Taller N°5 Interpolación polinomial

Emil Vega - Valentina Cordova

20 de junio de 2016

1. RESUMEN

1.1. LAGRANGE

Se consideran tres ejercicios en cuales se aplica los conocimientos adquiridos en clase acerca de la obtención de un polinomio de interpolación. Se hace uso de la base de Lagrange para el cual se necesita un conjunto de $n+1$ puntos en el plano, es decir distintas coordenadas en x:

$$(x_0, y_0), (x_1, y_1), (x_2, y_2), ..., (x_n, y_n)$$

El interés es encontrar una función polinómica que pase por esos $n+1$ puntos y que tenga el menor grado, este polinomio será el denominado polinomio de interpolación.

Así pues se hace uso de la fórmula general para el polinomio de interpolación de Lagrange:

$$P(x) = \sum_{i=0}^{n} l_i(x) \cdot y_i \tag{1}$$

Para ello son usados los polinomios básicos de Lagrange dados por:

$$l_i(x) = \prod_{\substack{m=0 \\ m \neq 0}}^{n} \frac{(x - x_m)}{(x_i - x_m)} \tag{2}$$

Al modificar el m-archivo *lagrange* como $lagrange_2$ para que ahora el parametro z sea un vector al igual que *valor*, donde nz es el tamaño de z y para que sea recorrido se utiliza un $for - end$ en MATLAB.

1.2. ESTIMACIÓN DEL ERROR DE INTERPOLACIÓN

Se utiliza la fórmula para el calculo del error abosulto:

$$|f(x) - p_n(x)| < 10^{-8} \tag{3}$$

donde,
$p_n(x)$: es el polino de interpolación.
$f(x)$: la función dada.

Como resultado se obtuvo que elmenor grado posible para el polinomio de interpolación fue 15 para la función $f(x) = senh(x) para x : [-1, 1]$.

1.3. FENÓMENO DE RUNGE

En el campo de análisis numérico el fenómeno de Runge es un problema que surge al utilizar interpolación polinómica con polinomios de alto grado utilizando nodos equidistantes.

Considerando la función $g(x) = \frac{1}{1+25\cdot x^2}$ se obtuvo para el apartado 'a' la grafica de la función y los polinomios que la interpolan. Que se podrá observar más adelante.

Además se graficó el error en la norma $\|\cdot\|_{\propto}$ en la aproximaciónn de g para $n = 10, 20, 40, 80$. Finalmente para el apartado 'c' se hizo la gráfica del error absoluto de la aproximación y el spline cúbico con la función de MATLAB.

1.4. FENÓMENO DE GIBBS

Este fenómeno describe el comportamiento que tiene la serie de Fourier asociada a una función definida a trozos periódica en una discontinuidad evitable de salto finito. Cuando la serie de Fourier presenta discontinuidades, no es posible una buena convergencia en sus entornos.

1.5. SPLINE CÚBICO

Se define como una curva diferenciable definida en partes mediante polinomios, este brinda resultados similares para la interpolación y solo requiere el uso de polinomios de bajo grado para evitar las oscilaciones. Para hallar el spline cúbico se utilizó la función "spline"de MATLAB para la evaluación numérica de los respectivos polinomios de interpolación.

2. PLANTEAMIENTO DEL PROBLEMA

2.1. EJERCICIO 1: Algoritmo en MATLAB

Cree un m-archivo con el nombre *lagrange* que evalúe numéricamente el polinomio de interpolación utilizando la base de Lagrange.

En el ejercicio se crea un m-archivo haciendo uso de matlab llamado $lagrange_2$ este evalúa el polinomio de interpolación utilizando la base de Lagrange.

2.2. EJERCICIO 2: Estimación del error de interpolación

Se tiene la función $f(x) = senh(x) para x : [-1, 1]$

a.Encuentre el menor grado posible para el polinomio de interpolación p_n de los puntos $\{(x\_i, f(x\_i))\}_{i=0}^{n}$, tal que se satisfaga la ecuación 3.

Para lo cual se utilizó:
$f'(x) = cosh(x)$ primera derivada de $f(x)$ Para n impar
$f''(x) = senh(x)$ segunda derivada de $f(x)$ RPara n impar.

Se define:
$M = |f^{n+1}(1)$ para poder obtener n que sea el menor grado del polinomio de interpolación, es decir utilizamos $f''(x)$" .
y a continuación **resolvemos en MATLAB:** $\frac{(n+1)!}{2^{n+1}} > M \cdot 10^8$

b. Utilice puntos equidistantes para graficar la función y el polinomio de interpolación en $[-1, 1]$. Utilice una escala logarítmica en base 10 para representar errores. Adicionalmente grafique el error absoluto utilizando los nodos:

$$x_i = -1 + \frac{2i}{n} para i = 0, 1, ..., n \tag{4}$$

Para graficar la función y el polinomio de interpolación en $[-1, 1]$ se utilizó lo siguiente:

$*f(x) = senh(xi)$'
$*x_i$
$*f(x) = senh(xnodos)$

* Se utilizó m-archivo creado en el ejercicio 1 llamado: $langrange_2$

2.3. EJERCICIO 3. Fenómeno de Runge

Considere la función $g(x) = \frac{1}{1+25\cdot x^2} x : [-1,1]$, x_i definido en la ecuación 4 y otro adicional definido como

$$w_i = cos(\frac{(2(n-i)+1)\cdot\pi)}{2(n+1)}) parai = 0,1,..,n \tag{5}$$

a. Grafique la función g, y los polinomios que interpolan los puntos $\{(x\_i, g(x\_i))\}_{i=0}^{10}$ y $\{(w\_i, w(x\_i))\}_{i=0}^{10}$ y el spline cúbico de los puntos $\{(x\_i, g(x\_i))\}_{i=0}^{10}$.

*Para la gráfica de g se define $x = [-1, (2/2000), 1]$ y $y_i = g(x_i)$

* Para el polinomio 1, $xnodos$ haciendo uso de la ecuación 3 y $ynodos$ utilizando la evaluación de la función $g(xnodos)$.

*Para el polinomio 2, $wnodos$ haciendo uso de la ecuación 4 y $ywnodos$ utilizando la evaluación de la función $g(wnodos)$.

b. Grafique el error en la norma $\|\cdot\|_\infty$ en la aproximaciónn de g por el polinomio de interpolación de los puntos $\{(x\_i, g(x\_i))\}_{i=0}^{n}$ para $n = 10, 20, 40, 80$.

Se define x_n y y_n respectivamente a $n = 10, 20, 40, 80$ y posteriormente se utiliza $lagrange_2$ para obtener cada uno de los polinomios de interpolación. Los cuales serán los valores del eje y y sus cordenadas $(x, p_n(xi))$. Finalmente el error con la norma infinito se halla con la función de MATLAB.

c. Grafique el error absoluto en la aproximación de g por el polinomio de interpolación de los puntos $\{(w\_i, w(x\_i))\}_{i=0}^{n}$ y el spline cúbico de los puntos $\{(x\_i, g(x\_i))\}_{i=0}^{n}$ para $n = n = 10, 20, 40, 80$.

Se utiliza $lagrange_2$ para hallar el p_n en cada $n = 10, 20, 40, 80$ y se determina el error absoluto utilizando w_i definido en la ecuación 5. Por ultimó se hace uso del comando "spline"de MATLAB para hallar el spline cubico con x_i definido en la ecuación 4.

