CAPÍTULO 2

INTERPOLACIÓN Y RAÍCES DE FUNCIONES

Bruno Julia Diaz, última revisión Sep 2016

2.1 Interpolación de funciones

Al realizar cálculo numérico es muy habitual encontrarse con que las funciones con las que se va a trabajar no son conocidas de manera analítica sino que se conocen en un conjunto finito (y numerable) de abscisas, $\{x_1, f_1\}, \{x_2, f_x\}, \ldots, \{x_N, f_N\}$. Esto ocurre siempre en cualquier experimento en física, por ejemplo si medimos volúmenes como función de la temperatura a presión constante iremos acumulando una serie de valores, $\{V_1, P_1\}, \{V_2, P_2\}, \ldots, \{V_N, P_N\}$ y en ningún caso tendremos una función explícita $P(V)$. Esto último si que lo tenemos si desarrollamos una teoría que prediga dicho comportamiento. Lo mismo ocurre con cualquier otra magnitud que midamos, siempre tendremos un conjunto de parejas de números. Siendo rigurosos de hecho cualquier cálculo que hagamos con el ordenador utilizará siempre funciones conocidas en un número finito de puntos. En estos casos suele construirse una función que interpole los puntos que conocemos, esto es, que basándose en la información que conocemos de una serie de valores a la función en puntos intermedios. Esta función $f_{\text{inter}}(x)$ estará construida de forma que en los puntos conocidos, x_k tome exactamente los valores f_k, conocidos. En los puntos intermedios producirá una aproximación continua y derivable, y normalmente suave, de los datos conocidos.

La interpolación más sencilla de construir es la interpolación polinómica, esto es, buscar el polinomio que pase por todos los puntos que conocemos. Un método sencillo de escribir este polinomio es utilizar la forma ideada por Lagrange [1]. Se define el polinomio de grado N que se

[1] Seguimos la notación de Abramowitz y Stegun.

anula en todos los puntos de la malla,

$$\pi_N(x) = \prod_{k=0}^{N}(x - x_k)\,, \tag{2.1}$$

cuya derivada evaluada en un punto de la malla es sencillamente,

$$\pi'_N(x_i) = (x_i - x_0)(x_i - x_1)\dots(x_i - x_{i-1})(x_i - x_{i+1})\dots(x_i - x_N)\,. \tag{2.2}$$

Con ellos se definen los polinomios,

$$\ell_i(x) = \frac{\pi_N(x)}{(x - x_i)\pi'_N(x_i)} = \prod_{k\neq i=1}^{N}\frac{(x - x_k)}{x_i - x_k} \tag{2.3}$$

que cumplen, $\ell_i(x_j) = \delta_{i,j}$. Con lo que estos polinomios se anulan en todos los puntos de la malla excepto en el x_i donde valen 1. Ahora es fácil ver que el polinomio interpolador será,

$$P_N(x) = \sum_{i=0}^{N} f_i\ell_i(x) = \sum_{i=0}^{N} f_i\ \left(\prod_{k\neq i=0}^{N}\frac{(x - x_k)}{x_i - x_k}\right)\,. \tag{2.4}$$

Es sencillo comprobar que $P(x_k) = f_k$ como queríamos.

Se puede demostrar [2] que

$$f(x) = P_N(x) + R_N(x) \quad \text{con } R_N(x) = \frac{f^{(N+1)}(\xi)\pi_N(x)}{(n+1)!} \quad \xi \in (x_0, x_N)\,. \tag{2.5}$$

Es importante notar que este resto es en general inútil desde el punto de vista práctico porque en general no dispondremos de la derivada $N+1$ de la función. Como ocurría con la fórmula de Taylor con resto en muchas ocasiones puede usarse para estimar la calidad de la interpolación.

Un par de puntos importantes que pretende ilustrar el Ejemplo 2.2 son,

- Habitualmente es mejor utilizar polinomios interpoladores de grados moderados, 3, o 4 puntos, en lugar de utilizar el polinomio de grado igual al número total de puntos de la malla. La razón es que este polinomio con demasiados puntos tiene una tendencia a producir demasiada estructura.
- Normalmente basta con interpolar utilizando 2 o 3 puntos, recordando que al menos 2 son necesarios para que haya curvatura y 3 para que describamos inflexiones.

