PRACTICA 2
Métodos Numéricos.

7/Diciembre/2018

Zepeda Montiel Antonio Aldair - NUA: 461770
Ramos Soto Michel Abraham - NUA: 768936

1 Introducción

La utilización de los diferentes modelos en métodos numéricos, se presentan con diversas aplicaciones en las ramas de la ingeniería, desde mediciones de señalesm sencillas, hasta aplicaciones que nos permiten la detección de células cancerigenas y cirugías de alta presición.

El desarrollo e implementación de estas metodologías se lleva a cabo desde modelos matemáticos basados en códigos de programación, que simplifican su desarrollo y amplian su aplicación con multiples propósitos.

2 Objetivos

Implementar los algoritmos para determinar los datos faltantes, analizar y comparar el desempeño de los diversos algoritmos y determinar cuando es mejor utilizar cada algoritmo en función de su desempeño y las características de los datos y el problema.

3 Procedimiento

Para la obtención y comprobación de nuestros modelos, utilizamos los datos de los archivos proporcionados por el profeso.

Mínimos cuadrados:

El método de mínimos cuadrados proporciona una forma de encontrar la mejor estimación, suponiendo que los errores sean aleatorios e imparciales, dados un conjunto de datos, se intenta determinar la función continua que mejor se aproxime a los datos, proporcionando una demostración visual de la relación entre los puntos de los mismos. Su expresión general se basa en la ecuación de una recta y = mx + b. Donde m es la pendiente y b el punto de corte, y vienen expresadas de la siguiente manera:

$$m = \frac{n \cdot \Sigma(x \cdot y) - \Sigma x \cdot \Sigma y}{n \cdot \Sigma x^2 - |\Sigma x|^2}$$

$$b = \frac{\Sigma y \cdot \Sigma x^2 - \Sigma x \cdot \Sigma(x \cdot y)}{n \cdot \Sigma x^2 - |\Sigma x|^2}$$

El método de mínimos cuadrados calcula a partir de los N pares de datos experimentales (x, y), los valores m y b que mejor ajustan los datos a una recta. Se entiende por el mejor ajuste aquella recta que hace mínimas las distancias d de los puntos medidos a la recta.

Teniendo una serie de datos (x, y), debemos aplicar el método de mínimos cuadrados, basándonos en su expresión general:

$$y = \left(\frac{n \cdot \Sigma(x \cdot y) - \Sigma x \cdot \Sigma y}{n \cdot \Sigma x^2 - |\Sigma x|^2}\right) x + \left(\frac{\Sigma y \cdot \Sigma x^2 - \Sigma x \cdot \Sigma(x \cdot y)}{n \cdot \Sigma x^2 - |\Sigma x|^2}\right)$$

Todos los resultados obtenidos y procedimientos implementados mediante la aplicación de nuestros modelos fué desarrollado bajo el lenguaje de programación MATLab, y consta principalmente de los siguientes elementos:

- **Regresión suponiendo un modelo lineal:**

  Para la aplicación del presente modelo la implementación del código queda desarrollada de la siguiente manera:

```
%%Para un modelo Lineal
figure(6)
plot(inf(1,:),inf(2,:),'r.');
hold on;
plot(x1,y1,'b.');
grid on;
x=inf(1,:);
x1=inf1(1,:);
y=inf(2,:);
y1=inf1(2,:);
nm = length(y1);
nt=1;
n=20;
    fin=fin+c;
    [B,f1]=MC(x,y,nt);
    yp=f1(x1);
    ea=(y1-yp).^2/c;
    xt=x(1):0.001:x(end);
yt=f1(xt);
plot(xt,yt,'b-');
xlabel('x');
ylabel('y');
title('Para un modelo lineal');
print('-f6', '-djpeg90', '-r300', 'Grafica1.jpg');
hold off;
```

