Interpolación

Lección 14

Dr. Pablo Alvarado Moya

CE3102 Análisis Numérico para Ingeniería
Área de Ingeniería en Computadores
Tecnológico de Costa Rica

I Semestre 2018

Contenido

Introducción

- Dados un conjunto de n puntos $x_0 < x_1 < x_2 < \ldots < x_{n-1}$ y los valores de una función $f(x)$ sobre esos puntos.
  - Esto puede surgir por ejemplo de procesos de medición físicos o computacionales

Introducción

- Dados un conjunto de n puntos $x_0 < x_1 < x_2 < \ldots < x_{n-1}$ y los valores de una función $f(x)$ sobre esos puntos.
    - Esto puede surgir por ejemplo de procesos de medición físicos o computacionales
- La tarea ahora es calcular $f(x)$ para cualquier x a partir de los n valores de medición.

Introducción

- Dados un conjunto de n puntos $x_0 < x_1 < x_2 < \ldots < x_{n-1}$ y los valores de una función $f(x)$ sobre esos puntos.
  - Esto puede surgir por ejemplo de procesos de medición físicos o computacionales
- La tarea ahora es calcular $f(x)$ para cualquier x a partir de los n valores de medición.
- **Interpolación** si $x \in [x_0; x_{n-1}]$
- **Extrapolación** si $x < x_0$ ó $x > x_{n-1}$

Introducción

- Dados un conjunto de n puntos $x_0 < x_1 < x_2 < \ldots < x_{n-1}$ y los valores de una función $f(x)$ sobre esos puntos.
  - Esto puede surgir por ejemplo de procesos de medición físicos o computacionales
- La tarea ahora es calcular $f(x)$ para cualquier x a partir de los n valores de medición.
- **Interpolación** si $x \in [x_0; x_{n-1}]$
- **Extrapolación** si $x < x_0$ ó $x > x_{n-1}$
- Métodos requieren un modelo general para la función entre o fuera de los puntos conocidos:
  1. Polinomios
  2. Funciones racionales (cociente de polinomios)
  3. Funciones trigonométricas (análisis de Fourier)
  4. Funciones gaussianas (funciones de base radial)

Interpolación polinomial

- La interpolación polinomial de n-ésimo orden utiliza al polinomio

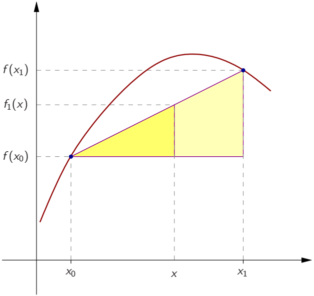
$$f(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_n x^n$$

para aproximar la función dados $n+1$ puntos.
- Solo existe un único polinomio de n-ésimo orden que pasa por $n+1$ puntos.
- Existen varios métodos para expresar y encontrar dicho polinomio:
  - Interpolación polinomial **de Newton**
  - Interpolación polinomial **de Lagrange**

Interpolación lineal

- Forma más simple de interpolación
- Se unen dos puntos adyacentes en los datos con una línea recta
- Se observa que

$$\frac{f_1(x) - f(x_0)}{x - x_0} = \frac{f(x_1) - f(x_0)}{x_1 - x_0}$$



- La interpolación lineal es

$$f_1(x) = f(x_0) + \frac{f(x_1) - f(x_0)}{x_1 - x_0}(x - x_0)$$

- Subíndice en $f_1(x)$ denota primer orden de interpolación

- Note aprox. en diferenc. divididas finitas

$$\frac{f(x_1) - f(x_0)}{x_1 - x_0} \approx \frac{df(x)}{dx}$$

Interpolación cuadrática

- Interpolación cuadrática utiliza 3 puntos:

$$f_2(x) = b_0 + b_1(x - x_0) + b_2(x - x_0)(x - x_1) \qquad (*)$$

- La expresión anterior es equivalente a:

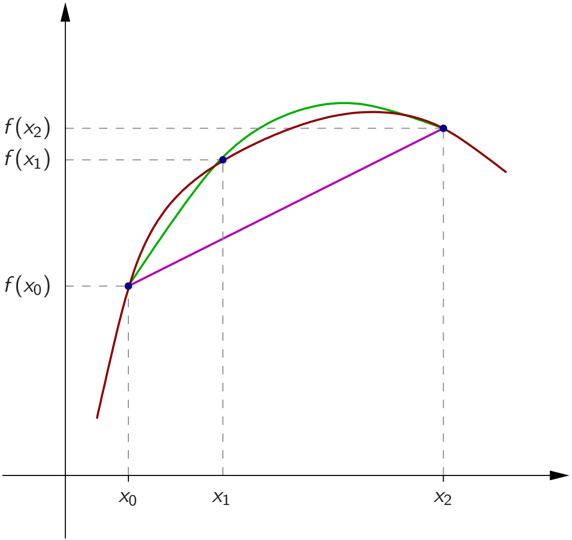
$$\begin{aligned} f_2(x) &= b_0 + b_1 x - b_1 x_0 + b_2 x^2 - b_2 x(x_0 + x_1) + b_2 x_0 x_1 \\ &= b_2 x^2 + [b_1 + b_2(x_0 + x_1)]\, x + (b_0 - b_1 x_0 + b_2 x_0 x_1) \\ &= a_2 x^2 + a_1 x + a_0 \end{aligned}$$

con

$$\begin{aligned} a_2 &= b_2 \\ a_1 &= b_1 - b_2 x_0 - b_2 x_1 \\ a_0 &= b_0 - b_2 x_0 + b_2 x_0 x_1 \end{aligned}$$

- Ventaja de forma $(*)$: error no varía incrementando x

Ejemplo gráfico de interpolaciones lineal y cuadrática



Cálculo de coeficientes en forma de Newton (1)

Para

$$f_2(x) = b_0 + b_1(x - x_0) + b_2(x - x_0)(x - x_1)$$

evaluando en x_0 se obtiene:

$$b_0 = f(x_0)$$

Evaluando en x_1 y utilizando a b_0 se obtiene

$$b_1 = \frac{f(x_1) - f(x_0)}{x_1 - x_0}$$

que es una aproximación en diferencias divididas finitas de $f'(x_0)$.

Cálculo de coeficientes en forma de Newton (2)

Para b_2 se usan los dos resultados anteriores y se evalúa en x_2 para obtener:

$$b_2 = \frac{\dfrac{f(x_2) - f(x_1)}{x_2 - x_1} - \dfrac{f(x_1) - f(x_0)}{x_1 - x_0}}{x_2 - x_0}$$

que es la aproximación en diferencias divididas finitas de $f''(x_0)$

Interpolación polinomial de Newton (1)

Diferencias finitas divididas

Lo anterior se generaliza para polinomios de n-ésimo grado con $n+1$ puntos:

$$f_n(x) = b_0 + b_1(x - x_0) + \cdots + b_n(x - x_0)(x - x_1)\cdots(x - x_{n-1})$$

y con los datos disponibles $[x_0, f(x_0)], [x_1, f(x_1)], \cdots, [x_n, f(x_n)]$ se calcula

$$\begin{aligned}
b_0 &= f(x_0) \\
b_1 &= f[x_1, x_0] \\
b_2 &= f[x_2, x_1, x_0] \\
&\vdots \\
b_n &= f[x_n, x_{n-1}, \cdots, x_1, x_0]
\end{aligned}$$

Interpolación polinomial de Newton (2)

Diferencias finitas divididas

Se usan paréntesis cuadrados para denotar diferencias divididas finitas:

$$
\begin{aligned}
f[x_i, x_j] &= \frac{f(x_i) - f(x_j)}{x_i - x_j} \\
f[x_i, x_j, x_k] &= \frac{f[x_i, x_j] - f[x_j, x_k]}{x_i - x_k} \\
&\vdots \\
f[x_n, x_{n-1}, \ldots, x_1, x_0] &= \frac{f[x_n, x_{n-1}, \ldots, x_1] - f[x_{n-1}, x_{n-2}, \ldots, x_0]}{x_n - x_0}
\end{aligned}
$$

Interpolación polinomial de Newton (3)