2.4. EJERCICIO 4. Fenómeno de Gibbs

Se considera la función $h(x) = |senh(x)| parax : [-1,1]$ y los nodos x_i y w_i definidos en la ecuaciones 4 y 5 respectivamente.

a. Grafique la función h, el polinomio de interpolación de los puntos $\{(w\_i, h(w\_i))\}_{i=0}^{n}$ y el spline cúbico para los puntos $\{(x\_i, h(x\_i))\}_{i=0}^{n}$ para $n = 10, 20, 40, 80$.

Para la cual se halla el polinomio de interpolación con $lagrange_2$, se realiza la gráfica de la función, del polinomio y del spline cúbico para $n = 10, 20, 40, 80$ utilizando el comando "spline"de MATLAB.

b. Grafique el error absoluto en la aproximanción de h con los polinomios definidos en 'a' para $n = 10, 20, 40, 80$.

Se ululiza los polinomios encontrados en el apartado .ªz se determina con el comando de MATLAB ïnf"la norma del error absoluto de la aproximación.

3. METODOLOGÍA

3.1. Descripción de los métodos numéricos utilizados y sus respectivas implementaciones computacionales

EJERCICIO 1

langrage: Es el método numérico utilizado para la obtención de los polinomios de inteporlación requeridos en el Taller N°5 haciendo uso de las ecuaciones 1 y 2 definidas anteriormente. Sin embargo el m-archivo original fue modificado y renombrado $legandre_2$ ya que el parametro z se convertirá en un vector de tamaño nz y la repuesta, es decir "$vector$" un tamaño $(1, nz)$ que será los polinomios básicos de Lagrange.

EJERCICIO 2

a.Aproximación del error de la interpolación de $f(x)$:
*Mediante la aproximación de la función $f(x) = senh(x) parax : [-1, 1]$ y sus derivadas $f1(x) = cosh(x)$ (primera derivada), para n impar y $f2(x) = senh(x)$ (segunda derivada) para n par.

Primero hallamos $M = |f2(1)|$ para poder obtener n mediante la fórmula $\frac{(n+1)!}{2^{n+1}} > M \cdot 10^8$ haciendo uso de las funciones de MATLAB. Como lo es "mod" para calcular el residuo y poder realizar la evaluación de n par o impar según la derivada que corresponda.

Al finalizar, el menor grado es = 15.

b. Coordenadas para realizar los gráficos

Para graficar la función y el polinomio de interpolación en $[-1, 1]$ se utilizó lo siguiente:
* $f(x) = senh(xi)$ para hallar yi.
*x_i para determinar los $xnodos(i)$
* $f(x) = senh(xnodos)$ para hallar $ynodos(i)$.

Lagrange * Se utilizó m-archivo creado en el ejercicio 1 $langrange_2(xnodos, ynodos, x)$ y asi se obtuvo el polinomio de interpolación. Luego mediante la función 'subplot' de MATLAB la gráfica de la función y del polinomio de interpolación.

EJERCICIO 3

a. Creación de nodos Se utilizó lo siguiente:

*$g(x) = \frac{1}{1+25 \cdot x^2} x : [-1, 1]$ y los puntos equidistantes $es = 2/2000$. Además $y_i = g(x_i)$ para graficar la función.

* $xnodos$ con la ecuación 4 y $ynodos$ será la evaluación de la función $g(xnodos)$.

*$wnodos$ con la ecuación 5 y $ywnodos$ será la evaluación de la función $g(wnodos)$.

lagrange Se utiliza el m-archivo $lagrange_2$ para hallar el polinomio 1 (pol1) con x_i y el polinomio 2 (pol2) con w_i.

Uso de MATLAB para graficar uso el comando "plotz uso las herramientas para que me aparezca en una única gráfica:

*plot(x,y), gráfica la función g(x)
*plot(x,pol1), gráfica el polinomio de interpolación $p_n(x)$ obtenido con los nodos (x_i)

*plot(x,pol2), gráfica el polinomio de interpolación $p_n(x)$ obtenido con los nodos (w_i)

* Se utiliza el comando "*spline*" para hallar el spline cúbico de $g(x)$ Se usa "plot.en MATLAB para observar su comportamiento con respecto a la función.

b. Creación de nodos Se utilizó lo siguiente:

*$g(x) = \frac{1}{1+25\cdot x^2}x : [-1,1]$ y los puntos equidistantes $es = 2/2000$. Además $y_i = g(x_i)$ para graficar la función.

* xn con la ecuación 4 y ynn será la evaluación de la función $g(xn)$. para $n = 10, 20, 40, 80$

lagrange Se utiliza el m-archivo $lagrange_2$ para hallar yn
Uso de MATLAB para graficar uso el comando "plotz uso las herramientas para que me aparezca en una única gráfica:

*plot(x,y), gráfica la función g(x)
*plot(x,yn), gráfica el polinomio de interpolación $p_n(x)$ obtenido con los nodos (x_i)

* Determino la norma infinito del error haciendo uso del comando "$norm(y - yn)$".

c. Creación de nodos Se utilizó lo siguiente:

*$g(x) = \frac{1}{1+25\cdot x^2}x : [-1,1]$ y los puntos equidistantes $es = 2/2000$. Además $y_i = g(x_i)$ para graficar la función.

* xn con la ecuación 5 y ynn será la evaluación de la función $g(xn)$. para cada $n = 10, 20, 40, 80$

lagrange Se utiliza el m-archivo $lagrange_2$ para hallar yn
Uso de MATLAB para graficar uso el comando "plotz uso las herramientas para que me aparezca en una única gráfica:

*plot(x,y), gráfica la función g(x)
*plot(x,yn), gráfica el polinomio de interpolación $p_n(x)$ obtenido con los nodos (w_i) de la ecuación 5.
* Se utiliza el comando "*spline*" para hallar el spline cúbico de $g(x)$ para cada $n = 10, 20, 40, 80$.
*Se usa "plot.en MATLAB para observar su comportamiento con respecto a la función.
* Determino la norma infinito del error haciendo uso del comando "$norm(y - yn)$".

EJERCICIO 4

a. Uso de $h(x)$ y de xi, wi
Se utilizó lo siguiente:

*$h(x) = |senh(x)|\, parax : [-1,1]$ y los nodos x_i y w_i definidos en la ecuaciones 4 y 5 respectivamente. Los puntos equidistantes definidos por $es = 2/2000$. Además $y_i = h(x_i)$ para graficar la función.

* $xnodos$ con la ecuación 4 y $ynodos$ será la evaluación de la función $h(xnodos)$.

*$wnodos$ con la ecuación 5 y $ywnodos$ será la evaluación de la función $h(wnodos)$.

lagrange Se utiliza el m-archivo $lagrange_2$ para hallar el polinomio 2 (pol2) con los nodos w_i.
* Se utiliza el comando "*spline*" para hallar el spline cúbico de $h(x)$ Se usa "plot.en MATLAB para observar su comportamiento con respecto a la función.

Uso de MATLAB para graficar uso el comando "plotz uso las herramientas para que me aparezca en una única gráfica:

*plot(x,y), gráfica la función h(x)
*plot(x,pol), gráfica el polinomio de interpolación $p_n(x)$ obtenido con los nodos (w_i)

Uso MATLAB para hallar error absoluto:

* Se determinó la norma infinito del error haciendo uso del comando "$norm(y - yn)$" para $n = 10, 20, 40, 80$.

