INTERPOLACIÓN POLINOMIAL, ALGUNAS TÉCNICAS Y SU PROGRAMACIÓN

POLYNOMIAL INTERPOLATION, SOME TECHNIQUES AND THEIR PROGRAMMING

Luis Alejandro Molano Molano *

Recepción 03/02/2012
Evaluación 10/04/2012
Aprobado 17/07/2012

Resumen

En este artículo se estudian algunas técnicas de interpolación polinomial e interpolación polinomial a trozos. Respecto a las primeras, se propone una demostración alternativa de su equivalencia. También se plantea una sencilla codificación de tales métodos en el software Matlab, usando comandos básicos y sin recurrir a rutinas de interpolación predefinidas por este software.

Palabras claves: Interpolación polinomial, interpolación polinomial a trozos, esplines cúbicos.

Abstract

This article explores some polynomial interpolation and also piecewise polynomial interpolation methods. With respect to the first one, it is suggested an alternative demonstration of its equivalence. Also is posed a simple encoding of such methods in Matlab software, by using basic commands, without resorting to interpolation routines predefined by this software.

Keywords: Polynomial interpolation, piecewise polynomial interpolation, cubic splines.

* Magíster en Matemáticas, docente ocasional Escuela de Matemáticas y Estadística UPTC. Correo electrónico: alejomolano.molano85@gmail.com

Rev. Ciencia en Desarrollo, Vol. 4 No. 1,
ISSN 0121-7488 - Diciembre de 2012, pp. 43-69

Introducción

Los polinomios algebraicos están entre las funciones más conocidas y prácticas del cálculo y el análisis. Su importancia radica en que por medio de ellos se puede aproximar uniformemente cualquier función continua con un dominio adecuado, y, entonces, acercarse a cuestiones como la derivación e integración, o, simplemente, el cálculo de un valor particular de la función tiene un tratamiento algebraico relativamente simple. Formalmente se hace referencia al siguiente teorema, que expresa de manera precisa y rigurosa estas ideas y cuya demostración puede verse, por ejemplo, en [1] o [2].

Teorema 1: Sea f una función de valor real, continua sobre un intervalo compacto de la recta real. Entonces f puede ser aproximada uniformemente por polinomios. Dicho de otro modo, para cada $E > 0$, existe un polinomio $P(x)$ tal que si $x \in [a, b]$ entonces $|f(x) - P(x)| < E$

La interpolación polinomial es una herramienta matemática de gran importancia en la teoría de aproximación, y, a grandes rasgos, busca dar solución al siguiente problema: dados los números reales distintos x_i (llamados nodos), con $i = 0,1,2,..., n$, y los reales y_i también con $i = 0,1,3,...,n$, encontrar un polinomio $P_n(x)$ de grado no mayor a n tal que $P_n(x_i) = y$ para $i = 1,2,...,n$. Dicho de otro modo, si cada x_i está en el dominio de una función f, e $y_i = f(x_i)$ entonces se busca que $P_n = y_i$ para $i = 1,2,...n$ y que para valores intermedios, el polinomio sea una aproximación razonablemente buena de la función.

En la práctica, resulta engorroso calcular los coeficientes de los polinomios de interpolación, y, por tanto, puede ser útil programar algoritmos que permitan obtenerlos para grados relativamente grandes, por medio de un software de computador, que permitan, incluso, calcular valores particulares, aunque no se tengan explícitamente los coeficientes de los mismos.

En la primera sección de este escrito, se mencionan los fundamentos teóricos de las principales técnicas de interpolación polinomial, y de interpolación polinomial a trozos, poniendo especial atención en estas últimas, a los esplines cúbicos, como ejemplos de curvas de interpolación de mayor suavidad. En la segunda parte, se propone una demostración alternativa para establecer la equivalencia de los polinomios interpolantes que se obtienen respectivamente