Diferencias finitas divididas

El polinomio de interpolación de Newton en diferencias divididas es

$$\begin{aligned} f_n(x) = & f(x_0) + (x - x_0)f[x_1, x_0] + (x - x_0)(x - x_1)f[x_2, x_1, x_0] \\ & + \cdots + (x - x_0)(x - x_1)\cdots(x - x_{n-1})f[x_n, x_{n-1}, \ldots, x_1, x_0] \end{aligned}$$

El cálculo de las diferencias divididas finitas es recursivo

i	x_i	$f(x_i)$	1era	2da	3ra
0	x_0	$f(x_0)$	$f[x_1, x_0]$	$f[x_2, x_1, x_0]$	$f[x_3, x_2, x_1, x_0]$
1	x_1	$f(x_1)$	$f[x_2, x_1]$	$f[x_3, x_2, x_1]$	
2	x_2	$f(x_2)$	$f[x_3, x_2]$		
3	x_3	$f(x_3)$			

Ejemplo de interpolación con Newton

```
#!/usr/bin/gnuplot

x0=1.0;  f0=0;
x1=2.5;  f1=1;
x2=3;    f2=3;
x3=4;    f3=1;

df0=(f1-f0)/(x1-x0);
df1=(f2-f1)/(x2-x1);
df2=(f3-f2)/(x3-x2);

ddf0=(df1-df0)/(x2-x0);
ddf1=(df2-df1)/(x3-x1);

dddf0=(ddf1-ddf0)/(x3-x0);

set xrange [x0-0.5:x3+0.5]

f(x)=f0 + (x-x0)*(df0 + (x-x1)*(ddf0 + (x-x2)*dddf0));

plot f(x) with lines notitle, \
     "data.dat" with impulses notitle, \
     "data.dat" with points notitle
```

Errores de la interpolación polinomial (1)

- La fórmula de interpolación

$$\begin{aligned} f_n(x) =& f(x_0) + (x - x_0) f[x_1, x_0] + (x - x_0)(x - x_1) f[x_2, x_1, x_0] \\ & + \cdots + (x - x_0)(x - x_1) \cdots (x - x_{n-1}) f[x_n, x_{n-1}, \ldots, x_1, x_0] \end{aligned}$$

  es similar a una serie de Taylor, donde cada término representa una nueva derivada.

- Recuérdese que el resíduo de la serie de Taylor es:

$$R_n = \frac{f^{(n+1)}(\xi)}{(n+1)!} (x_{i+1} - x_i)^{n+1}$$

  con $\xi \in [x_i, x_{i+1}]$.

Errores de la interpolación polinomial (2)

- Para polinomio de interpolación de n-ésimo grado, la expresión análoga del error es:

$$R_n = \frac{f^{(n+1)}(\xi)}{(n+1)!}(x - x_0)(x - x_1)\cdots(x - x_n)$$

- Usualmente la función y sus derivadas son desconocidas, pero se puede usar la diferencia dividida para aproximar la $(n+1)$-ésima derivada

$$R_n = f[x, x_n, x_{n-1}, \ldots, x_0](x - x_0)(x - x_1)\cdots(x - x_n)$$

que se puede calcular solo si se conoce otro punto adicional

$$R_n \approx f[x_{n+1}, x_n, x_{n-1}, \ldots, x_0](x - x_0)(x - x_1)\cdots(x - x_n)$$

Propiedades del algoritmo

Tres propiedades de los polinomios de interpolación de Newton los hacen atractivos para su implementación computacional:

1. Es posible desarrollar de manera secuencial versiones de grado superior con la adición de un solo término a la ecuación de grado inferior en

   $$f_n(x) = b_0 + b_1(x - x_0) + \cdots + b_n(x - x_0)(x - x_1)\cdots(x - x_{n-1})$$

   lo que es útil para estimar *en-línea* el grado del polinomio.
2. Las diferencias divididas finitas se pueden calcular eficientemente
3. El error estimado se puede incorporar al algoritmo.