- **Regresión en n segmentos suponiendo un modelo lineal:**

  Para la aplicación del presente modelo la implementación del código queda desarrollada de la siguiente manera:

```
%%Regresion lineal en n
nm = length(y1);
nt=1;
n=20;
c=floor(nm/n);
ini=1;
fin=0;
for i=1:n
    fin=fin+c;
    [B,f1]=MC(x(ini:fin),y(ini:fin),nt);
    yp=f1(x1(ini:fin));
    ea(ini:fin)=(y1(ini:fin)-yp).^2/c;
    xt=x(ini):0.001:x(fin);
    ini=fin;
yt=f1(xt);
plot(xt,yt,'b-');
end
xlabel('x');
ylabel('y');
title('n segmentos lineal');
print('-f1', '-djpeg90', '-r300', 'Grafica1n.jpg');
hold off;

one= ones(nm,1);
et= ea*one;
```

- **Regresión suponiendo un modelo cuadrático:**

  Para la aplicación del presente modelo la implementación del código queda desarrollada de la siguiente manera:

```
%%Para un modelo Cuadratico
figure(4)
plot(inf(1,:),inf(2,:),'r.');
hold on;
plot(x1,y1,'b.');
grid on;
x=inf(1,:);
x1=inf1(1,:);
y=inf(2,:);
y1=inf1(2,:);
nm = length(y1);
nt=2;
    fin=fin+c;
    [B,f1]=MC(x,y,nt);
    yp=f1(x1);
    ea=(y1-yp).^2/c;
    xt=x(1):0.001:x(end);
yt=f1(xt);
plot(xt,yt,'b-');
xlabel('x');
ylabel('y');
title('Para modelo cuadrado');
print('-f4', '-djpeg90', '-r300', 'Grafica2.jpg');
hold off;
```

- **Regresión en n segmentos suponiendo un modelo cuadrático:**

Para la aplicación del presente modelo la implementación del código queda desarrollada de la siguiente manera:

```
%%Regresion n para un modelo Cuadratico
figure(3)
plot(inf(1,:),inf(2,:),'r.');
hold on;
plot(x1,y1,'b.');
grid on;
x=inf(1,:);
x1=inf1(1,:);
y=inf(2,:);
y1 = inf1(2,:);
nm = length(y1);
nt=2;
n=20;
c=floor(nm/n);
ini=1;
fin=0;
for i=1:n
    fin=fin+c;
    [B,f1]=MC(x(ini:fin),y(ini:fin),nt);
    yp=f1(x1(ini:fin));
    ea2(ini:fin)=(y1(ini:fin)-yp).^2/c;
    xt=x(ini):0.001:x(fin);
    ini=fin;
yt=f1(xt);
plot(xt,yt,'b-');
end
one= ones(nm,1);
et2= ea2*one;
xlabel('x');
ylabel('y');
title('Para n segmentos cuadrado');
print('-f3', '-djpeg90', '-r300', 'Grafica2n.jpg');
hold off;
```

- **Regresión suponiendo un modelo cúbico:**

  Para la aplicación del presente modelo la implementación del código queda desarrollada de la siguiente manera:

```
%%Para un modelo cubico
figure(7)
plot(inf(1,:),inf(2,:),'r.');
hold on;
plot(x1,y1,'b.');
grid on;
x=inf(1,:);
x1=inf1(1,:);
y=inf(2,:);
y1=inf1(2,:);
nm = length(y1);
nt=3;
    fin=fin+c;
    [B,f1]=MC(x,y,nt);
    yp=f1(x1);
    ea=(y1-yp).^2/c;
    xt=x(1):0.001:x(end);
yt=f1(xt);
plot(xt,yt,'b-');
hold off;
xlabel('x');
ylabel('y');
title('Para modelo cubico');
print('-f7', '-djpeg90', '-r300', 'Grafica3.jpg');
hold off;
```

- **Regresión en n segmentos suponiendo un modelo cúbico:**

  Para la aplicación del presente modelo la implementación del código queda desarrollada de la siguiente manera:

```
%%Para n segmentos cubicos
figure(5)
plot(inf(1,:),inf(2,:),'r.');
hold on;
plot(x1,y1,'b.');
grid on;
x=inf(1,:);
x1=inf1(1,:);
y=inf(2,:);
y1=inf1(2,:);
nm = length(y1);
nt=3;
n=20;
c=floor(nm/n);
ini=1;
fin=0;
for i=1:n
    fin=fin+c;
    [B,f1]=MC(x(ini:fin),y(ini:fin),nt);
    yp=f1(x1(ini:fin));
    ea3(ini:fin)=(y1(ini:fin)-yp).^2/c;
    xt=x(ini):0.001:x(fin);
    ini=fin;
yt=f1(xt);
plot(xt,yt,'b-');
end
one= ones(nm,1);
et3= ea3*one;
xlabel('x');
ylabel('y');
title(' Para n segmentos cubico');
print('-f5', '-djpeg90', '-r300', 'Grafica3n.jpg');
hold off;
```

Interpolación:

Se denomina interpolación a la obtención de nuevos puntos partiendo del conocimiento de un conjunto discreto de puntos.