Ejemplo 2.1: Si tenemos 3 puntos en la malla, tendremos un polinomio de segundo grado, de la forma,

$$P_2(x) = \frac{(x - x_1)(x - x_2)}{(x_0 - x_1)(x_0 - x_2)}f_0 + \frac{(x - x_0)(x - x_2)}{(x_1 - x_0)(x_1 - x_2)}f_1 + \frac{(x - x_0)(x - x_1)}{(x_2 - x_0)(x_2 - x_1)}f_2\,. \tag{2.6}$$

En muchas aplicaciones la malla está equiespaciada, de forma que $x_k = x_0 + (k-1)h$, con h la distancia entre dos puntos consecutivos de la malla. En este caso las fórmulas se simplifican notablemente.

Ejemplo 2.2: Construye una tabla de 20 puntos equiespaciados con valores de x, x_k, entre 0 y π y el valor correspondiente del $\sin(x_k)$.

Compara la representación gráfica exacta del $\sin(x)$ con:

[2] Abramowitz y Stegun, p. 878.

a) la interpolación usando todos los puntos,

b) interpolación lineal en cada sub intervalo, y

c) interpolación cuadrática en cada subintervalo.

¿Cómo organizaremos el programa? Básicamente el proceso será el siguiente. Necesitamos un número de valores, **nval**, unos extremos del intérvalo, $x_1, x_{\rm nval}$ y unos valores para la función en todos esos puntos, $f_1, f_2, \ldots, f_{\rm nval}$. Todos estos serán datos de entrada en el código y son necesarios.

La salida del código serán los tres conjuntos de pares de números que nos están pidiendo para poder hacer una figura con ellos.

2.2 Raíces de funciones

Comenzaremos por un problema que es relativamente sencillo de exponer que es el de resolver una ecuación no lineal o, lo que es lo mismo, encontrar los ceros de una función dada. El problema será pues encontrar los números reales x, tales que,

$$f(x) = 0\,. \tag{2.7}$$

Supondremos que somos capaces de evaluar la función en cualquier punto que se nos ocurra x_k en un tiempo razonable. ¿Seremos capaces de encontrar todos los puntos que satisfacen la Eq. (2.7)? Primero, observemos que el problema está de partida mal planteado ya que no sabemos cuantas soluciones hay, excepto si la función $f(x)$ es un polinomio de grado n en cuyo caso sabemos por el teorema fundamental del álgebra que hay n soluciones (complejas).

Así pues nos preocuparemos de encontrar alguna solución y ya iremos viendo según el problema en cuestión.

2.2.1 Método de la bisección

El método más primario aunque robusto y potente es el método de la bisección. Recordemos que si la función con la que trabajamos es continua tenemos un teorema fascinante que nos asegura que si la función cambia de signo entre dos puntos entonces seguro que hay al menos un punto intermedio donde se anula. Aprovecharemos este teorema y construiremos un método de intervalos encajados del siguiente modo.

Primero tenemos $x = a$ y $x = b$, $a < b$ tales que $f(a)f(b) < 0$ lo que nos asegura que en el intervalo (a, b) existe algún ξ tal que $f(\xi) = 0$. Procedamos del siguiente modo, como de costumbre buscamos la solución con una cierta precisión ε.

1) Definimos
$$c = \frac{a+b}{2} \tag{2.8}$$
y evaluamos $f(c)$.

2) Si $f(c)f(a) < 0$ entonces redefinimos nuestro intervalo como $b = c$ y $a = a$ y volvemos a 1).

3) Si $f(c)f(b) < 0$ entonces redefinimos nuestro intervalo como $b = b$ y $a = c$ y volvemos a 1).

4) Si $f(c) = 0$, tenemos la solución y acabamos.

5) Si $(b - a) < \varepsilon$ tenemos que el error que cometemos al decir que la solución es c es menor que la precisión que queremos, con lo que hemos acabado.