Algoritmo de interpolación de Newton

Proponga un algoritmo

Algoritmo de interpolación de Newton

```
template<typename T>
T newtonInterpolation(const vector<T>& x,const vector<T>& y, const int n,
                      const T xi) {

  vector<T> yint(n+1,T()); // contiene las aproximaciones según orden
  vector<T> ea(n+1,T());   // contiene error aprox. según orden
  matrix<T> fdd(n+1,n+1,T()); // diferencias finitas divididas

  for (int i=0;i<=n;++i) {
    fdd[i][0]=y[i];
  }
  for (int j=1;j<=n;++j) {
    for (int i=0;i<=n-j;++i) {
      fdd[i][j] = (fdd[i+1][j-1]-fdd[i][j-1])/(x[i+j]-x[i]);
    }
  }
  T xterm=T(1); // acumulación geométrica de términos (x-x0)(x-x1)...
  yint[0] = fdd[0][0];
  for (int order=1;i<=n;++i) {
    xterm *= (xi-x[order-1]);
    T yint2 = yint[order-1] + fdd[0][order]*xterm;
    ea[order-1] = yint2-yint[order-1];
    yint[order] = yint2;
  }
  return yint[n];
}
```

Polinomios de interpolación de Lagrange (1)

Reformulación del polinomio de Newton que evita cálculo de las diferencias divididas:

$$f_n(x) = \sum_{i=0}^{n} L_i(x) f(x_i)$$

donde

$$L_i(x) = \prod_{\substack{j=0 \\ j\neq i}}^{n} \frac{x - x_j}{x_i - x_j}$$

- Obsérvese que cada término $L_i(x)$ es 1 en $x = x_i$ y 0 en todos los otros puntos.

Polinomios de interpolación de Lagrange (2)

- Así, cada producto $L_i(x)f(x_i)$ toma el valor de $f(x_i)$ en el punto x_i.
- La suma de todos los términos es el único polinomio de n-ésimo orden que pasa a través de los $n+1$ puntos asociados a los datos.
- El error se obtiene con

$$R_n = f[x, x_n, x_{n-1}, \ldots, x_0] \prod_{i=0}^{n} (x - x_i)$$

que se puede calcular contando con un punto adicional $x = x_{n+1}$, pero puesto que no se cuentan aquí con las diferencias divididas finitas, no es tan directo su cálculo.

Algoritmo de interpolación de Lagrange

```
template<typename T>
T lagrangeInterpolation(const vector<T>& x,
                        const vector<T>& y,
                        const int n,
                        const T xi) {

  T sum=T(0);
  for (int i=0;i<=n;++i) {
    T product = y[i];
    for (int j=0;j<=n;++j) {
      if (i!=j) {
        product*=(xi-x[j])/(x[i]-x[j]);
      }
    }
    sum+=product;
  }
  return sum;
}
```

Coeficientes

- En ocasiones se requieren conocer los coeficientes del polinomio interpolado:

$$y = c_0 + c_1 x + c_2 x^2 + \cdots + c_{N-1} x^{N-1}$$

Coeficientes

- En ocasiones se requieren conocer los coeficientes del polinomio interpolado:

$$y = c_0 + c_1 x + c_2 x^2 + \cdots + c_{N-1} x^{N-1}$$

- donde los coeficientes deben satisfacer el sistema de **Vandermonde**

$$\begin{bmatrix} 1 & x_0 & x_0^2 & \cdot & x_0^{N-1} \\ 1 & x_1 & x_1^2 & \cdot & x_1^{N-1} \\ \vdots & \vdots & \ddots & \vdots & \\ 1 & x_{N-1} & x_{N-1}^2 & \cdot & x_{N-1}^{N-1} \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \\ \vdots \\ c_{N-1} \end{bmatrix} = \begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_{N-1} \end{bmatrix}$$

Coeficientes

- En ocasiones se requieren conocer los coeficientes del polinomio interpolado:

$$y = c_0 + c_1 x + c_2 x^2 + \cdots + c_{N-1} x^{N-1}$$

- donde los coeficientes deben satisfacer el sistema de **Vandermonde**

$$\begin{bmatrix} 1 & x_0 & x_0^2 & \cdot & x_0^{N-1} \\ 1 & x_1 & x_1^2 & \cdot & x_1^{N-1} \\ \vdots & \vdots & \ddots & \vdots & \\ 1 & x_{N-1} & x_{N-1}^2 & \cdot & x_{N-1}^{N-1} \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \\ \vdots \\ c_{N-1} \end{bmatrix} = \begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_{N-1} \end{bmatrix}$$

- Existen métodos especiales para resolver este sistema en $\mathcal{O}\left(n^2\right)$ en vez de $\mathcal{O}\left(n^3\right)$ de los métodos convencionales.