3.2. Listado de los m-archivos realizados:

EJERCICIO 1

-$lagrange\_2.m$

EJERCICIO 2

-**a**: $T5\_ejercicio2\_A.m$
-**b**: $T5\_ejercicio2\_B.m$

EJERCICIO 3

-**a**: $T5\_ejercicio3\_A.m$
-**b**: $T5\_ejercicio3\_B.m$
-**c**: $T5\_ejercicio3\_C.m$

EJERCICIO 4

- **a y b**: $T5\_ejercicio4.m$

4. DISCUSIÓN Y RESULTADOS

EJERCICIO 1

Para el ejercicio 1 creamos la función $lagrange_2$ el cual se lo llama de la siguiente manera:

$$function[valor] = lagrange_2(x, y, z)$$

Para el cual los parametros de entrada serán las coordenadas de los nodos en x e y y z será un vector con valores que serán evaluados en polinomio de lagrange y presentados en $valor$ como un vector.

EJERCICIO 2

-**a**: El menor grado del polinoio de interpolación de la función $f(x) = senh(x)enx : [-1, 1]$ es 15. Esto quiere decir que con n =15 tenemos un error menor que 10^{-8} para el polinomio encontrado.

-**b**: Se obtuvo la figura 1 donde se puede apreciar la forma de la función y el polinomio de interpolación y la gráfica del error absoluto en escala logaritmica de base 10.

En la primera gráfica se observa que el polinomio y la función tienen practicamente la misma forma es por eso que el error absoluto es muy pequeño. Sin embargo, en la gráfica del error absoluto se nota que en los extremos el error es mayor disminuyendo hacia el centro.

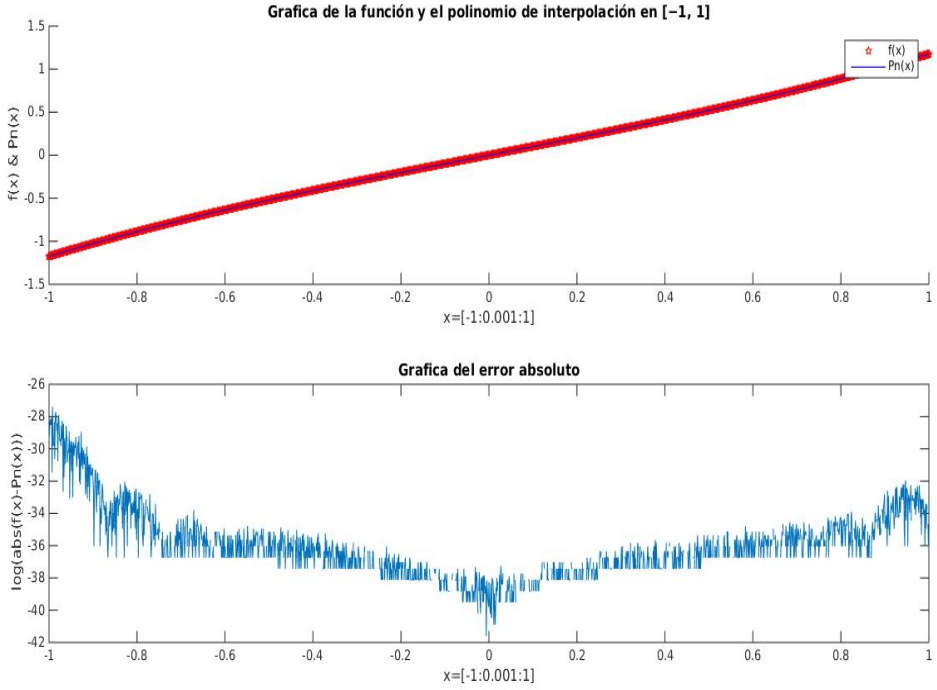

Figura 1: Gráfica de la función f y polinomio de innterpolación en [-1,1]

EJERCICIO 3

-a: Se obtuvo la figura 2 en la cual se muestra la interpolación de cada polinomio con los difrentes nodos.

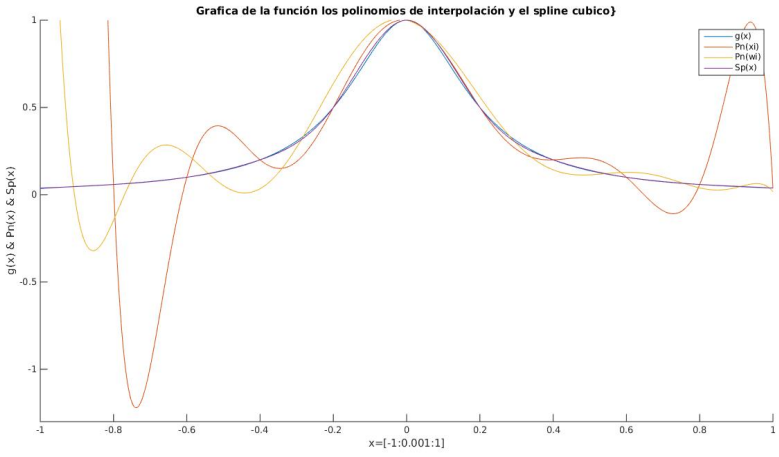

Figura 2: Gráfica de Función g, Polinomios de interpolación con x_i y w_i, Spline cúbico en [-1,1]

Se notó que el polinomio creado a partir de los nodos x_i en el extremo de la función se aleja más que el polinomio creado a partir de los nodos w_i. Esto muestra que los nodos w_i resultaron mucho más eficientes, esto se debe a que son los llamados nodos de Chebyshev los cuales estan desarrolados con la finalidad de que sean los más apropiadas para crear polinomios de interpolación. Sin embrago, la mejor interpolación la realizó el spline cúbico con los nodos x_i. Así podemos concluir que el spline cúbico es el método más acertado para realizar un polinomio de interpolación. Se debe tomar en cuenta que el spline cúbico se lo desarrollo con los nodos x_i los cuales no son tan eficientes como los nodos de Chebyshev denominados w_i.

-b: Se obtuvo una comparación de los errores en la norma infinito para las graficas del polinomio de grado 10, 20, 40 y 80 y la función g, como muestra la figura 3.

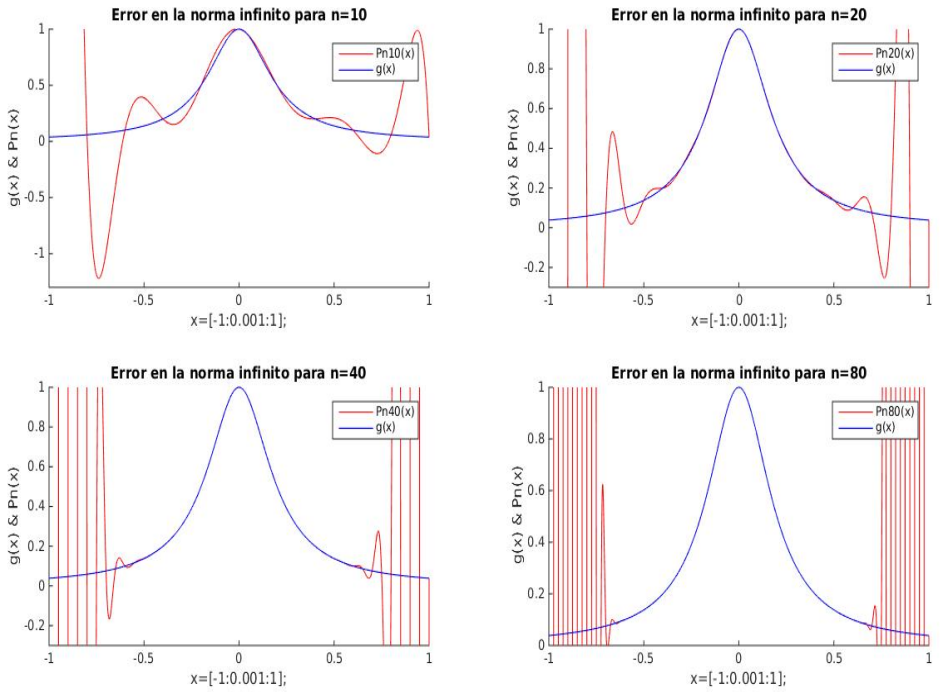

Figura 3: Gráfica de Función g, Polinomios de interpolación con grados 10, 20, 40 y 80