Rev. Ciencia en Desarrollo, Vol 4 No. 1,
ISSN 0121-7488 - Diciembre de 2012, pp. 43-69

con el uso de las técnicas de interpolación polinomial descritas. Finalmente se propondrá una codificación de las técnicas tratadas, con la ayuda del software Matlab. Respecto a este último, la programación que se propone en este escrito no es única, casi todos los textos de análisis numérico exponen algoritmos programables de las técnicas de interpolación que tratan, por ejemplo, si el lector está interesado en ver otro tipo de programación de tales técnicas usando Matlab, en donde se usen directamente rutinas de este software, (programas de propósito general), puede remitirse a [3], por ejemplo; o si quiere ver codificaciones con el mismo software, que arrojen valores particulares de las curvas interpolantes, puede ver por ejemplo [4]. Lo novedoso de las técnicas cuya programación se propone en este escrito (programas de propósito específico), es que todas arrojan explícitamente los coeficientes de los polinomios de interpolación o de cada polinomio cúbico cuando se requiera determinar algún esplín, sin necesidad de usar funciones propias de Matlab, y con una codificación, en cada caso, relativamente sencilla.

Este escrito, fundamentalmente quiere brindar una herramienta que permita un contacto directo con las técnicas básicas de interpolación. De esta forma, se quiere que el docente que trate con este tópico en un curso introductorio al análisis numérico, o de métodos numéricos, disponga de una estrategia para reforzar la enseñanza de los fundamentos de la interpolación, y, en ese sentido, se busca que el estudiante pueda "palpar" lo que en teoría puede resultar tedioso y poco interesante, mediante el uso de un medio computacional, que, sin lugar a dudas, resultará una actividad académica más significativa.

Algunas técnicas de interpolación polinomial

Polinomio de interpolación de Lagrange

La primera técnica de interpolación polinomial que se describe, corresponde al polinomio de interpolación de Lagrange[2]. El siguiente teorema establece la existencia y unicidad de dicho polinomio y la forma de calcularlo; su demostración puede verse en [5 - 7] o en [8].

2 Joseph Louis Lagrange, (1736-1813), nacido en Turín, reconocido por sus importantes contribuciones al cálculo variacional y a la teoría de funciones analíticas. Fundó la Academia de Ciencias de Turín. Enseñó en la Academia de Ciencias de Berlín sucediendo en la dirección a L. Euler. En la última etapa de su vida fue profesor en la Academia de Ciencias de París.

Rev. Ciencia en Desarrollo, Vol. 4 No. 1,
ISSN 0121-7488 - Diciembre de 2012, pp. 43-69

Teorema 2. Sea $n \geq 1$, e $i = 0,1,...,n$. Dados x_i, números reales distintos y y_i, $n + 1$ números reales, existen $n + 1$ polinomios l_k, (llamados polinomios coordenados), de grado n, con $k = 0,1,...,n$, tales que $l_k(x_j)=\delta_{kj}$, donde δ_{kj} representa la función delta de Kronecker, con

$$l_k(x) = \prod_{j=0,\ j\neq k}^{n} \frac{(x - x_j)}{(x_k - x_j)}$$

Adicionalmente existe un único polinomio L_n de grado n, tal que $L_n(x_i) = y_i$, para $i = 0,1,2,...,n$, el cual tiene la forma

$$L_n(x) = \sum_{k=0}^{n} l_k(x)\, y_k$$

Método de Neville

Si se quiere interpolar cierto conjunto J de puntos, la determinación de los polinomios de interpolación relativos a subconjuntos propios, no aporta a la construcción del polinomio interpolante asociado a J. Esto puede ser una desventaja desde el punto de vista algorítmico, y, en este sentido, un método alternativo que usa interpolaciones de grado menor para obtener otras de mayor grado es el método de Neville[3]. El siguiente teorema expresa la forma que toma el polinomio interpolante cuando se usa esta técnica, y su demostración puede verse en [5] o en [9].

Teorema 3. Dados los $k + 1$ nodos $x_i, x_{i+1}, ..., x_{i+k}$, en el dominio de una función f, con i entero no negativo, y suponiendo que $y_j = f(x_j)$ con $i \leq j \leq k + i$, el único polinomio de grado k, que interpola los $k + 1$ puntos, (llamados puntos soporte), (x_j, y_j) notado como $P_k^{[i,i+1,...i+k]}(x)$, tiene la forma

$$P_k^{[i,i+1,\cdots,i+k]}(x) = \frac{(x-x_{i+k})P_{k-1}^{[i,\cdots,i+k-1]}(x)-(x-x_i)P_{k-1}^{[i+1,\cdots,i+k]}(x)}{(x_i-x_{i+k})},$$

donde $P_0^{[i+l]}(x) = y_{i+l}$, con $l = 0,1,..., k$.

