Interpolación para nodos uniformemente espaciados

María Teresa Hoyo García
Roberto Hueso Gómez
Abel José Sánchez Alba

29 de Mayo de 2015

Índice

1. Interpolación

Un problema que se presenta con frecuencia en las ciencias experimentales y en ingeniería es tratar de construir una función (denominada "función interpolante") de la que se conoce una serie de datos (denominados "datos de interpolación"). Estos datos pueden ser fruto de las observaciones realizadas en un determinado experimento en el que se relacionan dos o más variables e involucran valores de una función y/o de sus derivadas.

El objetivo será determinar una función que verifique estos datos y que además sea fácil de construir y manipular. Por su sencillez y operatividad los polinomios se usan frecuentemente como funciones interpolantes.

Denominaremos nodos al conjunto de datos x_i, $i \in \mathbb{N}$ pertenecientes a una función y a partir de estos hallaremos la función interpolante.

Con el fin de plantear una solución a este problema se han desarrollado diversos métodos:

1.1. El problema de Lagrange

Conocidos los valores de una función f en n+1 puntos distintos (x_i, $i = 0, 1, \ldots, n$) de un intervalo [a, b], nos planteamos obtener un polinomio $P(x)$ de grado no superior a n, que coincida con la función f en estos n+1 puntos.

$$P(x) : Grado(P(x)) \leq n$$

Así con 3 nodos y conociendo su valor podemos formular un polinomio de grado menor o igual que 2.
En el caso general, la distancia entre los nodos x_i puede ser arbitraria aunque en este caso estudiaremos los nodos equiespaciados. Sea $h \in \mathbb{R}^+$ denominamos **nodo equidistante** a todo conjunto de $n + 1$ nodos generados a partir de un nodo x_0 tal que:

$$x_i = x_0 + ih \quad \forall i \in \mathbb{N} \tag{1}$$

En primer lugar estudiaremos el caso de interpolación con polinomios de potencias clásicas. Para resolver el problema se debe resolver un sistema de n ecuaciones con n incógnitas siendo n el número de nodos equiespaciados.
El problema tendrá solución única si la matriz de Gram:

$$A = \begin{pmatrix} 1 & x_0 & \cdots & x_0^n \\ 1 & \vdots & \ddots & \vdots \\ 1 & x_n & \cdots & x_n^n \end{pmatrix}$$

Tiene determinante $det(A) \neq 0$ donde

$$det(A) = \prod_{i>j} (x_i - x_j)$$

Se puede deducir fácilmente que tiene solución única si los nodos x_i son distintos.
El problema consiste en hallar los valores de a_i de manera que

$$P_n(x) = a_n x^n + a_{n-1} x^{n-1} \ldots a_0 \ \forall a \in \mathbb{R}$$

Sin embargo, al aumentar el número de nodos, este polinomio comienza a comportarse de manera abrupta con el inconveniente añadido de que será necesario resolver un sistema de ecuaciones de mayor dimensión y por lo tanto aumentará el coste computacional de la operación.

1.2. Diferencias progresivas y regresivas

1.2.1. Diferencias progresivas

Consideremos la función $f : \{x_i\}_{i\in\mathbb{N}} \to \mathbb{R}$ de manera que la distancia Δx (De ahora en adelante la llamaremos h)entre los nodos x_i sea un valor constante (estén equiespaciados) y $h \geq 0$ llamamos diferencia progresiva de f en x_i a

$$\Delta f(x_k) = f(x_{k+h}) - f(x_k)$$

Para simplificar la notación $\Delta f_k = \Delta f(x_k)$

Dado que la diferencia progresiva puede calcularse para todo k se tiene que

$$\Delta(\Delta f_k) = \Delta f_{k+1} - \Delta f_k = \Delta^2 f_k$$

Luego de forma general de define

$$\Delta^{n+1} f_k = \Delta(\Delta^n f_k) - \Delta^n f_k$$

A $\Delta^n f_k$ se le llama diferencia progresiva de orden n de f en x_k.
Se puede construir así una tabla de diferencias progresivas:

f_0	0	0	0	0	0
f_1	Δf_0	0	0	0	0
f_2	Δf_1	$\Delta^2 f_0$	0	0	0
f_3	Δf_2	$\Delta^2 f_1$	$\Delta^3 f_0$	0	0
f_4	Δf_3	$\Delta^2 f_2$	$\Delta^3 f_1$	$\Delta^4 f_0$	0
f_5	Δf_4	$\Delta^2 f_3$	$\Delta^3 f_2$	$\Delta^4 f_1$	$\Delta^5 f_0$