Si tenemos una función cuyo cálculo resulta costoso, podemos partir de un cierto número de sus valores e interpolar dichos datos construyendo una función más simple. En general, por supuesto, no obtendremos los mismos valores evaluando la función obtenida que si evaluamos la función original, si bien dependiendo de las características del problema y del método de interpolación usado la ganancia en eficiencia puede compensar el error cometido.

En todo caso, se trata de, a partir de n parejas de puntos (xk,yk), obtener una función **f** que verifique:

$$f(x_k) = y_k \ , k = 1, \ldots, n$$

- **Interpolación Lineal:**

En la interpolación lineal se utilizan dos puntos, (xa,ya) y (xb,yb), para obtener un tercer punto interpolado (x,y) a partir de la siguiente fórmula:

$$y = y_a + (x - x_a)\frac{(y_b - y_a)}{(x_b - x_a)}$$

La interpolación lineal es rápida y sencilla, pero en ciertos casos no muy precisa.

- **Interpolación cuadrática:**

En la interpolación cuadrática, para mejorar la aproximación introducimos cierta curvatura en la linea que conecta a los puntos. Si se dispone de tres puntos se puede llevar a cabo un polinomio de segundo orden:

$$f(x) = a_2x^2 + a_1x + a_0$$

donde:

$$\mathrm{a}_2 = b_2$$
$$\mathrm{a}_1 = b_1 - b_2x_0 - b_2x_1$$
$$\mathrm{a}_0 = b_0 - b_1x_0 + b_2x_0x_1$$

Teniendo los valores de b0, b1 y b2 definidos de la siguiente manera:

$$\mathrm{b}_0 = f(x_0)$$
$$\mathrm{b}_1 = \frac{f(x_1) - f(x_0)}{x_1 - x_0}$$
$$\mathrm{b}_2 = \frac{\frac{f(x_2) - f(x_1)}{x_2 - x_1} - \frac{f(x_1) - f(x_0)}{x_1 - x_0}}{x_2 - x_0}$$

(1)

Interpolación polinomial:

Dado un conjunto de k+1 puntos (x0 , y0), . . . ,(xk , yk) donde todos los xj se asumen diferentes, el polinomio de interpolación es la combinación lineal:

$$L(x) = \sum_{j=0}^{k} y_j \ell_j(x)$$

- **Interpolación lineal, cuadrática, cúbica y cuarta:** Finalmente, el código correspondiente a las diferentes interpolaciones en una dimensión queda expresado de la siguiente manera:

```
clc;
close all;
clear all;

xl=load('Datos_2_2.txt');
yl=load('Datos_2_1.txt');
inc=0.1;
np = length(yl);
y=yl(1,:);
fxl=yl(2,:);
for il=1:np
    xp = xl(1,il);
    x=interpn(yl,xp,1);
   interli(1,il) = x;
end
ea=(xl(2,:)-interli).^2/np;
one=ones(np,1);
et=ea*one;
figure(1);
plot(xl(1,:),xl(2,:),'r.',yl(1,:),yl(2,:),'b.',xl(1,1:end),interli,'g.');
grid on;
xlabel('x');
ylabel('y');
title('Interpolacion lineal');
print('-f1', '-djpeg90', '-r300', 'GraficaI1.jpg');

fx = zeros(1,np);
for il=1:np
    xp = xl(1,il);
    x=interpn(yl,xp,2);
   interli2(1,il) = x;
end
ea=(xl(2,1:end)-interli2).^2/np;
one=ones(np,1);
et2=ea*one;
figure(2);
plot(xl(1,:),xl(2,:),'r.',yl(1,:),yl(2,:),'b.',xl(1,1:end),interli2,'g.');
grid on;
```

```
grid on;
xlabel('x');
ylabel('y');
title('Interpolacion Cuadrada');
print('-f2', '-djpeg90', '-r300', 'GraficaI2.jpg');

for il=1:np
    xp = xl(1,il);
    x=interpn(yl,xp,3);
   interli3(1,il) = x;
end
ea=(xl(2,1:end)-interli3).^2/np;
one=ones(np,1);
et3=ea*one;
figure(3);
plot(xl(1,:),xl(2,:),'r.',yl(1,:),yl(2,:),'b.',xl(1,1:end),interli3,'g.');
grid on;
xlabel('x');
ylabel('y');
title('Interpolacion Cubica');
print('-f3', '-djpeg90', '-r300', 'GraficaI3.jpg');

for il=1:np
    xp = xl(1,il);
    x=interpn(yl,xp,4);
   interli4(1,il) = x;
end
ea=(xl(2,1:end)-interli4).^2/np;
one=ones(np,1);
et4=ea*one;
figure(4);
plot(xl(1,:),xl(2,:),'r.',yl(1,:),yl(2,:),'b.',xl(1,1:end),interli3,'g.');
grid on;
xlabel('x');
ylabel('y');
title('Interpolacion Cuarta');
print('-f4', '-djpeg90', '-r300', 'GraficaI4.jpg');
```

