2. Optimización en ajuste de curvas

Para poder hablar de la optimización de ajuste de curvas, tenemos que comenzar con saber que es el ajuste de curvas, El ajuste de curvas implica encontrar una curva que sea contenida dentro de una serie de puntos los cuales satisfacen otras restricciones.



Por otro lado, la optimización de una función, implica encontrar sus valores máximos y mínimos. La optimización matemática es un concepto matemático esencial para abordar problemas de la vida cotidiana.

### 3. Métodos de optimización

En los métodos de optimización existen diversos modelos, todos estos para resolver diferentes problemas, ejemplo de ellos son:

- Reacción de orden cero: La velocidad de una reacción de orden cero es una constante, la cual no depende de la unión de los reactivos.

$$Q_1 = Q_0 + K_0 t$$

- Reacción de primer orden: Es una reacción donde la velocidad depende de la concentración de un reactivo elevado a la primera potencia.

$$\ln(Q)_t = \ln(Q)_0 + k_1 t$$

- Reacción de segundo orden: La velocidad con que ocurra una reacción depende que la concentración del reactivo elevado al cuadrado o de dos reactivos, de manera en que cada uno quede elevado a la uno.

$$Q_t / Q_\infty (Q_\infty - Q_t) K_2 t$$

- Modelo de Hixon-Crowel: También conocido como el modelo de la raíz cúbica.

$$Q_0^{1/3} - Q_t^{1/3} = K_s t$$

- Modelo de Weibull: Este modelo describe la conducta de los sistemas o eventos que tienen algún grado de alteración.

$$\log[-\ln(1 - (Q_t / Q_\infty))] = b \times \log(t) - \log(a)$$

- Modelo de Higuchi:.

$$Q_t = K_H \sqrt{t}$$

- Modelo de Baker-Lonsdale:

$$(3/2)[1 - (-1(Q_t / Q_\infty))^2/3] - (Q_t / Q_\infty) = K_t$$

- Modelo de Korsmeyer-Peppas:

$$Q_t / Q_\infty = K_h t^n$$

- Modelo Cuadrático: Este modelo describe una función cuadrática para representar una situación u objeto real.

$$Q_t = 100(K_1 t^2 + K_2 t)$$

- Modelo Logístico: Este modelo constituye un perfeccionamiento del modelo exponencial para el aumento de una magnitud. Modela la función sigmoidea de crecimiento de un conjunto Q.

$$Q_t = A / [1 + e^{-k(t-y)}]$$

- Modelo de Gompertz: Este modelo describe el crecimiento más lento al comienzo y al final de un período de tiempo establecido.

$$Q_t = Ae^{-e^{-K(t-y)}}$$

- Modelo de Hopfenberg: Este modelo describe la liberación por erosión de la superficie de la matriz.

$$Q_t / Q_\infty = 1 - [1 - K_0 t / C_0 a_0]^n$$

En este trabajo vamos a considerar dos modelos y su función costo: Con un parámetro (simplificación del modelo de Weibull).

$$y(x) = 1 - e^{-ax}$$

Con dos parámetros (modelo de Korsmeyer-Peppas):

$$y(x) = ax^b$$

Función Costo: Es aquella que mide la diferencia entre lo pronosticado por el modelo y los datos experimentales.

$$\sum_{i=1}^n (y(p; x_i) - \hat{y}_i)^2$$

La tabla con los datos a probar con los modelos que tenemos arriba es la siguiente:

$t_i$	0,1	0,2	0,3	0,4	0,5	0,6	0,7	0,8	0,9	1,0
$u_i$	0,2	0,3	0,45	0,55	0,6	0,7	0,75	0,8	0,8	0,8

#### 4. Contexto de ámbito profesional en donde se deberá implementar distintos tipos de optimización

Durante la vida profesional un ingeniero en informática puede encontrarse con variadas situaciones de optimización dependiendo del campo al que se dedique, aunque existe un campo donde estos problemas son más notorios, este es el campo llamado “Machine Learning”, en donde se pueden encontrar diversos algoritmos de trabajo para calcular distintos tipos de información o datos que se tienen. El que más se relaciona con esta materia que estamos viendo es la interpolación, puesto que el ajuste de curvas consisten en encontrar una curva que contenga una serie de puntos y que posiblemente cumpla una serie de restricciones (previamente dada) adicionales a este. Un ejemplo de aquello sería realizar el ajuste de un conjunto de datos pertenecientes a una ecuación cuadrática.