1.2.2. Diferencias regresivas

Análogamente se definen las diferencias regresivas.

$$\nabla f_k = f_k - f_{k-1}$$

$$\nabla^{n+1} f_k = \nabla(\nabla^n f_k) = \nabla^n f_k - \nabla^n f_{k-1}$$

Y se cumple entonces que

$$\nabla f_k = \Delta f_{k-1} \Longrightarrow \nabla^n f_k = \Delta^n f_{k-n}$$

De esta manera, los valores que obtengamos en la tabla de diferencias progresivas, serán los mismos valores que en la tabla de diferencias regresivas: $\Delta f_0 = \nabla f_1, \Delta f_1 = \nabla f_2 ...$

1.2.3. Propiedades

Tanto las diferencias progresivas como las diferencias regresivas, como las diferencias centrales, se las conoce con el nombre de **diferencias finitas**. Cabe destacar las siguientes 2 propiedades de las diferencias finitas.

1. Relación entre diferencias finitas y diferencias divididas:

$$\Delta^n f_k = n!h^n f[x_k, x_{k+1}..., x_{k+n}] \ \forall n \geq 0 \tag{2}$$

2. $\forall n \geq 0$

$$\Delta^n f_k = \sum_{i=0}^{n} (-1)^{n-i} \binom{n}{i} f_{k+i}$$

1.3. Fórmulas de interpolación de Newton

1.3.1. Fórmula progresiva

Expresión del polinomio de interpolación usando las diferencias finitas partiendo de Newton

$$P(x) = \sum_{i=0}^{n} f[x_0, ..., x_i] \prod_{j=0}^{i-1} (x - x_j) \tag{3}$$

y sabiendo que los puntos x_i son equidistantes (1), mediante (2), podemos expresar $P(x)$ como

$$P(x) = \sum_{i=0}^{k} \frac{\Delta^i f_0}{i!h^i} \prod_{j=0}^{i-1} (x - x_j)$$

es sencillo, haciendo un cambio de variable $x = x_0 + th$ deducir:

$$P(x_0 + th) = \sum_{i=0}^{n} \frac{\Delta^i f_0}{i!} \prod_{j=0}^{i-1} (t - j) = P(x) \tag{4}$$

A ésta ultima expresión (4) se la conoce por **la fórmula de Newton progresiva**.
También hemos visto a (4) expresada como

$$P(x) = \sum_{i=0}^{n} \binom{t}{i} \Delta^i f_0$$

sabiendo, eso si, que abusamos del lenguaje e interpretamos

$$\binom{t}{i} = \frac{t(t-1)...(t+1-i)}{i!}$$

1.3.2. Fórmula regresiva

El polinomio $P(x)$ interpola análogamente a f en orden inverso de los nodos $(x_n, x_{n-1}, x_{n-2}...x_0)$ de manera que

$$P(x) = \sum_{i=0}^{n} f[x_n, x_{n-1}, ..., x_{n-i}] \prod_{j=0}^{i-1} (x - x_{n-j})$$

y de nuevo al ser equidistantes los nodos (1), podemos expresarlo como

$$P(x) = \sum_{i=0}^{n} \frac{\nabla^i f_n}{i!h^i} \prod_{j=0}^{i-1} (x - x_{n-j}) \tag{5}$$

A esta expresión (5) se la denomina **Fórmula de Newton regresiva**
Análogamente al caso anterior, si hacemos el cambio de variable $x = x_n + th$ entonces

$$P(x_n + th) = \sum_{i=0}^{n} \binom{t+i-1}{i} \nabla^i f_n = P(x)$$

eso si, abusando del lenguaje de manera que se entiende

$$\binom{t+i-1}{i} = \frac{(t+i-1)(t+i-2)...t}{i!}$$

1.4. Ejemplos

1.4.1. Problema de Lagrange

Enunciado: Sean los puntos $x_0 = -2, x_1 = -1, x_2 = 0, x_3 = 1$ y sean sus respectivas imágenes $f(x_0) = -2, f(x_1) = -1, f(x_2) = 0, f(x_3) = -5$. Escríbase la expresión del polinomio de interpolación de f según el problema de Lagrange.