Observemos una serie de puntos importantes.

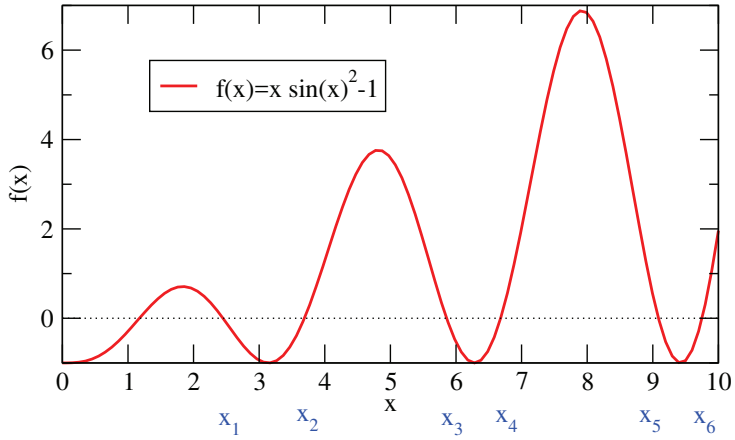

Figura 2.1: Representación gráfica de la función $f(x) = xsin^2(x) - 1$ en el intervalo $[0, 10]$. Los ceros de la función estar marcados como x_k, $k = 1, 6$.

- El método siempre converge
- El error disminuye un factor 2 en cada interacción, de hecho el nuevo error es proporcional al anterior.
- El método encuentra "una" solución, aunque obviamente podría haber más de una.
- Tal y como lo hemos descrito el método se puede programar de manera muy sencilla "machacando" variables.

Un ejemplo de programación sería,

```
C PROGRAM BISECCION
C
C THE PROGRAM LOOKS FOR A SOLUTION TO F(X)=0.
C THE FUNCTION IS DEFINED EXTERNAL TO THE ROUTINE
C THE TWO INITIAL VALUES, A, B SHOULD FULFILL F(A) F(B)<0
C THE PROGRAM STOPS EITHER IF THE EXACT POINT IS FOUND OR
C IF THE DESIRED ACCURACY HAS BEEN ACHIEVED

C BJD SEP 2015
C LAST REVISED 1 OCT 2015
C
       IMPLICIT NONE
       REAL A,B,C,F
       REAL FA,FB,FC
       REAL EPS,DIFF
       INTEGER I,MAXITER
C       EXTERNAL F

C PRECISION REQUERIDA
       EPS=0.0001
```

```
C VALORES EXTREMOS INICIALES
       A=3.
       B=8.
       WRITE(*,500) A,B
500    FORMAT('A=',F9.3,2X,'B=',F9.3)

C CALCULA MAXITER

MAXITER=NINT(LOG((B-A)/EPS)/LOG(2.))+1
        WRITE(*,*) "MAXITER=",MAXITER
C COMIENZA EL METODO
       DO I=1,MAXITER
          C=(A+B)/2.
          FA=F(A)
          FB=F(B)
          FC=F(C)
          IF (FA*FB.GE.0.) THEN
            PRINT*,"LA FUNCION NO CAMBIA DE SIGNO EN A,B",A,B,FA,FB
            STOP
          ENDIF
          IF (FC.eq.0.) THEN
           PRINT*,"SOLUCION EXACTA X=",C
           STOP
          ENDIF
          IF (F(A)*F(C).LT.0.) THEN
             B=C
           ELSE
             A=C
          ENDIF
          DIFF=(B-A)

      IF (DIFF.LE.EPS) THEN
         PRINT*,"SOLUCION APROXIMADA X=",C
         PRINT*,"ERROR <",DIFF
         STOP
      ENDIF

       PRINT*,"ITERACION NUMERO:",I,"C=",C, " error=",diff
ENDDO
       END


       FUNCTION F(X)
       IMPLICIT NONE
       REAL X,F
       F=SIN(X)**2*X-1.
       END
```