Coeficientes

- En ocasiones se requieren conocer los coeficientes del polinomio interpolado:

$$y = c_0 + c_1 x + c_2 x^2 + \cdots + c_{N-1} x^{N-1}$$

- donde los coeficientes deben satisfacer el sistema de **Vandermonde**

$$\begin{bmatrix} 1 & x_0 & x_0^2 & \cdot & x_0^{N-1} \\ 1 & x_1 & x_1^2 & \cdot & x_1^{N-1} \\ \vdots & \vdots & \ddots & \vdots & \\ 1 & x_{N-1} & x_{N-1}^2 & \cdot & x_{N-1}^{N-1} \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \\ \vdots \\ c_{N-1} \end{bmatrix} = \begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_{N-1} \end{bmatrix}$$

- Existen métodos especiales para resolver este sistema en $\mathcal{O}\left(n^2\right)$ en vez de $\mathcal{O}\left(n^3\right)$ de los métodos convencionales.
- Sistema es usualmente mal condicionado.

Problemas de la interpolación polinomial

- Mientras mayor el orden de la interpolación, menor es la precisión, pues la posibilidad de oscilaciones (que alejan la función real de los datos provistos para la interpolación) es mayor.

Problemas de la interpolación polinomial

- Mientras mayor el orden de la interpolación, menor es la precisión, pues la posibilidad de oscilaciones (que alejan la función real de los datos provistos para la interpolación) es mayor.
- Para interpolación polinomial la peor distribución de datos x_i es la homogénea (igual separación), que lamentablemente es además la más usual.

Problemas de la interpolación polinomial

- Mientras mayor el orden de la interpolación, menor es la precisión, pues la posibilidad de oscilaciones (que alejan la función real de los datos provistos para la interpolación) es mayor.
- Para interpolación polinomial la peor distribución de datos x_i es la homogénea (igual separación), que lamentablemente es además la más usual.
- La interpolación polinomial de orden elevado es mal condicionada: cambios pequeños en los datos producen grandes diferencias en las oscilaciones entre puntos.

Problemas de la interpolación polinomial

- Mientras mayor el orden de la interpolación, menor es la precisión, pues la posibilidad de oscilaciones (que alejan la función real de los datos provistos para la interpolación) es mayor.
- Para interpolación polinomial la peor distribución de datos x_i es la homogénea (igual separación), que lamentablemente es además la más usual.
- La interpolación polinomial de orden elevado es mal condicionada: cambios pequeños en los datos producen grandes diferencias en las oscilaciones entre puntos.
- Con interpolación polinomial, el error decrece conforme crece el orden, pero hasta cierto punto, a partir del cual el error "explota".

Problemas de la interpolación polinomial

- Mientras mayor el orden de la interpolación, menor es la precisión, pues la posibilidad de oscilaciones (que alejan la función real de los datos provistos para la interpolación) es mayor.
- Para interpolación polinomial la peor distribución de datos x_i es la homogénea (igual separación), que lamentablemente es además la más usual.
- La interpolación polinomial de orden elevado es mal condicionada: cambios pequeños en los datos producen grandes diferencias en las oscilaciones entre puntos.
- Con interpolación polinomial, el error decrece conforme crece el orden, pero hasta cierto punto, a partir del cual el error "explota".
- Por ello, la interpolación se debe realizar "por partes".

Resumen

1. Introducción
2. Interpolación polinomial
   - Interpolación de Newton
   - Polinomios de interpolación de Lagrange

Este documento ha sido elaborado con software libre incluyendo LaTeX, Beamer, GNUPlot, GNU/Octave, XFig, Inkscape, LTI-Lib-2, GNU-Make y Subversion en GNU/Linux