Solución:

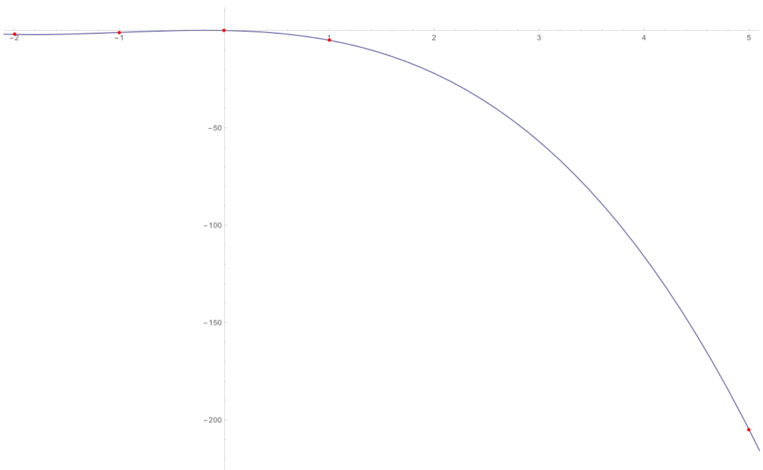
$$\begin{pmatrix} 1 & x_0 & x_0^2 & x_0^3 \\ 1 & x_1 & x_1^2 & x_1^3 \\ 1 & x_2 & x_2^2 & x_2^3 \\ 1 & x_3 & x_3^2 & x_3^3 \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{pmatrix} = \begin{pmatrix} -2 \\ -1 \\ 0 \\ -5 \end{pmatrix}$$

Al resolver el sistema de ecuaciones resultante, obtenemos que los coeficientes

$$a_0 = 0, a_1 = -1, a_2 = -3, a_3 = -1$$

Por lo tanto el polinomio interpolante es

$$f(x) = -x^3 - 3x^2 - x$$



1.4.2. Diferencias progresivas, regresivas y fórmula de Newton

Enunciado: Construir la tabla de diferencias progresivas de $f(x) = x^3$ para los puntos $x_0 = 0, x_1 = 2, x_2 = 4, x_3 = 6$. Construir las fórmulas de Newton progresiva y regresiva.

Solución:

x_i	$f(x_i)$	Δf	$\Delta^2 f$	$\Delta^3 f$
0	0			
2	8	8		
4	64	56	48	
6	216	152	96	48

De donde deducimos:

1. Fórmula progresiva:

$$P(x) = 4x + 6x(x-2) + x(x-2)(x-4)$$

2. Fórmula regresiva:

$$P(x) = 216 + 76x(x-6) + 12(x-6)(x-4) + (x-6)(x-4)(x-2)$$

1.5. Implementación en GNU Octave

1.5.1. Problema de Lagrange

```
function q = LaGrange(x,y)
  n=length(x);
  A=ones(n);
  for I=1:1:n
    for J=1:1:n
      A(I,J)=A(I,J)*(x(I)^(J-1));
      A
    endfor
  endfor
  A
  B=A';
  inv(B)
  q=y*inv(B)
endfunction

#Ejemplo:
#x=[0 1 3 4];
#y=[0 1 27 64];
#LaGrange(x,y)
```

Esta función devuelve los coeficientes que multiplican a

$$1, x, x^2 ...$$

1.5.2. Diferencias progresivas

```
function q = DifProg(x,y,h)
  n=length(y);
  A=zeros(n,n+1);
  for j=1:1:n
    A(j,1)=x(j);
  endfor
  for j=1:1:n
    A(j,2)=y(j);
  endfor
  for i=3:1:n+1
    for j=i-1:1:n
      A(j,i)=A(j,i-1)-A(j-1,i-1);
    endfor
  endfor
  B=zeros(1,n)
  for k=2:1:n+1
    i=k-1;
    B(1,k-1)=(A(i,k))/((factorial(k-2))*(h^(k-2)));
  endfor
  for k=1:1:n
    disp(B(1,k))
  endfor
endfunction

#EJEMPLO:
#x=[0 2 4 6];
#y=[0 8 64 216];
#h=2;
#DifProg(x,y,h)
```