A continuación en el cuadro 1 se muestra el error en la norma infinito con respecto al número de grado del polinomio:

Cuadro 1: Número de grados del polinomio y error en la norma infinito

Número de grado del polinomio	10	20	30	40
Error en la norma infinita	82.099548	6329.891138	26236649.508879	315907842570667.312500

Al observar las gráficas en la figura 3, a medida que se aumenta el grado se notó que en los extremos el polinomio se aleja cada vez más. Esto es conocido como el fenomeno de Runge. En el cuadro 1 se comprobó este fenomeno debido a que el error en la norma infinita aunmenta cuando se incrementa el grado del polinomio.

-c: En la figura 4 se muestra como varía el error en la norma infinito para n =10, 20,40,80 siendo los grados del polinomio. Se realizó una grafica para el polinomio obtenido con lagrange y para el spline cúbico.

Se notó que a medida que aumenta los grados el error disminuye no cumpliendose el fenómeno de Runge para los nodos de Chebyshev denotados como w_i. Al contrario con estos nodos se da una aproximación más cercana a la función orginal. No obstante, el spline cúbico mejora cada vez que aumentamos el grado del polinomio aún cuando estemos usando los nodos x_i que cumplian con el fenomeno. Así, se puede decir una vez más que este último es el mejor método para hallar un polinomio de interpolación.

En el cuadro 2 se puede comprobar los valores de los errores obtenidos del polinomio usando los nodos w_i y el spline cúbico con los nodos x_i, demostrando que el error disminuye a medida que aumenta el grado polinomial. Por lo tanto, siendo el caso del polinomio obtenido a partir del método de lagrange con los nodos w_i se puede decir que estos nodos minimizan el problema del

fenómeno de Runge.

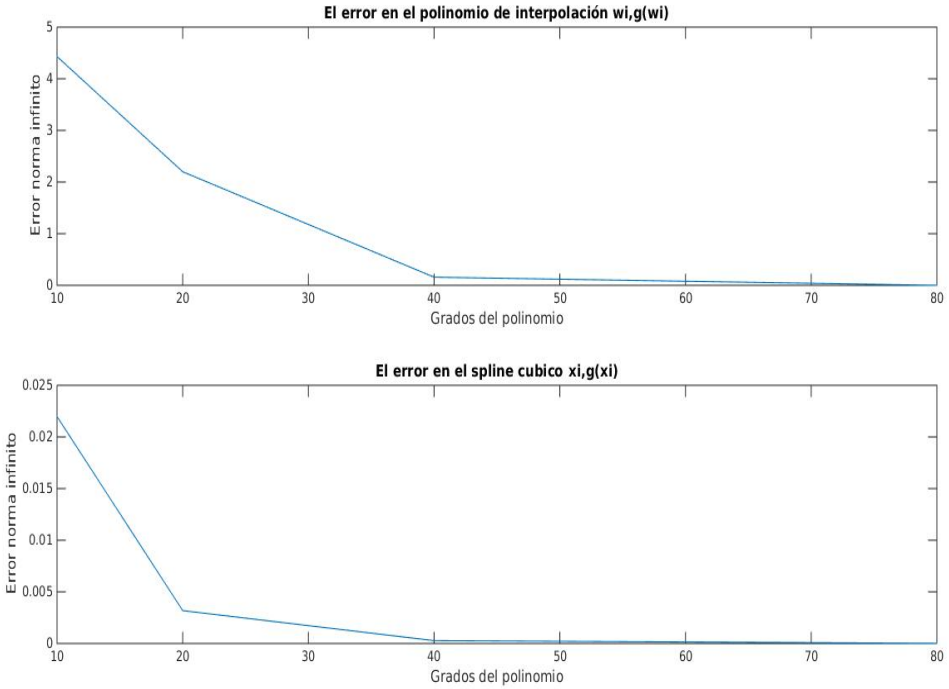

Figura 4: Gráficas de variación del error dependiendo del grado n

Cuadro 2: Error de los métodos lagrange y spline cubico variando el número de grado

Error norma infinito	n=10	n=20	n=40	n=80
Spline cúbico	2.197763e-02	3.182771e-03	2.779680e-04	1.610029e-05
Langrange	4.430319	2.199555	0.157861	0.000218

EJERCICIO 4

-**a**: Se realizó dos gráficas que muestran la función y el polinomio obtenido con el método de lagrange y el spline cúbico respectivamente.

En la figura 5 se muestran las gráficas del polinomio obtenido con el método de lagrange en los grados n =10,20,40,80. Se nota que a medida que se aumenta el grado polinomial el polinomio se aproxima más a la función real. Sin embrago, no se llega a una aproximación óptima a pesar de que se ha usado los nodos de Chebyshev denotados como w_i.

Para el caso del polinomio obtenido con el método del spline cúbico como muestra la figura 6, desde el grado más bajo tenemos una buena aproximación a la función real. A medida que se aumenta el grado se obtiene una aproximación casi exacta a la función. Si comparamos las figuras 5 y 6 se observa que el método del spline cúbico una vez más es mucho más confiable y eficiente que la interpolación poliomial.

En el cuadro 3 se observa los valores de los errores en la norma infinito tanto de la figura 5 como de la figura 6. Para el polinomio obtenido con el método de lagrange en el cuadro 3 se observa claramente que la disminución del error es muy pequeño, esto comprueba el fenómeno de Gibbs, el cual menciona que siempre se va a tener unos picos de error que por más que se aumente el grado polinomial no se eliminarán por completo. En el caso del spline cúbico también se observa que existe un error aunque este es mucho más pequeño en comparación al método de lagrange.

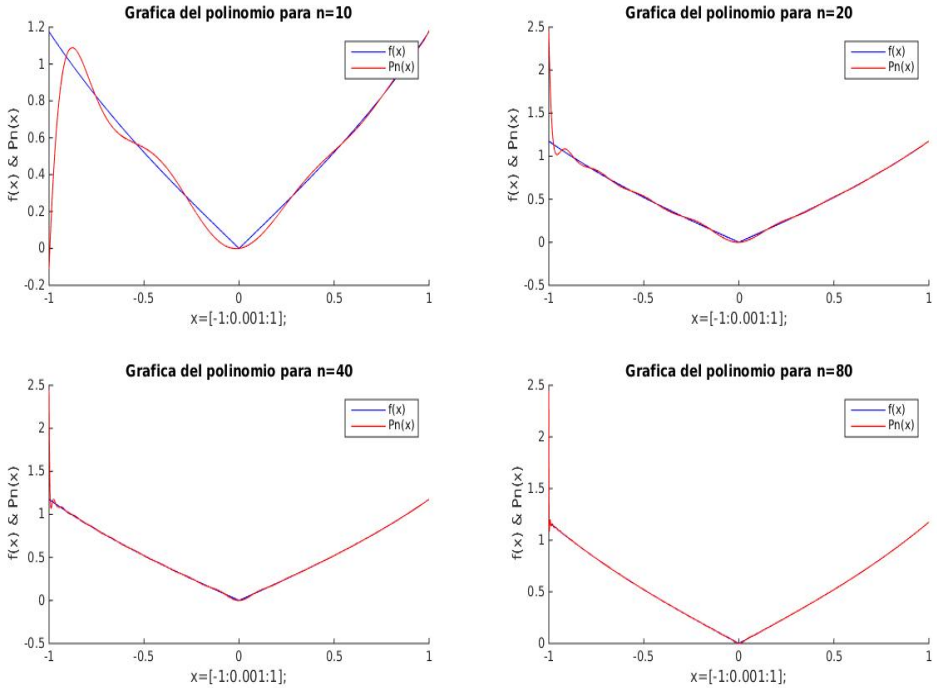

Figura 5: Gráficas del polinomio de interpolación por el método de lagrange para distintos grados