Interpolación en dos dimensiones:
La interpolación bilineal es una extensión de la interpolación lineal para interpolar funciones de dos variables (por ejemplo, x e y) en una malla regular de dos dimensiones.

La idea principal es realizar una interpolación lineal en una dirección, y después en la otra. Aunque cada uno de estos pasos es lineal, la interpolación en su conjunto no es lineal sino cuadrática.

Supóngase que se quiere encontrar el valor para la función f desconocida en el punto P = (x, y). Conocemos el valor de f en los cuatro puntos Q11 = (x1, y1), Q12 = (x1, y2), Q21 = (x2, y1) y Q22 = (x2, y2).

Primero se hace una interpolación lineal en la dirección x. Esto genera:

$$f(R_1) \approx \frac{x_2 - x}{x_2 - x_1} f(Q_{11}) + \frac{x - x_1}{x_2 - x_1} f(Q_{21})$$

donde R1 = (x,y1)

$$f(R_2) \approx \frac{x_2 - x}{x_2 - x_1} f(Q_{12}) + \frac{x - x_1}{x_2 - x_1} f(Q_{22})$$

donde R2 = (x,y2)
Después se hace una interpolación en la dirección y:

$$f(P) \approx \frac{y_2 - y}{y_2 - y_1} f(R_1) + \frac{y - y_1}{y_2 - y_1} f(R_2).$$

Esto proporciona una estimación de f(x, y).

$$\begin{aligned}
f(x,y) \approx & \frac{f(Q_{11})}{(x_2 - x_1)(y_2 - y_1)}(x_2 - x)(y_2 - y) + \\
& \frac{f(Q_{21})}{(x_2 - x_1)(y_2 - y_1)}(x - x_1)(y_2 - y) + \\
& \frac{f(Q_{12})}{(x_2 - x_1)(y_2 - y_1)}(x_2 - x)(y - y_1) + \\
& \frac{f(Q_{22})}{(x_2 - x_1)(y_2 - y_1)}(x - x_1)(y - y_1) \\
= & \frac{1}{(x_2 - x_1)(y_2 - y_1)} \Big(f(Q_{11})(x_2 - x)(y_2 - y) + \\
& f(Q_{21})(x - x_1)(y_2 - y) + \\
& f(Q_{12})(x_2 - x)(y - y_1) + \\
& f(Q_{22})(x - x_1)(y - y_1) \quad \Big).
\end{aligned}$$

- **Interpolación bi-lineal:**

```
xlabel('x');
ylabel('y');
title('3D');
print('-f1', '-djpeg90', '-r300', 'GraficaID1.jpg');

figure(2)
mesh(y1(1,:), y1(2,:), y1(3:end,:));
hold on;
for i=1:np
    x(i,:)=x1(1,:);
    x2(i,:)=x1(2,:);
end
xi=x1(1,:);
xi2=x1(2,:);
yi=y1(1,:);
yi2= y1(2,:);
    plot3(x, x2', x1(3:end,:),'b.');
    mesh(y1(1,:), y1(2,:), y1(3:end,:));
grid on;
xlabel('x');
ylabel('y');
title('3D');
print('-f2', '-djpeg90', '-r300', 'GraficaID2.jpg');

%%
xi=x1(1,:);
xi2=x1(2,:);
yi=y1(1,:);
yi2= y1(2,:);
fxy=y1(3:end,:);
fxy2=x1(3:end,:);
incx=yi(2)-yi(1);
incy=yi2(2)-yi2(1);
nm=1;
```