cuya salida es,

```
A=    3.000  B=    8.000
 MAXITER=          17
 ITERACION NUMERO:           1 C=   5.50000000      error=   2.50000000
 ITERACION NUMERO:           2 C=   4.25000000      error=   1.25000000
 ITERACION NUMERO:           3 C=   3.62500000      error=  0.625000000
 ITERACION NUMERO:           4 C=   3.93750000      error=  0.312500000
 ITERACION NUMERO:           5 C=   3.78125000      error=  0.156250000
 ITERACION NUMERO:           6 C=   3.70312500      error=   7.81250000E-02
 ITERACION NUMERO:           7 C=   3.66406250      error=   3.90625000E-02
 ITERACION NUMERO:           8 C=   3.68359375      error=   1.95312500E-02
 ITERACION NUMERO:           9 C=   3.69335938      error=   9.76562500E-03
 ITERACION NUMERO:          10 C=   3.68847656      error=   4.88281250E-03
 ITERACION NUMERO:          11 C=   3.69091797      error=   2.44140625E-03
 ITERACION NUMERO:          12 C=   3.68969727      error=   1.22070312E-03
 ITERACION NUMERO:          13 C=   3.68908691      error=   6.10351562E-04
 ITERACION NUMERO:          14 C=   3.68939209      error=   3.05175781E-04
 ITERACION NUMERO:          15 C=   3.68923950      error=   1.52587891E-04
 SOLUCION APROXIMADA X=   3.68916321
 ERROR <   7.62939453E-05
```

En este caso hemos encontrado una solución de las muchas que tiene este problema, como podemos ver en la figura 2.1, en particular en este caso hemos encontrado el valor de x_2.

2.2.2 Método de Regula-Falsi

El método de Regula Falsi es muy parecido a la bisección, en el sentido de que requiere la misma información para funcionar. Por ejemplo si buscamos un cero de la función entre $A \equiv A_0$ y $B \equiv B_0$ necesitaremos saber inicialmente que $f(A)f(B) < 0$, como en el caso de la bisección. La diferencia entre ambos está en que el punto intermedio que elegíamos en la bisección ahora es reemplazado por un punto diferente. El nuevo punto C dentro del intervalo $[A, B]$ se escoge buscando el corte de la recta que pasa por $(A, f(A))$ y $(B, f(B))$ y el eje de abscisas. Este punto estará dentro de $[A, B]$ siempre que se cumpla $f(A)f(B) < 0$. Como hacíamos en la bisección, ahora nos quedaremos con el subintervalo en el que la función siga cambiando de signo. En la figura 2.2 mostramos como funciona el método.

La propia descripción anterior del método proporciona el algoritmo que tendríamos que implementar. Lo que cabe destacar es que mientras en el método de la bisección el tamaño del intervalo disminuye un factor dos en cada iteración en este método el tamaño del intervalo normalmente satura hasta un valor fijo, ya que una vez el punto intermedio está suficientemente cerca de la raíz que buscamos el método solo se acerca a la raíz por un lado.

Así en este método el criterio de convergencia que debemos establecer tiene que estar basado en la diferencia entre los valores en dos iteraciones sucesivas.

Resumamos:

(1) Arrancamos con dos valores A y B tales que $f(A)f(B) < 0$ y una precisión ε deseada.

(2) Elegimos el punto

$$C = \frac{Af(B) - Bf(A)}{f(B) - f(A)} \tag{2.9}$$

(3) Es $f(A)f(C) < 0$? En caso afirmativo reemplacemos $B = C$.

(4) En caso contrario:

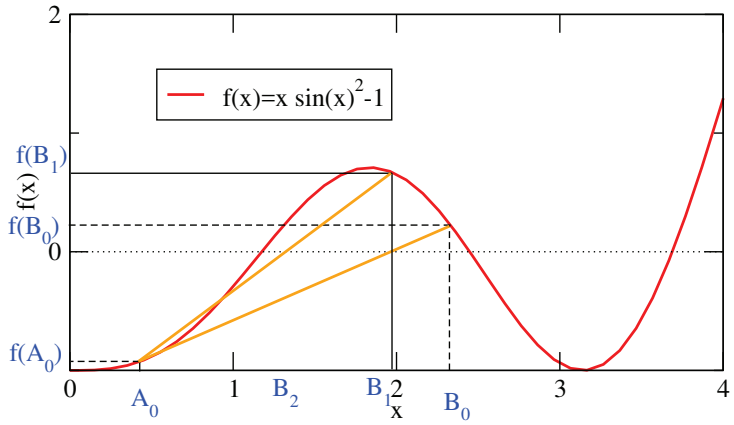

Figura 2.2: Representación gráfica de la función $f(x) = xsin^2(x)-1$ en el intervalo $[0, 4]$. Ilustración el método Regula-Falsi (3 iteraciones).

(4a) Es $f(c) = 0$? si si, escribe C y acaba.

(4b) Es $f(B)f(C) < 0$? En caso afirmativo reemplacemos $A = C$.

(5) Criterio de convergencia. $\Delta = \min(C - A, B - C)$. Es $\Delta < \varepsilon$? En caso afirmativo, escribimos el punto C y acabamos. En caso negativo volvamos a (1).

El método se puede pensar como que utiliza en cada iteración la interpolación de Lagrange de grado 1 usando los puntos del extremo del intervalo. Con esta aproximación se encuentra rápidamente el corte con el eje de ordenadas, el punto C y se vuelve a empezar.

Ejemplo 2.3: Utiliza el método de regula falsi para encontrar todos los ceros del polinomio cúbico $P(x) = x^3/5 + x^2 - 2x - 1$ con $x \in [-5, 1]$. Compara el número de iteraciones que necesitar realizar con este método con las que necesitarías con el método de bisección.

2.2.3 Método de Newton-Raphson

El método de Newton-Raphson es un método que se puede demostrar que es muy rápido, pero requiere saber la derivada de la función. El método funciona del siguiente modo.

(1) Necesitamos comenzar con un valor x_0 que este relativamente cerca de la raíz que buscamos.

(2) Construimos el polinomio de Taylor de grado uno alrededor de dicho punto (la recta de pendiente igual a la de la función que pasa por el punto $(x_0, f(x_0))$),

$$T_1(x; x_0) = f(x_0) + (x - x_0)f'(x_0) \tag{2.10}$$

Resolvemos el problema lineal, $T_1(x_1; x_0) = 0$, cuya solución es,

$$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)} \tag{2.11}$$

(3) Criterio de convergencia. Si $|\Delta_1| = |x_1 - x_0| < \epsilon$ finalizamos, en caso contrario, asignamos $x_0 = x_1$ y volvemos a (1).

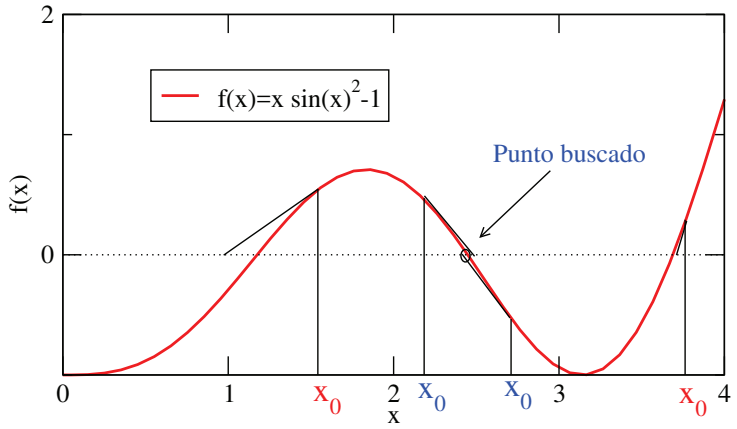

Figura 2.3: Representación gráfica de la función $f(x) = xsin^2(x)-1$ en el intervalo $[0,4]$. Ilustración de los puntos iniciales, x_0, razonables (color azul) y menos razonables (color rojo) si buscamos la raíz marcada en la figura.