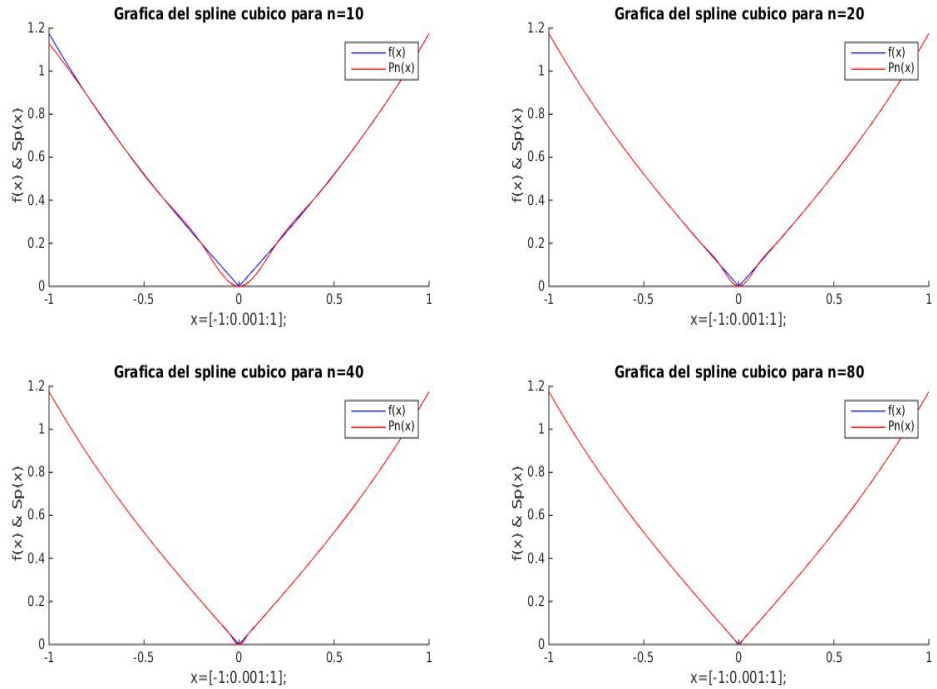

Figura 6: Gráficas del polinomio por el método del spline cúbico para distintos grados

Cuadro 3: Error de los métodos lagrange y spline cubico variando el número de grado

Error norma infinito	n=10	n=20	n=40	n=80
Lagrange	1.288333	1.277400	1.274330	1.273519
Spline cúbico	4.703981e-02	1.700610e-02	8.503048e-03	4.243078e-03

-b: En la figura 7 se pude observar qde el error en la norma infinto para el polinomio obtenido con el método de lagrange y con el spline cúbico para los grados n =10,20,40,80 disminuye a medida que aumentamos el grado.

En la figura 7 se aprecia claramente que los errores del spline cúbico son mucho menores al de lagrange. Sin embargo, en los dos métodos se observa que el error no será un cero exacto debido a que disminuye muy lentamente apegandose al fenomeno de Gibbs, es decir permaneciendo los picos de error.

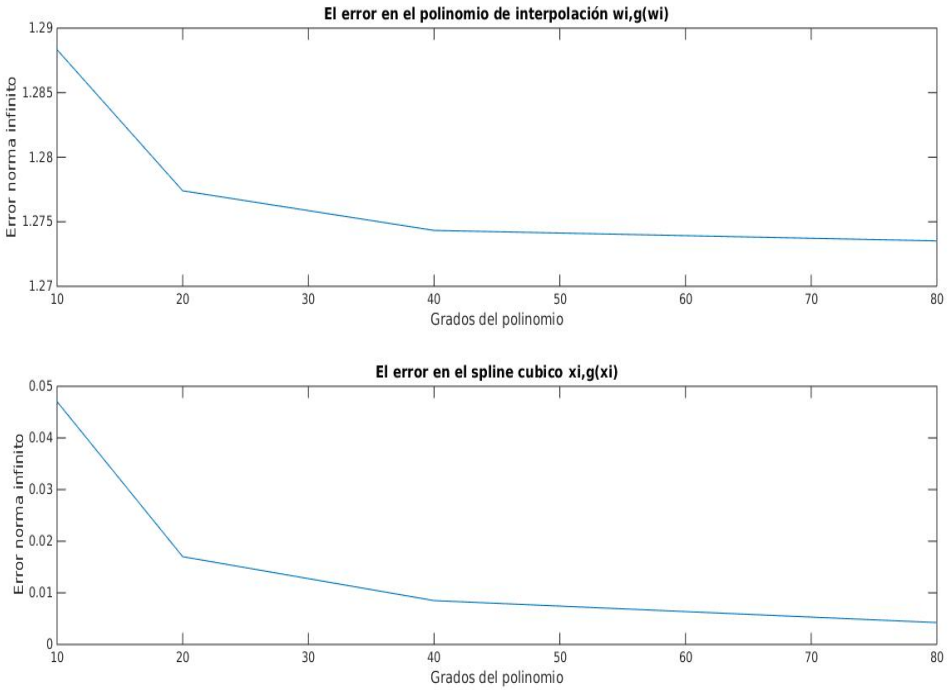

Figura 7: Gráficas del errores de norma infinito versus los polinomios obtenidos por lagrange y spline cúbico

5. CONCLUSIONES

- El método de lagrange ha resultado ser efectivo en la obtención del polinomio de interpolación para una función, debido a que su algoritmo no utiliza muchos recursos computacionales debido a que sus iteraciones contienen operaciones matemáticas sencillas.

- El error máximo en todos los polinomios de interpolación que analizamos se encuentra siempre en los extremos del intervalo del dominio, sin importar que método se utilice.

- Los nodos de Chebyishev minimizan el problema del fenómeno de Runge, es decir, a medida que aumenta el grado polinomial se dismuye el error. Así se puede decir que los nodos de Chebyshef pueden resultar muy eficientes para la realización de un polinomio de interpolación.

- En el ejercicio 4 se observó que se cumple el fenómeno de Gibbs debido a que el error disminuye en el tercer decimal para el método de lagrange y el spline cúbico mostrando así que siempre va a existir un pico de error sin importar cuanto se aumente el grado polinomial.

- Se aprendió que el método del spline cúbico ha resultado ser el más eficiente sin importar que nodos se utilicen y el fenómeno que se esté probando.