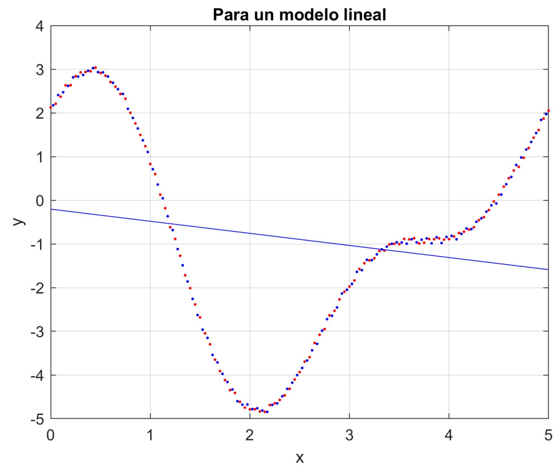
- **Interpolación bi-cuadrática:**

```
for i=1:np
    for j=1:np
        yp=xi2(j);
        xp=xi(i);
       [it,xt]=interp2(yi,xp,incx);
       [it2,yt]=interp2(yi2,yp,incy);
        n=length(xt);
        for k=1:n
            p(2,k)=interp2([yi;fxy(:,it2(k))'],xp,nm);
        end
        p(1,:)=yi(it2);
        fxyp3(i,j)=interp2(p,yp,nm);
    end
end
for i=1:n
    ea(i,:)=(fxyp3(i,:)-fxy2(i,:)).^2;
end
et3=0;
for i=1:n
    et3=et3+sum(ea(i,:))/(n*n);
end
    figure(5)
    plot3(x, x2', fxyp3,'g.');
    hold on;
    plot3(x, x2', x1(3:end,:),'b.');
    mesh(y1(1,:), y1(2,:), y1(3:end,:));
    xlabel('x');
    ylabel('y');
    title('3D');
    grid on;
    print('-f5', '-djpeg90', '-r300', 'GraficaID5.jpg');
```

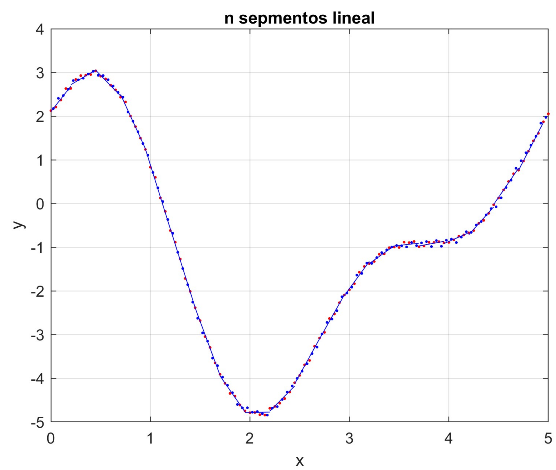
4 Resultados

Para los resultados de los métodos aplicados previamente, obtenemos los siguientes resultados mostrados en matlab:

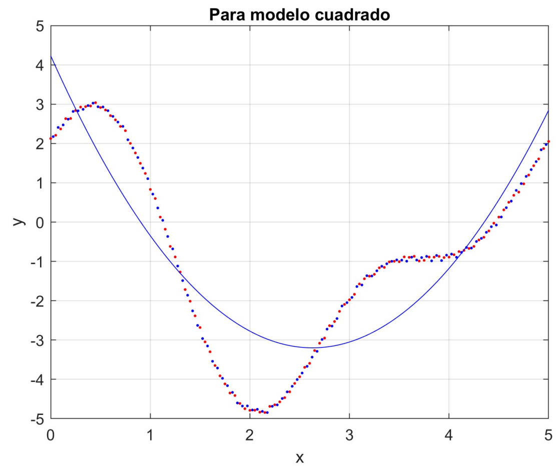
- **Regresión por mínimos cuadrados:**
  - **Regresión suponiendo un modelo lineal:**