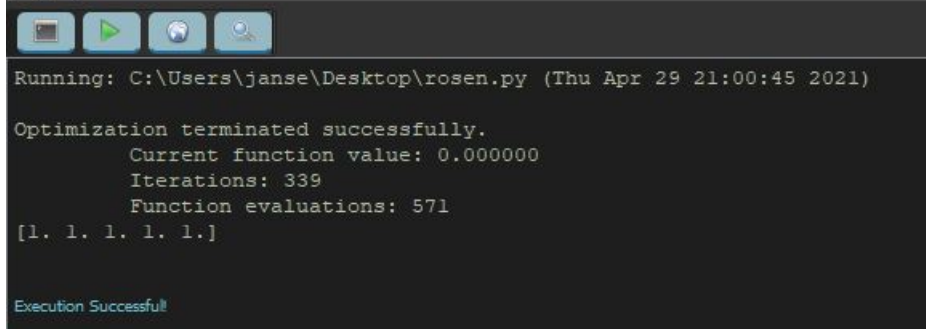
En el campo de Machine Learning de la informática estos problemas son muy comunes, por lo cual en este contexto profesional del M.L se trabajará de manera continua con este y otros algoritmos existentes. Pero si abarcamos mucho más en el ámbito de la Ingeniería en sí, a modo de visión general, el ajuste de curvas también se puede buscar la forma de implementarlo en la programación lineal, en el cálculo de máximos y mínimos que tendrán una serie de restricciones específicas dependiendo la situación que se está trabajando, Ejemplo de aquello, sería minimizar el costo de alguna fábrica y buscar sobre este mismo, la implementación de un ajuste de curvas al método gráfico que se está trabajando. Esto siempre y cuando se pueda hacer uso de este con los datos y restricciones que esta clase de problemas plantea.

## 5. Otros ejemplos de CurveFitting, Interpolaciones y optimización en Python

### Optimización

La función Minimize de la librería Scipy nos provee con algoritmos de optimización para funciones. En la siguiente imagen se puede apreciar un código que en pocas líneas, optimiza la función Rosenbrock.

```
1 import numpy as np
2 from scipy.optimize import minimize
3 def rosen(x):
4     return sum(100.0*(x[1:]-x[:-1]**2.0)**2.0 + (1-x[:-1])**2.0)
5 x0 = np.array([1.3, 0.7, 0.8, 1.9, 1.2])
6 res = minimize(rosen, x0, method='nelder-mead',
7               options={'xatol': 1e-8, 'disp': True})
8 print(res.x)
```



```
Running: C:\Users\janse\Desktop\rosen.py (Thu Apr 29 21:00:45 2021)

Optimization terminated successfully.
    Current function value: 0.000000
    Iterations: 339
    Function evaluations: 571
[1. 1. 1. 1. 1.]

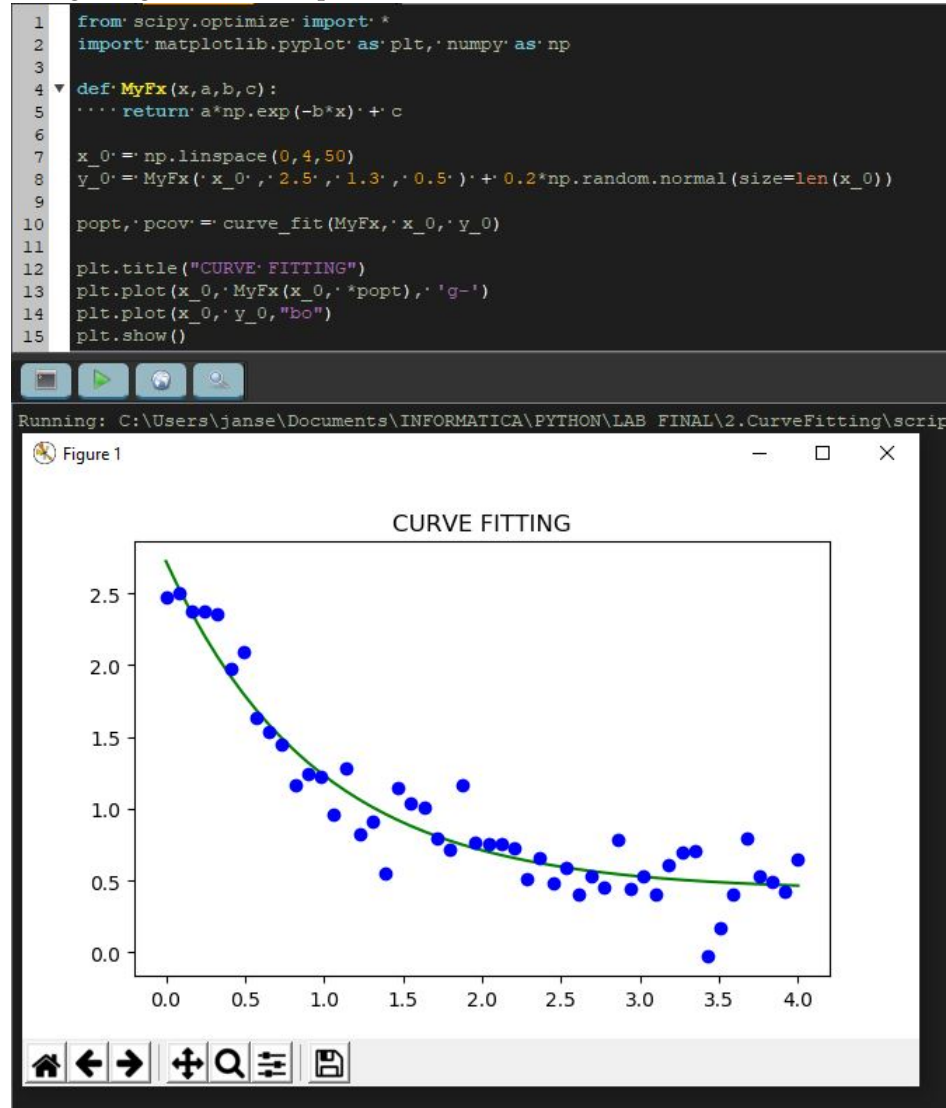
Execution Successful
```