6. REFERNCIAS BIBLIOGRÁFICAS

- http://www.matematicasvisuales.com/html/analisis/interpolacion/lagrange.html
- Hazewinkel, Michiel, ed. (2001), «Gibbs phenomenon», Encyclopaedia of Mathematics (en inglés), Springer, ISBN 978-1556080104
- Module for Chebyshov Polynomials by John H. Mathews

7. APENDICE

***lagrange_2**

```
%-------------------------------------------------------------------------
%Nombre del programa: lagrange_2.m
%Autor(es): Emil Vega y Valentina Cordova
%Email del (los) autor(es): airina.cordova@yachaytech.edu.ec y
%emil.vega@yachaytech.edu.ec
%Fecha de elaboracion: Junio 19 de 2016
%Breve descripcion del programa: Crea un polinomio de interpolación para
%una función ----------------------.
%Datos de entrada:
%-----------------
%x: Vector de nodos
%y: Vector de nodos
%z: Vector de valores a evaluar en polinomio de inteporlación
%Datos de salida:
%-----------------
%vector : Vector columna con los polinomios base de Lagrange
%-------------------------------------------------------------------------
function [valor] = lagrange_2(x,y,z)
n = length(x);
if length(y)~= n
    disp('Tamaño de  x e y deben ser el mismo');
    return
end
nz=length(z);
valor = zeros(nz,1);
for k=1:nz
for i = 1:n
    l = y(i);
    for j=1:n
        if i~= j
            l = l*(z(k)-x(j))/(x(i)-x(j));
        end
    end
    valor(k) = valor(k)+l;
end
end
```

***T5_ejercicio2_A**

```
%-------------------------------------------------------------------------
%Nombre del programa: T5\_ejercicio2\_A.m
%Autor(es): Emil Vega y Valentina Cordova
```

```
%Email del (los) autor(es): airina.cordova@yachaytech.edu.ec y
%emil.vega@yachaytech.edu.ec
%Fecha de elaboracion: Junio 19 de 2016
%Breve descripcion del programa: Utiliza la función y sus derivadas para
%obtener el menor grado del polinomio de interpolación__.
%Datos de salida:
%----------------
%n : Grado menor del polinomio de interpolación
%-------------------------------------------------------------------------

f1 = inline('sinh(x)'); %función y a la vez por def. es su segunda derivada.
                                                    %para valores de n pares
f2 = inline('cosh(x)');% primera derivada, para valores de n impares
n=1;
M=abs(f2(1));
r=((factorial(n+1))/2^(n+1));
e=10^(8);
while r<(M*e)
    n=n+1;
    if mod(n,2)==0 %condición para que sea par y se pueda usar f(x).
        M=abs(f1(1));
    else
        M=abs(f2(1));
    end
    r=((factorial(n+1))/2^(n+1));
end
fprintf('El menor grado es = %3i\n',n);
```

***T5_ejercicio2_B**

```
%-------------------------------------------------------------------------
%Nombre del programa: T5\_ejercicio2\_B.m
%Autor(es): Emil Vega y Valentina Cordova
%Email del (los) autor(es): airina.cordova@yachaytech.edu.ec y
%emil.vega@yachaytech.edu.ec
%Fecha de elaboracion: Junio 19 de 2016
%Breve descripcion del programa:Utiliza puntos equidistantes para graficar
%la función y el polinomio de interpolación en [-1, 1]. Adicionalmente gráfica
%el error absoluto
%Datos de entrada:
%----------------
%
%Datos de salida:
%----------------
%plot(x,y,'rp'): grafica f(x)=senh(x)
%plot(x,pol,'b'):grafica p_n
%plot(x,log(err)): Gráfica el error absoluto de la función.\\

%Para hallar los 2001 puntos\\ equidistantes de x que pertenece a [1,-1]\\

f1 = inline('sinh(x)');
f2 = inline('cosh(x)');
n=1;
M=abs(f2(1));
r=((factorial(n+1))/2^(n+1));
e=10^(8);
while r<(M*e)
    n=n+1;
    if mod(n,2)==0
```

```
        M=abs(f1(1));
    else
        M=abs(f2(1));
    end
    r=((factorial(n+1))/2^(n+1));
end
fprintf('El menor grado es = %3i\n',n);
```

*__T5_ejercicio3_A__

```
%-------------------------------------------------------------------------
%Nombre del programa: T5\_ejercicio3\_A.m
%Autor(es): Emil Vega y Valentina Cordova
%Email del (los) autor(es): airina.cordova@yachaytech.edu.ec y
%emil.vega@yachaytech.edu.ec
%Fecha de elaboración: Junio 19 de 2016
%Breve descripción del programa:Realiza la gráfica de g, los polinomios de
%interpolación y el spline cúbico para i=(0,..,10)
%Datos de entrada:
%-----------------
%
%Datos de salida:
%----------------
%plot(x,y): Gráfica de g(x)
%plot(x,pol1): Gráfica de polinomio de interpolación usando los nodos 'xi'
%plot(x,pol2):Gráfica de polinomio de interpolación usando los nodos 'wi'
%plot(x,sp):Gráfica el comportamiento del spline con respecto a g(x).
%-------------------------------------------------------------------------

g=inline('1/(1+25*x^2)');
n=10;
es=2/2000;
x=[-1:es:1];
y=zeros(length(x),1);
for i=1:length(x)
    y(i)=g(x(i));
end
xnodos=zeros(n,1);
ynodos=zeros(n,1);
for i=1:n
    xnodos(i)=-1+(2*i)/n;
    ynodos(i)=g(xnodos(i));
end
wnodos=zeros(n,1);
ywnodos=zeros(n,1);
for i=1:n
    wnodos(i)=cos(((2*(n-i)+1)*pi)/(2*(n+1)));
    ywnodos(i)=g(wnodos(i));
end
pol1=lagrange_2(xnodos,ynodos,x);
pol2=lagrange_2(wnodos,ywnodos,x);
sp = spline(xnodos,ynodos,x);
hold on
plot(x,y)
plot(x,pol1)
plot(x,pol2)
plot(x,sp)
axis([-1 1 -1.3 1])
title('Grafica de la función los polinomios de interpolación y el spline cubico')
```

```
xlabel('x=[-1:0.001:1]')
ylabel('g(x) & Pn(x) & Sp(x)')
legend ('g(x)','Pn(xi)','Pn(wi)','Sp(x)')
hold off
```