3 Eric H. Neville, (1889-1961), matemático inglés nacido en Londres, profesor de Cambridge y de la Universidad de Reading. Reconocido por sus contribuciones en geometría diferencial y educación matemática.

Rev. Ciencia en Desarrollo, Vol 4 No. 1,
ISSN 0121-7488 - Diciembre de 2012, pp. 43-69

Método de Aitken

El método de Aitken[4] es similar al de Neville, en el sentido en que interpolaciones polinomiales de menor grado contribuyen a formar otras de grado mayor. Puede citarse su publicación original [10] o también [11], para ver los detalles de las ideas expuestas en el siguiente:

Teorema 4. Sea $i = 0,1,\ldots,n$. Dado el conjunto de $n + 1$ reales distintos x_i, y los reales y_i, el polinomio de grado n, notado como $P_{0\ldots n}(x)$, que interpola los $n + 1$ puntos de soporte (x_i, y_i), se obtiene a través de la fórmula:

$$P_{0\ldots n}(x) = \frac{1}{x_{n-1} - x_n}\begin{vmatrix} (x - x_n) & P_{0\ldots n-2,n}(x) \\ (x - x_{n-1}) & P_{0\ldots n-1}(x) \end{vmatrix}$$

$$= \frac{(x - x_n)P_{0\ldots n-1}(x) - (x - x_{n-1})P_{0\ldots n-2,n}(x)}{x_{n-1} - x_n}$$

Donde $P_k(x) = y_k$, con $k = 0,1,2,\ldots,n$.

De acuerdo con el teorema anterior, en algún momento deben calcularse polinomios relativos a nodos que no son consecutivos, sin embargo el método es el mismo, si se quiere construir el polinomio relativo a j +1 nodos no consecutivos, se pueden numerar sus subíndices x_{no}, x_{n1},..., x_{nj}, y la fórmula para obtener el polinomio sería la misma:

$$P_{n_0\ldots n_j}(x) = \frac{1}{x_{n_{j-1}} - x_{n_j}}\begin{vmatrix} (x - x_{n_j}) & P_{n_0\ldots n_{j-2},n_j}(x) \\ (x - x_{n_{j-1}}) & P_{n_0\ldots n_{j-1}}(x) \end{vmatrix}$$

$$= \frac{\left(x - x_{n_j}\right)P_{n_0\ldots n_{j-1}}(x) - (x - x_{n_{j-1}})P_{n_0\ldots n_{j-2},n_j}(x)}{x_{n_{j-1}} - x_{n_j}}$$

El proceso iterativo de construcción, en el caso particular $n = 4$, puede verse de la siguiente forma:

$$\begin{array}{c|ccccc} x_0 & P_0 & & & & \\ x_1 & P_1 & P_{01} & & & \\ x_2 & P_2 & P_{02} & P_{012} & & \\ x_3 & P_3 & P_{03} & P_{013} & P_{0123} & \\ x_4 & P_4 & P_{04} & P_{014} & P_{0124} & P_{01234} \end{array}$$

4 Alexander Craig Aitken, (1895-1967) fue un brillante matemático neozelandés nacido en Dunedin. Fue profesor hasta su muerte de la Universidad de Edinburgo en Escocia. Es reconocido por sus importantes aportes al análisis numérico y por su inigualable memoria y genio versátil.

Rev. Ciencia en Desarrollo, Vol. 4 No. 1,
ISSN 0121-7488 - Diciembre de 2012, pp. 43-69

Teniendo en cuenta este esquema las diferencias de la variable y el nodo, respectivamente, en cada entrada de la primera columna de la matriz a la que se le calcula el determinante, se identifican considerando el máximo índice, y su respectivo nodo en cada caso. Por ejemplo, si el polinomio es P_{014}, el esquema sugiere usar a P_{01} y P_{04}, luego respectivamente, los nodos seleccionados serían x_1 y x_4, y por tanto, la expresión del polinomio sería:

$$P_{014}(x) = \frac{1}{x_1 - x_4}\begin{vmatrix}(x - x_4) & P_{04}(x)\\ (x - x_1) & P_{01}(x)\end{vmatrix} = \frac{(x - x_4)P_{01}(x) - (x - x_1)P_{04}(x)}{x_1 - x_4}$$