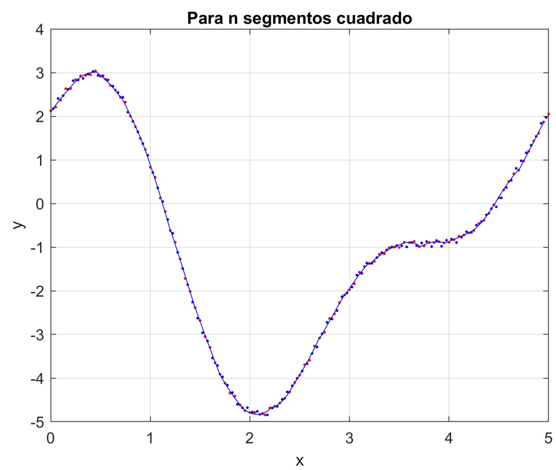
– **Regresión en n segmentos suponiendo un modelo lineal:**



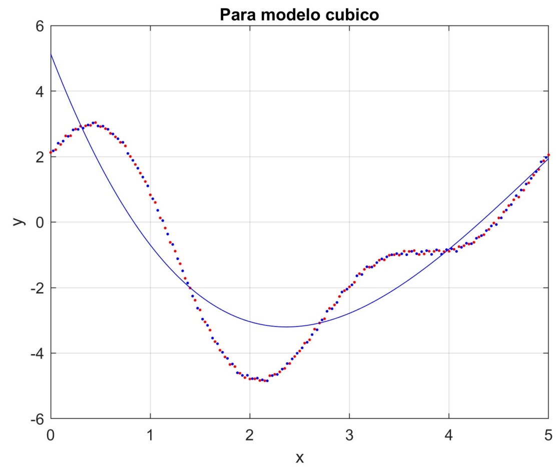
– **Regresión suponiendo un modelo cuadrático:**



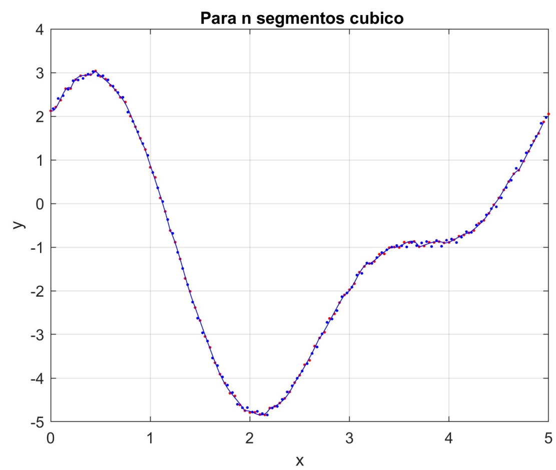
– **Regresión en n segmentos suponiendo un modelo cuadrático:**



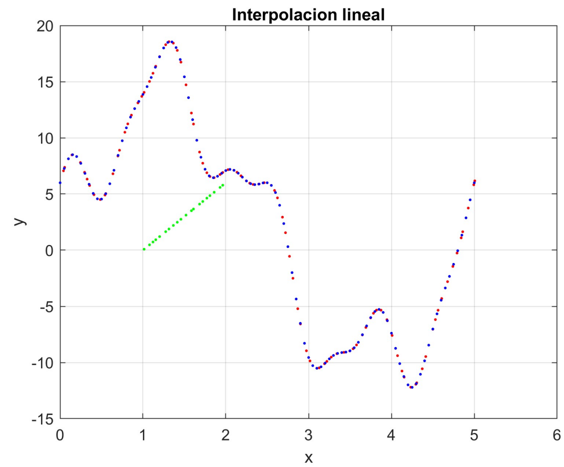
– **Regresión suponiendo un modelo cúbico:**