Este método es muy rápido si todo va bien. De hecho se puede acotar como disminuye el error cometido en la iteración $i+1$ si sabemos el valor de la derivada segunda. Veamos, podemos escribir el desarrollo de Taylor de $f(x)$ alrededor de x_i y evaluarla en la raíz (desconocida), x^*.

$$\begin{aligned} f(x^*) &= f(x_i) + f'(x_i)(x^* - x_i) + \frac{f''(x_i)}{2}(x^* - x_i)^2 + \dots \\ &\Rightarrow \\ f(x_i) &\simeq f'(x_i)\Delta_i - \frac{f''(x_i)}{2}\Delta_i^2 \end{aligned} \tag{2.12}$$

Veamos ahora el error en el paso $i+1$, tenemos,

$$\begin{aligned} \Delta_{i+1} &= x_{i+1} - x^* = x_i - \frac{f(x_i)}{f'(x_i)} - x^* = \Delta_i - \frac{f(x_i)}{f'(x_i)} \simeq \Delta_i + \frac{-f'(x_i)\Delta_i + \frac{f''(x_i)}{2}\Delta_i^2}{f'(x_i)} \\ &\simeq \frac{f''(x_i)}{2f'(x_i)}\Delta_i^2 \end{aligned} \tag{2.13}$$

Obteniendo una mejora cuadrática del error.

El problema más habitual es que el punto inicial tenga una derivada cercana a cero. Esto hace que el siguiente punto este aun más lejos del cero y podría estar fuera del intervalo en que conocemos la función. Igualmente, si la derivada en el punto tiene la pendiente opuesta a la que interesa, el método convergerá bien a otra raíz o divergirá, ver Fig. 2.3.

Ejemplo 2.4: Compara el método de Newton-Raphson con el de bisección para el caso del Ejemplo 2.3. Por ejemplo, para obtener una precisión de 0.000001 partiendo de $x_0 = 1$, el Newton Raphson realiza 7 iteraciones, mientras que el de la bisección requiere 21 iteraciones si usamos el intervalo $[1,5]$.

```
Newton Raphson
Iteración Valor X1
1         3.99999976
2         2.73076916
3         2.09210801
```

```
4       1.87921441
5       1.85334921
6       1.85297227
7       1.85297227
```

Ejemplo 2.5: El método de Newton Raphson se puede utilizar para calcular raíces cuadradas. Considera la ecuación,

$$x^2 - b = 0 \tag{2.14}$$

que sólo tiene una solución positiva, $\sqrt{b}$. Si aplicamos el método de Newton Raphson obtenemos, $f(x) = x^2 - b$, $f'(x) = 2x$,

$$x_{n+1} = x_n - \frac{x_n^2 - b}{2x_n} = \frac{1}{2}\left(x_n + \frac{b}{x_n}\right) . \tag{2.15}$$

Que de hecho es una formula recursiva, que converge muy rápido, para calcular la raíz cuadrada de b utilizada por los Babilonios.

Ejemplo 2.6: Transforma los códigos que hayas realizado para los métodos de bisección, regula falsi y Newton-Raphson en subroutines. De forma, que puedas llamarlas para cualquier función predefinida. Añade a la lista de argumentos de las subrutinas una etiqueta de error (un entero), p. ej. **ierr**. Esta etiqueta será 0 si todo ha ido bien y el método ha convergido con la precisión deseada, 1 si no ha convergido.

2.2.4 Método de la secante

Hay una variación del método de Newton Raphson para casos en los que, bien no se conozca la derivada de la función, o esta se anule en el punto. El método reemplaza la derivada en el punto inicial por la pendiente de la secante usando dos puntos cercanos, viene definido por la siguiente recurrencia,

$$x_{n+1} = x_n - f(x_n)\frac{x_n - x_{n-1}}{f(x_n) - f(x_{n-1})} . \tag{2.16}$$

Como se puede observar el método necesita dos valores iniciales, x_0 y x_1 que idealmente son próximos y no muy alejados de la raíz. El método converge algo más lentamente que el de Newton Raphson.

2.2.5 Métodos híbridos

Normalmente se utilizan métodos híbridos en los que se combinan métodos de convergencia cuadrática como el Newton Raphson o el de la secante con el método de la bisección.