Note también que de acuerdo con el esquema, la construcción de todos los polinomios de cualquier columna involucra a dos polinomios de la columna anterior: el primero y el que está en su misma fila. Otros esquemas de construcción son posibles para exhibir el método de Aitken, pero, si se desea, en alguna situación particular, hallar el polinomio de interpolación para aproximar un valor específico de un real t_0, debe tenerse en cuenta que, al igual que las técnicas anteriores, los nodos no obedecen a ninguna ordenación particular, por tanto se sugiere que se arreglen considerando su distancia a t_0, es decir, x_0 será el nodo más cercano a t_0, y así sucesivamente. Con este arreglo de los datos dados usando el esquema de construcción descrito en el teorema 4, se espera que el primer polinomio de cada columna represente la mejor estimación posible en el respectivo grado de interpolación, esto es, se espera que $P_0(t_0)$ sea la mejor aproximación a través de un polinomio de grado cero, que $P_{01}(t_0)$ sea la mejor aproximación posible a través de una interpolación de grado 1, y así sucesivamente.

Interpolación de Newton con diferencias divididas

Dado que el polinomio de interpolación de Lagrange es único, evidentemente los polinomios de interpolación, hasta ahora descritos, deben ser el mismo, aunque sus representaciones, aparentemente distintas, pudieran sugerir lo contrario. En la siguiente sección se hará una demostración de la equivalencia de tales fórmulas, y, de esa manera, se enfatizará que solo se está haciendo referencia a técnicas distintas que conducen al mismo polinomio interpolante. Ahora, una nueva técnica consiste en considerar el polinomio, (llamado polinomio de Newton):

Rev. Ciencia en Desarrollo, Vol 4 No. 1,
ISSN 0121-7488 - Diciembre de 2012, pp. 43-69

como la herramienta de interpolación, e intentar encontrar los reales a_k bajo la suposición que este polinomio realmente interpola los puntos soporte. El método se conoce usualmente como método de Newton, y las expresiones que identifican los coeficientes están dadas en términos de lo que se conoce en la literatura como diferencias divididas. El siguiente teorema relata tales aspectos, y su prueba, al igual que un tratamiento amplio y riguroso de las diferencias divididas, puede verse en [11].

$$P_n(x) = a_0 + a_1(x - x_0) + a_2(x - x_0)(x - x_1) + \cdots + a_n(x - x_0)(x - x_1)\cdots(x - x_{n-1})$$

Teorema 5. Sea $i = 0,1,\ldots, n$. Dados los $n + 1$ nodos distintos x_i en el dominio de una función f, y suponiendo que $f(x_i) = y_i$, con $0 \leq i \leq n$, un polinomio $N_n(x)$ de grado n, que interpola los $n + 1$ puntos soporte (x_i, y_i), viene dado por la fórmula:

$$N_n(x) = a_0 + \sum_{k=1}^{n}\left[a_k \prod_{j=1}^{k}(x - x_{j-1})\right]$$

donde $a_0 = y_0$, y para $k = 1,2,\ldots n$, $a_k \equiv f[x_0,x_1,\ldots,x_k]$ $k = 1,2,\ldots,n$, representa la k - *ésima* diferencia dividida que se define recursivamente de la siguiente forma:

$$f[x_i, x_{i+1}, \cdots, x_{i+k}] = \frac{f[x_{i+1}, \cdots, x_{i+k}] - f[x_i, \cdots, x_{i+k-1}]}{x_{i+k} - x_i}$$

Siendo i cualquier entero no negativo y $f[x_j] \equiv y_j$.

Polinomios osculantes

Cuando se desea construir el polinomio de Taylor para cierta función f alrededor de algún punto interior de su dominio, solo se requiere información de la función y sus derivadas en tal punto, dicho de otro modo, un único punto del dominio y mucha información acerca de la variación de f y sus derivadas en este. Por otra parte, se ha visto hasta ahora que para interpolar dos o más puntos soporte, se pide únicamente conocer el valor de la función en los nodos, en otras palabras, "menos" información de f respecto a más puntos del dominio. La situación ideal sería conocer la variación de la función y algunas de sus derivadas, en varios puntos del dominio, y aprovechar toda esta información para encontrar un polinomio que coincida, al igual que sus

Rev. Ciencia en Desarrollo, Vol. 4 No. 1,
ISSN 0121-7488 - Diciembre de 2012, pp. 43-69

derivadas, con los valores de f y de sus derivadas, respectivamente, en cada uno de los puntos. Tales polinomios se conocen como polinomios osculantes, y se describen a continuación.