*__T5_ejercicio3_B__

```
%-------------------------------------------------------------------------
%Nombre del programa: T5\_ejercicio3\_B.m
%Autor(es): Emil Vega y Valentina Cordova
%Email del (los) autor(es): airina.cordova@yachaytech.edu.ec y
%emil.vega@yachaytech.edu.ec
%Fecha de elaboración: Junio 19 de 2016
%Breve descripción del programa: gráfica el error en la  norma infinito
%en la aproximación de g
%por el polinomio hecho con los nodos $xi$.
%Datos de entrada:
%-----------------
%
%Datos de salida:
%----------------
%Para n=(10,20,40,80) se hacen las gráficas por separado y respectivamente
%plot(x,yn,'r'): gráfica el polinomio en el grado 'n'.
%plot(x,y,'b'): gráfica la función g(x).
%fprintf('El error en la norma infinito para n= () es: %f \n',en):
%Este mensaje me muestra según el valor del grado del polinomio 'n',
%el error en la norma infinito.

%-------------------------------------------------------------------------

g=inline('1/(1+25*x^2)');
n10=10;
es=2/2000;
x=[-1:es:1];
y=zeros(length(x),1);
for i=1:length(x)
    y(i)=g(x(i));
end
x10=zeros(n10,1);
yn10=zeros(n10,1);
for i=1:n10
    x10(i)=-1+(2*i)/n10;
    yn10(i)=g(x10(i));
end
y10=lagrange_2(x10,yn10,x);
e10 = norm((y-y10),'inf');
subplot(2,2,1)
hold on
plot(x,y10,'r')
plot(x,y,'b')
axis([-1 1 -1.3 1])
title('Error en la norma infinito para n=10')
xlabel('x=[-1:0.001:1];')
ylabel('g(x) & Pn(x)')
legend('Pn10(x)','g(x)')
hold off
fprintf('El error en la norma infinito para n=10 es: %f \n',e10)

%=========================================================================
```

```
n20=20;
x20=zeros(n20,1);
yn20=zeros(n20,1);
for i=1:n20
    x20(i)=-1+(2*i)/n20;
    yn20(i)=g(x20(i));
end
y20=lagrange_2(x20,yn20,x);
e20 = norm((y-y20),'inf');
subplot(2,2,2)
hold on
plot(x,y20,'r')
plot(x,y,'b')
axis([-1 1 -0.3 1])
title('Error en la norma infinito para n=20')
xlabel('x=[-1:0.001:1];')
ylabel('g(x) & Pn(x)')
legend('Pn20(x)','g(x)')
hold off
fprintf('El error en la norma infinito para n=20 es: %f \n',e20)

%=========================================================================

n40=40;
x40=zeros(n40,1);
yn40=zeros(n40,1);
for i=1:n40
    x40(i)=-1+(2*i)/n40;
    yn40(i)=g(x40(i));
end
y40=lagrange_2(x40,yn40,x);
e40 = norm((y-y40),'inf');
subplot(2,2,3)
hold on
plot(x,y40,'r')
plot(x,y,'b')
axis([-1 1 -0.3 1])
title('Error en la norma infinito para n=40')
xlabel('x=[-1:0.001:1];')
ylabel('g(x) & Pn(x)')
legend('Pn40(x)','g(x)')
hold off
fprintf('El error en la norma infinito para n=40 es: %f \n',e40)

%=========================================================================

n80=80;
x80=zeros(n80,1);
yn80=zeros(n80,1);
for i=1:n80
    x80(i)=-1+(2*i)/n80;
    yn80(i)=g(x80(i));
end
y80=lagrange_2(x80,yn80,x);
e80 = norm((y-y80),'inf');
subplot(2,2,4)
hold on
```

```
plot(x,y80,'r')
plot(x,y,'b')
axis([-1 1 -0 1])
title('Error en la norma infinito para n=80')
xlabel('x=[-1:0.001:1];')
ylabel('g(x) & Pn(x)')
legend('Pn80(x)','g(x)')
hold off
fprintf('El error en la norma infinito para n=80 es: %f \n',e80)
```

***T5_ejercicio3_C**

```
%-------------------------------------------------------------------------
%Nombre del programa: T5\_ejercicio3\_C.m
%Autor(es): Emil Vega y Valentina Cordova
%Email del (los) autor(es): airina.cordova@yachaytech.edu.ec y
%emil.vega@yachaytech.edu.ec
%Fecha de elaboración: Junio 19 de 2016
%Breve descripción del programa: gráfica el error absoluto en la aproximación de g
%por el polinomio hecho con los nodos $wi$. Además, el spline cúbico de (xi,g(xi))
%Datos de entrada:
%-----------------
%
%Datos de salida:
%----------------
%Para n=(10,20,40,80) se hacen las gráficas por separado y respectivamente
%plot(nm,error): El error en el polinomio de interpolación {wi,g(wi)
%plot(nm,errorsp):El error en el spline cubico {xi,g(xi)}

%-------------------------------------------------------------------------
g=inline('1/(1+25*x^2)');
n10=10;
es=2/2000;
x=[-1:es:1];
y=zeros(length(x),1);
for i=1:length(x)
    y(i)=g(x(i));
end

x10=zeros(n10,1);
yn10=zeros(n10,1);
xs10=zeros(n10,1);
yns10=zeros(n10,1);
for i=1:n10
    x10(i)=cos(((2*(n10-i)+1)*pi)/(2*(n10+1)));
    yn10(i)=g(x10(i));
    xs10(i)=-1+(2*i)/n10;
    yns10(i)=g(xs10(i));
end
y10=lagrange_2(x10,yn10,x);
e10 = norm((y-y10),'inf');
sp10=spline(xs10,yns10,x);
es10 = norm((y-sp10'),'inf');
fprintf('El error para n=10 en el polinomio de lagrange es: %f\n', e10)
fprintf('El error para n=10 en el spline cubico es: %d\n',es10)

%=========================================================================
```

```
n20=20;
x20=zeros(n20,1);
yn20=zeros(n20,1);
xs20=zeros(n20,1);
yns20=zeros(n20,1);
for i=1:n20
    x20(i)=cos(((2*(n20-i)+1)*pi)/(2*(n20+1)));
    yn20(i)=g(x20(i));
    xs20(i)=-1+(2*i)/n20;
    yns20(i)=g(xs20(i));
end
y20=lagrange_2(x20,yn20,x);
e20 = norm((y-y20),'inf');
sp20=spline(xs20,yns20,x);
es20 = norm((y-sp20'),'inf');
fprintf('El error para n=20 en el polinomio de lagrange es: %f\n', e20)
fprintf('El error para n=20 en el spline cúbico es: %d\n',es20)

%=========================================================================

n40=40;
x40=zeros(n40,1);
yn40=zeros(n40,1);
xs40=zeros(n40,1);
yns40=zeros(n40,1);
for i=1:n40
    x40(i)=cos(((2*(n40-i)+1)*pi)/(2*(n40+1)));
    yn40(i)=g(x40(i));
    xs40(i)=-1+(2*i)/n40;
    yns40(i)=g(xs40(i));
end
y40=lagrange_2(x40,yn40,x);
e40 = norm((y-y40),'inf');
sp40=spline(xs40,yns40,x);
es40 = norm((y-sp40'),'inf');
fprintf('El error para n=40 en el polinomio de lagrange es: %f\n', e40)
fprintf('El error para n=40 en el spline cubico es: %d\n',es40)

%=========================================================================

n80=80;
x80=zeros(n80,1);
yn80=zeros(n80,1);
xs80=zeros(n80,1);
yns80=zeros(n80,1);
for i=1:n80
    x80(i)=cos(((2*(n80-i)+1)*pi)/(2*(n80+1)));
    yn80(i)=g(x80(i));
    xs80(i)=-1+(2*i)/n80;
    yns80(i)=g(xs80(i));
end
y80=lagrange_2(x80,yn80,x);
e80 = norm((y-y80),'inf');
sp80=spline(xs80,yns80,x);
es80 = norm((y-sp80'),'inf');
fprintf('El error para n=80 en el polinomio de lagrange es: %f\n', e80)
fprintf('El error para n=80 en el spline cubico es: %d\n',es80)
```

```
%=========================================================================

error=[e10;e20;e40;e80];
errorsp = [es10;es20;es40;es80];
nm = [10;20;40;80];
subplot(2,1,1)
plot(nm,error)
title('El error en el polinomio de interpolación {wi,g(wi)}')
xlabel('Grados del polinomio')
ylabel('Error norma infinito')
subplot(2,1,2)
plot(nm,errorsp)
title('El error en el spline cubico {xi,g(xi)}')
xlabel('Grados del polinomio')
ylabel('Error norma infinito')
```