El código nos arroja en la terminal los resultados de la optimización, las evaluaciones a la función e iteraciones.



## CurveFitting

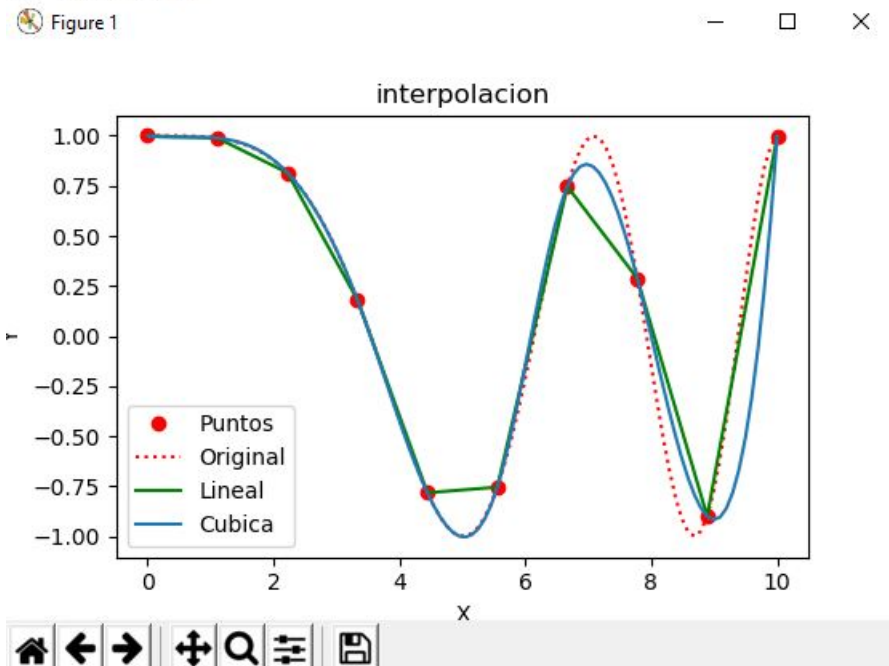
La función Curvefit de la librería Scipy.optimize.curvefit nos provee con algoritmos para lograr curve fitting. En el siguiente código implementamos el curve fitting con gráficos de matplotlib.