Esta función devuelve los coeficientes que multiplican a

$$1, (x-x_1), (x-x_1)*(x-x_2)...$$

1.5.3. Diferencias regresivas

```
function q = DifReg(x,y,h)
  n=length(y);
  A=zeros(n,n+1);
  for j=1:1:n
    A(j,1)=x(j);
  endfor
  for j=1:1:n
    A(j,2)=y(j);
  endfor
  A
  for j=3:1:n+1
    for i=n:-1:j-1
      A(i,j)=A(i,j-1)-A(i-1,j-1);
      A
    endfor
```

```
  endfor
  B=zeros(1,n);
  for k=2:1:n+1
    B(1,k-1)=(A(n,k))/((factorial(k-2))*(h^(k-2)));
  endfor
  for k=1:1:n
    disp(B(1,k))
  endfor
endfunction

#EJEMPLO:
#x=[0 2 4 6];
#y=[0 8 64 216];
#h=2;
#DifReg(x,y,h)
```

Esta función devuelve los coeficientes que multiplican a

$$1, (x - x_n), (x - x_n) * (x - x_{n-1})...$$

2. Interpolación por recurrencia

La fórmula de Newton es un ejemplo de métodos recurrentes para resolver un problema de interpolación. Otro procedimiento recurrente, evita el cálculo explícito de diferencias divididas utilizando dos polinomios de interpolación, se basa en el lema de Aitken.

2.1. Algoritmo de Aitken

Sean $N = Grado(P(x))$, A un punto en el dominio del polinomio interpolante y conozcamos los nodos y sus respectivas evaluaciones siendo $x_0, x_1...x_n$ es posible obtener el valor de la evaluación del polinomio en el punto A siguiendo el Algoritmo de Aitken:

1. Sea $I = 0, 1, ...N \quad \forall N > 0$ definimos

$$F(I) = P(I)$$

2. Sea $Z = 1, 2, ..., N$ tomamos $I = Z, Z + 1, ..., N$ de manera que

$$P(I) = \frac{(X(I) - A) * P(Z - 1) - (X(Z - 1) - A) * P(I)}{X(I) - X(Z - 1)} \tag{6}$$

Se hace evidente, que es un método recursivo y que en el momento en el que en $I = N$ en $P(I)$ obtendremos el valor $F(A)$

2.1.1. Implementación en computador

```
function salida = aitken(t,x,y)
  n=length(y);
  Z=zeros(n);
```

```
  Z(1,:)=y;
  aux=zeros(1,length(t));
  for k=1:1:length(t)
    for j=1:n-1
      for i=j+1:n
        Z(j+1,i)=(Z(j,i)*(t(k)-x(j))-Z(j,j)
                 *(t(k)-x(i)))/(x(i)-x(j));
      endfor
    endfor
    aux(k)=Z(n,n);
  endfor
  Z'
  salida=aux
endfunction
```

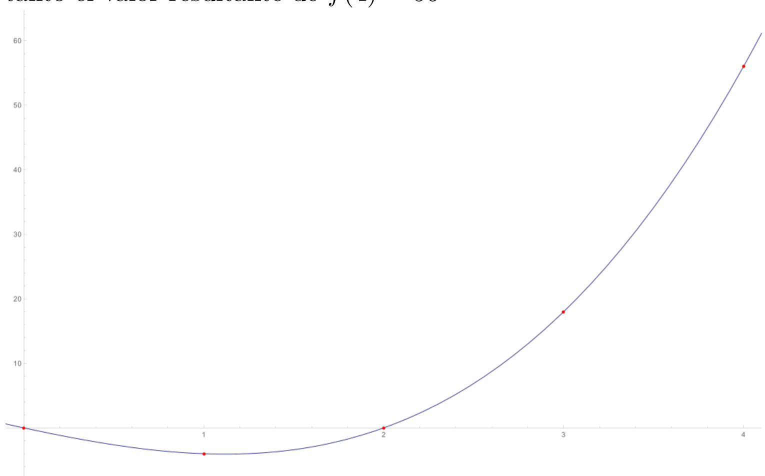
2.1.2. Ejemplo

Enunciado: Calcular con el método de Aitken, el valor en el punto $A = 4$ sabiendo que $f(x) = \{0, -4, 0, 18\}$ en los puntos $x = \{0, 1, 2, 3\}$ respectivamente