*__Ejercicio 4__

```
%-------------------------------------------------------------------------
%Nombre del programa: T5\_ejercicio4.m
%Autor(es): Emil Vega y Valentina Cordova
%Email del (los) autor(es): airina.cordova@yachaytech.edu.ec y
%emil.vega@yachaytech.edu.ec
%Fecha de elaboración: Junio 19 de 2016
%Breve descripción del programa:  Gráfica la función $h$, el polinomio de
%interpolación de los puntos {(wi, h(wi))} y el spline cúbico de
%los puntos {(xi, h(xi))}ni =0 para n = (10, 20, 40, 80). Además, grafica
%el error absoluto en la aproximación de h con los polinomios definidos antes.
%Datos de entrada:
%-----------------
%
%Datos de salida:
%-----------------
%*fprintf('El error para n=() en el polinomio de lagrange es: %f\n', e e(n):Devuelve
en el polinomio de lagrange para el grado de polinomio respectivo. ej si n=10, entonc
%*fprintf('El error para n=() en el spline cubico es: %d\n',esn): Devuelve el error
%*plot(x,y,'b'): Gráfica de la función h(x).
%plot(x,sp80,'r'): Gráfica del spline cubico para n= (10,20,40,80).
%plot(nm,errorsp): Gráfica un vector nm= [10, 20, 40, 80] vs los errores en el polino


------------------------------------------------------------------------
h=inline('abs(sinh(x))');
n10=10;
es=2/2000;
x=[-1:es:1];
y=zeros(length(x),1);
for i=1:length(x)
    y(i)=h(x(i));
end

x10=zeros(n10,1);
yn10=zeros(n10,1);
xs10=zeros(n10,1);
yns10=zeros(n10,1);
for i=1:n10
    x10(i)=cos(((2*(n10-i)+1)*pi)/(2*(n10+1)));
    yn10(i)=h(x10(i));
```

```
    xs10(i)=-1+(2*i)/n10;
    yns10(i)=h(xs10(i));
end
y10=lagrange_2(x10,yn10,x);
e10 = norm((y-y10),'inf');
sp10=spline(xs10,yns10,x);
es10 = norm((y-sp10'),'inf');
fprintf('El error para n=10 en el polinomio de lagrange es: %f\n', e10)
fprintf('El error para n=10 en el spline cubico es: %d\n',es10)

%=========================================================================

n20=20;
x20=zeros(n20,1);
yn20=zeros(n20,1);
xs20=zeros(n20,1);
yns20=zeros(n20,1);
for i=1:n20
    x20(i)=cos(((2*(n20-i)+1)*pi)/(2*(n20+1)));
    yn20(i)=h(x20(i));
    xs20(i)=-1+(2*i)/n20;
    yns20(i)=h(xs20(i));
end
y20=lagrange_2(x20,yn20,x);
e20 = norm((y-y20),'inf');
sp20=spline(xs20,yns20,x);
es20 = norm((y-sp20'),'inf');
fprintf('El error para n=20 en el polinomio de lagrange es: %f\n', e20)
fprintf('El error para n=20 en el spline cubico es: %d\n',es20)

%=========================================================================

n40=40;
x40=zeros(n40,1);
yn40=zeros(n40,1);
xs40=zeros(n40,1);
yns40=zeros(n40,1);
for i=1:n40
    x40(i)=cos(((2*(n40-i)+1)*pi)/(2*(n40+1)));
    yn40(i)=h(x40(i));
    xs40(i)=-1+(2*i)/n40;
    yns40(i)=h(xs40(i));
end
y40=lagrange_2(x40,yn40,x);
e40 = norm((y-y40),'inf');
sp40=spline(xs40,yns40,x);
es40 = norm((y-sp40'),'inf');
fprintf('El error para n=40 en el polinomio de lagrange es: %f\n', e40)
fprintf('El error para n=40 en el spline cubico es: %d\n',es40)

%=========================================================================

n80=80;
x80=zeros(n80,1);
yn80=zeros(n80,1);
xs80=zeros(n80,1);
yns80=zeros(n80,1);
for i=1:n80
```

```
    x80(i)=cos(((2*(n80-i)+1)*pi)/(2*(n80+1)));
    yn80(i)=h(x80(i));
    xs80(i)=-1+(2*i)/n80;
    yns80(i)=h(xs80(i));
end
y80=lagrange_2(x80,yn80,x);
e80 = norm((y-y80),'inf');
sp80=spline(xs80,yns80,x);
es80 = norm((y-sp80'),'inf');
fprintf('El error para n=80 en el polinomio de lagrange es: %f\n', e80)
fprintf('El error para n=80 en el spline cubico es: %d\n',es80)

%GRAFICAS
%Polinomio de Interpolación Wi
figure, subplot(2,2,1)
hold on
plot(x,y,'b')
plot(x,y10,'r')
title('Grafica del polinomio para n=10')
xlabel('x=[-1:0.001:1];')
ylabel('f(x) & Pn(x)')
legend('f(x)','Pn(x)')
hold off
subplot(2,2,2)
hold on
plot(x,y,'b')
plot(x,y20,'r')
title('Grafica del polinomio para n=20')
xlabel('x=[-1:0.001:1];')
ylabel('f(x) & Pn(x)')
legend('f(x)','Pn(x)')
hold off
subplot(2,2,3)
hold on
plot(x,y,'b')
plot(x,y40,'r')
title('Grafica del polinomio para n=40')
xlabel('x=[-1:0.001:1];')
ylabel('f(x) & Pn(x)')
legend('f(x)','Pn(x)')
hold off
subplot(2,2,4)
hold on
plot(x,y,'b')
plot(x,y80,'r')
title('Grafica del polinomio para n=80')
xlabel('x=[-1:0.001:1];')
ylabel('f(x) & Pn(x)')
legend('f(x)','Pn(x)')
hold off

%Spline cubico
figure, subplot(2,2,1)
hold on
plot(x,y,'b')
plot(x,sp10,'r')
title('Grafica del spline cubico para n=10')
xlabel('x=[-1:0.001:1];')
```

```
ylabel('f(x) & Sp(x)')
legend('f(x)','Pn(x)')
hold off
subplot(2,2,2)
hold on
plot(x,y,'b')
plot(x,sp20,'r')
title('Grafica del spline cubico para n=20')
xlabel('x=[-1:0.001:1];')
ylabel('f(x) & Sp(x)')
legend('f(x)','Pn(x)')
hold off
subplot(2,2,3)
hold on
plot(x,y,'b')
plot(x,sp40,'r')
title('Grafica del spline cubico para n=40')
xlabel('x=[-1:0.001:1];')
ylabel('f(x) & Sp(x)')
legend('f(x)','Pn(x)')
hold off
subplot(2,2,4)
hold on
plot(x,y,'b')
plot(x,sp80,'r')
title('Grafica del spline cubico para n=80')
xlabel('x=[-1:0.001:1];')
ylabel('f(x) & Sp(x)')
legend('f(x)','Pn(x)')
hold off

%=========================================================

error=[e10;e20;e40;e80];
errorsp = [es10;es20;es40;es80];
nm = [10;20;40;80];
figure, subplot(2,1,1)
plot(nm,error)
title('El error en el polinomio de interpolación {wi,g(wi)}')
xlabel('Grados del polinomio')
ylabel('Error norma infinito')
subplot(2,1,2)
plot(nm,errorsp)
title('El error en el spline cubico {xi,g(xi)}')
xlabel('Grados del polinomio')
ylabel('Error norma infinito')

%-------------------------------------------------------------------------
```