## Interpolación

La función `interpolate` de la librería `Scipy` nos provee con algoritmos para lograr la interpolación. En el siguiente código implementamos la interpolación con gráficos de `matplotlib`.

```
1 from scipy.interpolate import *
2 import matplotlib.pyplot as plt, 'numpy' as np
3 #XXXXXXXXXXXXXXXXX (PUNTOS) XXXXXXXXXXXXXXXXXXXX
4 x_0 = np.linspace(0,10,10)
5 y_0 = np.cos(x_0*2.0/8.0)
6 #XXXXXXXXXXXXXXXXX (ORIGINAL) XXXXXXXXXXXXXXXXXXXX
7 x_1 = np.linspace(0,10,100)
8 y_1 = np.cos(x_1*2.0/8.0)
9 print y_0
10 #XXXXXXXXXXXXXXXXX (LINEAL) XXXXXXXXXXXXXXXXXXXX
11 f1 = interpolate.interp1d(x_0, y_0, kind='linear')
12 y_2 = f1(x_1)
13
14 #XXXXXXXXXXXXXXXXX (CUBICA) XXXXXXXXXXXXXXXXXXXX
15 f2 = interpolate.interp1d(x_0, y_0, kind='cubic')
16 y_3 = f2(x_1)
17 plt.plot(x_0, y_0, "ro", x_1, y_1, "r:", x_1, y_2, "g-", x_1, y_3, "b-")
18 plt.title("interpolacion")
19 plt.xlabel("X")
20 plt.ylabel("Y")
21 plt.legend(["Puntos", "Original", "Lineal", "Cubica"], loc='best')
22 plt.show()
```



## 6. Código para graficar funciones

```
import math as ma,matplotlib.pyplot as plt,numpy as np

def funcionHiguchi(a,t):
    return a*(t**0.5)
def funcionWeibull(a,t):
    return 1-np.exp(-(a*t))
def modeloCuadratico(a,t):
    return 100*(a*t**2+a*t)

t = [0.5,1.0,1.5,2.0,2.5,3.0,3.5,4.0,4.5,5.0]
a = [-3.5,-2.5,-2,-1.5,-0.5,1,1.5,2,2.5,3.0]
length = len(a)

print("Ingrese un número:")
print("1.- Modelo de Higuchi \n2.- Modelo de Weibull \n3.- Modelo Cuadrático")
x = int(input("Número:"))
print("-----")
if (x==1):
    print("Higuchi!");
    for i in range(length):
        n = funcionHiguchi(a[i],t[i])
        plt.plot(t[i],n,marker="x", color="r")
    plt.plot(t,a)
elif (x==2):
    print("Weibull!");
    for i in range(length):
        n = funcionWeibull(a[i],t[i])
        plt.plot(t[i],n,marker="x", color="r")
    plt.plot(t,a)
elif (x==3):
    print("Cuadrático!");
    for i in range(length):
        n = modeloCuadratico(a[i],t[i])
        plt.plot(t[i],n,marker="x", color="r")
    plt.plot(t,a)
plt.grid(True)
plt.show()
```

Ahora ingresamos el número para el modelo que necesitamos graficar por pantalla.

Ingrese un número:

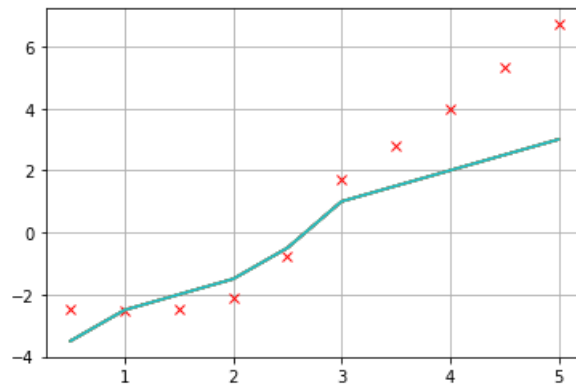
1.- Modelo de Higuchi

2.- Modelo de Weibull

3.- Modelo Cuadrático

Número:1

Higuchi!



## 7. Referencias

[Scipy, 2021] The SciPy community(2021, 26 abril). optimize.curve . Scipy.  
<https://docs.scipy.org/doc/scipy/reference/interpolate.html>

[Scipy, 2021] The SciPy community(2021, 26 abril). Interpolation \_fit. Scipy.  
[https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.curve\\_fit.html](https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.curve_fit.html)

[Numpy, 2021] The Numpy community(2021). Numpy.arange, Numpy.  
<https://numpy.org/doc/stable/reference/generated/numpy.arange.html>

[UACH, 2013] YÁÑEZ D.(2013). EVALUACIÓN DE CINÉTICAS DE LIBERACIÓN DE ALIMENTOS MEDICADOS A BASE DE BENZOATO DE EMAMECTINA, UTILIZANDO MEDIOS ALTERNATIVOS DE DISOLUCIÓN. cybertesis. <http://cybertesis.uach.cl/tesis/uach/2013/fcy.22e/doc/fcy.22e.pdf>

[Google Sites, 2021] ittgmetodosnumericos(2021). Unidad 4: Ajuste de curvas e interpolación. Google Sites.  
<https://sites.google.com/site/ittgmetodosnumericos/home/unidad-4-ajuste-de-curvas-e-interpolacion>