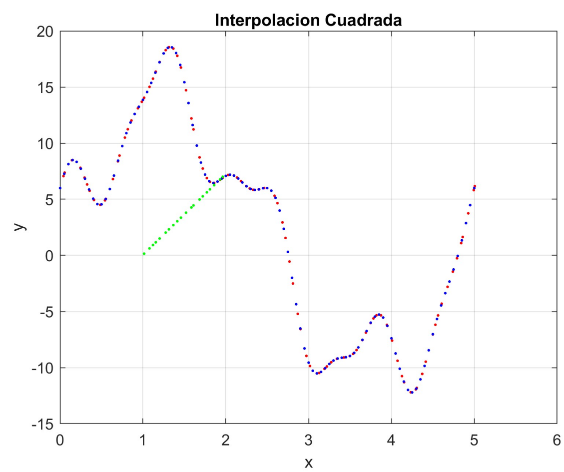
– **Regresión en n segmentos suponiendo un modelo cúbico:**



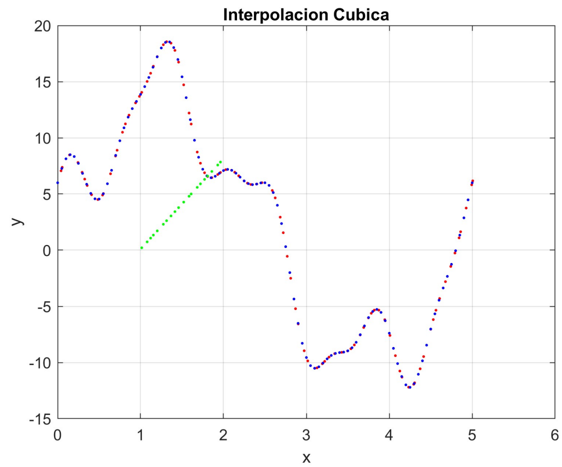
- **Interpolación en una dimensión:**
  – **Interpolación lineal:**



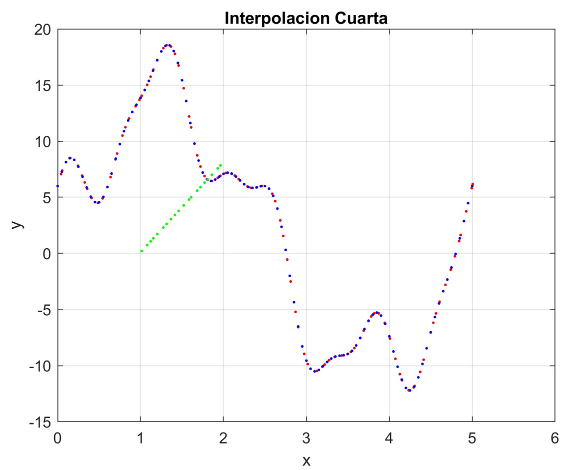
– **Interpolación cuadrática:**



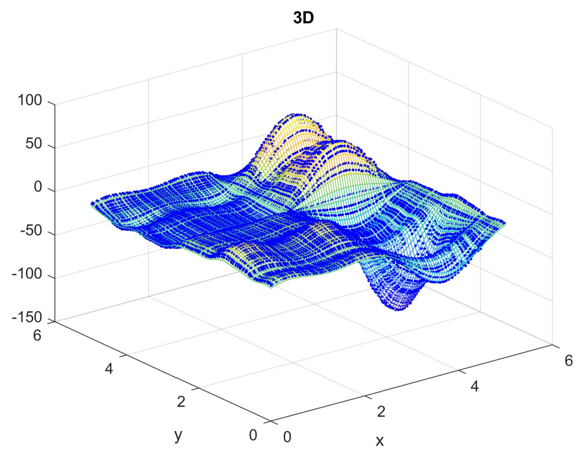
– **Interpolación cúbica:**

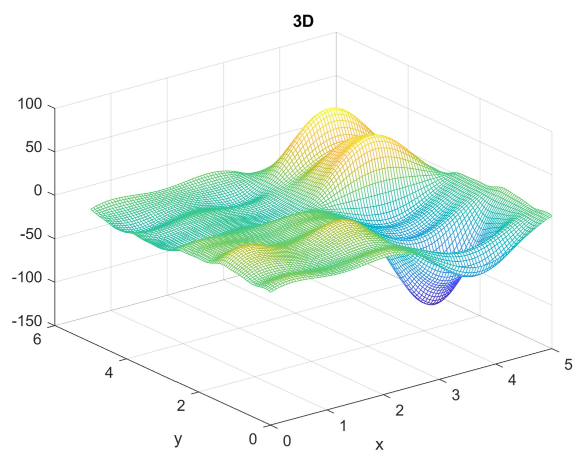


– **Interpolación cuarta:**



- **Interpolación bidimensional:**





5 Tabla comparativa

Finalmente, una vez obtenidos los resultados de los valores evaluados la simulación, podemos apreciar variación entre los diferentes valores obtenidos a mayor profundidad mediante los valores de las tablas comparativas que se presentarán a continuación:

Datos_1		Mod Lin	Cuadratico	Cubico
x1	y1	yt	yt	yt
0.025	2.1739086	-0.2111523	4,0884.00	3,29933337413494.00
0.525	2.9276934	-0.34962717	1,5586939037317508.00	1,581135233328698.00
1.025	0.71107859	-0.48810203	-4.336937186	0,796716543826416.00
1.525	-2.9622422	-0.62657689	1,857688890354630.00	2,306956385735350.0
2,025	-4.7791873	-0.76505175	2,775949639282830.00	3,061532061167476.0
2,525	-3.8393187	-0.90352662	3,186660481368919.00	-3.172391339
3,025	-1.916253	-1.04200148	3,186660481368911.00	2,7514819876787434.0
3,525	-0.96822777	-1.18047634	2,335141569660006.00	-1.910751776
4,025	-0.81456522	-1.3189512	1,103333087740744.00	-7.62148E+14

Como podemos apreciar, las variaciones entre los diferentes modelos varías sus acercamientos a los resultados de manera significativa, en caso de los diferentes modelos, la lejanía entre el cúbico y lineal son significativas, en cambio, comparandolo con el modelo cuadrático, existe mayor similitud entre ellos, sin embargo, el modelo lineal mantiene una cercanía mayor al valor y1.

En caso contrario, la cercanía del modelo cuadrático y cúbico es mayor para muchos de los puntos por el modelo cúbico y cuadrático que por el modelo lineal.
Debido a la aleatoriedad de los errores, podemos diferenciar que existen valores outliers en cualquiera de los metodos, pero se aproximan o se alejan dependiendo del tipo de modelo utilizado.

Datos_2		Mod Lin (s)	Cuadráticos (s)	Cubico (s)
x1	y1	yt	yt	yt
0.05	2,2085578.0	0.75851154	0.796495582045298	0.800777040844340
0.525	2.9156934	0.80237879	0.899499980001547	0.903803124607482
2,050	-4.784844	0.8901133	0.942572086948809	1,00302483732.0
2.55	-3.694884	0.933980556	1,44547837069521.00	0,238714564936466.00
4,050	-0.81456522	0.986621259	2,944506118920890.0	0,957986747056602.0

Para las aplicaciones con los modelos en segmentos n, existe una variación mayor cuando se trata del modelo cuadrático, esto debido a que conforme se van alejando los valores hacia los extremos, al tratarse de una expresión cuadrática, el incremento en sus datos tiende a aumentar de manera significatia, a diferencia del modelo cubico, que mantiene una cercanía mayor. Sin embargo, el modelo lineal mantiene valores que se ubican ligeramente mas lejos que los valores obtenidos del modelo cúbico, esto puede apreciarse facilmente con los resultados de cada valor y final con respecto a los de y1.

6 Conclusiones

El poder realizar esta práctica nos permitió profundizar más en los métodos de regresión para la obtención de un modelo que permitiera aproximar los valores de una función determina, esto nos da la posibilidad de aplicar con multiples funcionalidades dichas metodologías en diversas ramas de la ingeniería.

A pesar de que dichos modelos y algoritmos representen modelos relativamente complitados, y a pesar de haber obtenido diferentes complicaciones durante el desarrollo del código, pudimos apreciar algunas caracteristicas de los modelos presentados e implentarlos de manera satisfactoria en los objetivos básicos para nuestra carrera.

7 Bibliografía

References

[1] Interpolación - http://aniei.org.mx/paginas/uam/CursoMN/curso$_m n_0 5.html$.

[2] Interpolación bilineal- http://www.wikiwand.com/es/Interpolaciónbilineal

[3] Regresión Lineal - http://www.sc.ehu.es/sbweb/fisica/cursoJava/numerico/regresion/reg