Solución:

x_i	$x - (x_i)$	$f(x_i)$			
0	4	0			
1	3	-4	-16		
2	2	0	0	32	
3	1	18	24	44	56

Por lo tanto el valor resultante de $f(4) = 56$



2.2. Algoritmo de Neville

Teniendo exactamente los mismos datos que en 2.1 y siguiendo los mismos procedimientos a excepción de que en (6) no tomamos la misma fórmula de P(I), en su lugar:

$$P(I) = \frac{(X(I) - A) * P(I-1) - (X(I-Z) - A) * P(I)}{X(I) - X(I-Z)} \tag{7}$$

e igualmente $P(N)$ será el valor de $F(A)$

2.2.1. Implementación en computador

```
function salida = neville(t,x,y)
  n=length(x);
  z=zeros(n,n+1);
  z(:,1);
  z(:,1)=x';
  z(:,2)=y';
  a=0;
  for i=3:1:n+1
    for j=i-1:1:n
      z(j,i)=z(j,i-1)+((z(j,1)-t)/(z(j,1)-z(j-1-a,1)))
                *(z(j-1,i-1)-z(j,i-1));
    endfor
    a=a+1;
  endfor
  salida = z(n, n+1)
endfunction
```

2.2.2. Ejemplo

Enunciado: Calcular con el método de Neville, el valor en el punto $A = 5$ sabiendo que $f(x) = \{-2, -1, 0, -5\}$ en los puntos $x = \{-2, -1, 0, 1\}$ respectivamente

Solución:

x_i	$x - (x_i)$	$f(x_i)$			
-2	7	-2			
-1	6	-1	5		
0	5	0	5	5	
1	4	-5	-25	-85	-205

Por lo tanto el valor resultante de $f(5) = -205$

Referencias

[1] Universidad de Sevilla Departamento de ecuaciones diferenciales y análisis numérico. Introducción a la interpolación y a la integración numérica. http://departamento.us.es/edan/php/asig/LICMAT/LMCN1/Tema3CN10809.pdf, 2014. [Pag 2/52; Online; 29 Mayo 2015].

[2] Mariano Gasca González. *Cálculo Numérico I*. UNED, Madrid, 1989.

[3] Universidad Granada. Logo de la Universidad de Granada.

[4] Universidad de las Palmas de Gran Canaria Jesús García Quesada. Interpolación: Las fórmulas de Newton en diferencias finitas. http://numat.net/tutor/newtondf.pdf, 2000. [Online; 29 Mayo 2015].

[5] Universidad Granada José Martínez Aroza. Interpolación. http://www.ugr.es/~mpasadas/ftp/Inter1.pdf, 2006-2007. [Online; 29 Mayo 2015].

[6] Wolfram Mathematica. Gráficas.

[7] Universidad Politécnica de Madrid Prof.Alfredo López Benito Prof.Carlos Conde Lázaro Prof.Arturo Hidalgo López. INTERPOLACIÓN: Fórmulas en diferencias finitas. http://ocw.upm.es/matematica-aplicada/programacion-y-metodos-numericos/contenidos/TEMA_3/Presentaciones/I4_Interpolacion_Dif_Fin_ocw.pdf, 2007. [Online; 29 Mayo 2015].

[8] Universidad de Bilbao Purificación González. Interpolación. http://www.ehu.eus/pegonzalez/I.Teleco/Apuntes/tema5.pdf, 2014. [Pag 2/54, Online; 29 Mayo 2015].

[9] Instituto universitario politécnico Santiago Mariño Roxana Velásquez. Interpolación. http://es.scribd.com/doc/166976041/interpolacion-doc#scribd. [Online; 29 Mayo 2015].

[10] Javier Segura. Interpolación. http://personales.unican.es/segurajj/interp_p.pdf, 2014. [Online; 29 Mayo 2015].