Definición 1. Sea f una función definida en un intervalo $[a,b]$. Sea $i = 0,1,...,n$. Sea el conjunto de $x + 1$ reales distintos x_i, y suponga que existen $n + 1$ enteros no negativos $m_0, m_1, ..., m_n$, tales que para cada punto x_j, las derivadas hasta el orden m_j existen en ese punto, dicho de otra forma, $f^{(k)}(x_j)$ existe para $k = 0,1,..., m_j$, con $j = 0,1,...,n$. Además, sea $m = max\{m_0, m_1,..., m_n\}$ Entonces si $f \in C^m([a,b])$, se desea encontrar un polinomio de grado a los más N, $O_N(x)$, con $N = \sum_{i=0}^{n} m_i + n + 1$, tal que para cada j, $\frac{d^k}{dx^k} P_N(x_j) = \frac{d^k}{dx^k} f(x_j)$ para $k = 0,1,..., m_j$. Este polinomio se denomina polinomio osculante.

Es claro que esta idea generaliza la construcción del polinomio de Taylor, cuando n = 0, y la construcción de un polinomio de interpolación cuando $m_j = 0$, para todo j. Un estudio detallado de la forma general de este tipo de polinomios, puede verse en [12] o en [13]. Este escrito solo tratará un caso particular de este tipo de polinomios, el cual se menciona a continuación.

Polinomio de Hermite

Se considera ahora un caso particular de polinomios osculantes, conocido como polinomio de Hermite[5], cuando $m_j = 1$, para todo j. La demostración del siguiente teorema puede verse en [5, 14, 9] o [8].

Teorema 6. Sea f una función continuamente diferenciable en el intervalo $[a,b]$, y sean $x_0, x_1,...,x_n$, n + 1 puntos del mismo. Existe un único polinomio de grado a lo más $2n+1$, $H_{2n+1}(x)$, tal que para cada $i = 0,1,...,n+1$, $H_{2n+1}(x_i) = f(x_i)$, y $\frac{d}{dx} H_{2n+1}(x_i) = \frac{d}{dx} f(x_i)$. Dicho polinomio viene dado por la fórmula:

$$H_{2n+1}(x) = \sum_{i=0}^{n} P_i(x) f(x_i) + \sum_{i=0}^{n} Q_i(x) f'(x_i),$$

donde

$$P_i(x) = [l_i(x)]^2 [1 - 2(x - x_i) l_i'(x_i)],$$

y

$$Q_i(x) = [l_i(x)]^2 (x - x_i).$$

5 Charles Hermite, (1822-1901), matemático francés, contribuyó significativamente al desarrollo del álgebra, los polinomios ortogonales y la teoría de números, entre muchas otras. Fue profesor en el College de Francia, en Ecolle Normale y en la Universidad La Sorbona.

Rev. Ciencia en Desarrollo, Vol 4 No. 1,
ISSN 0121-7488 - Diciembre de 2012, pp. 43-69

teniendo en cuenta que $l_i(x)$ representa al i - *ésimo* polinomio coordenado relativo al polinomio de interpolación de Lagrange para los $n + 1$ nodos dados.

Interpolación polinomial a trozos, esplines

Las ideas exhibidas hasta ahora tienen que ver con el uso de un polinomio de interpolación para dar una aproximación global de la función f en cuestión, es decir, se trata un único polinomio de interpolación en todo el intervalo que contiene los nodos. Ahora exhibiremos algunas técnicas que permitan construir una aplicación interpolante que, localmente, es decir, restringida a cada uno de los subintervalos determinados por los nodos, sea un polinomio de cierto grado no muy grande. Este tipo de interpolación a trozos es conocida en la literatura como esplín, y tiene la tendencia de no exhibir las severas oscilaciones que en general tienen los polinomios de grado muy grandes, si se establecen ciertas condiciones de frontera; dicho de otra forma, la curva es "más suave" comparada con cualquier interpolación polinomial. Su uso no solamente se restringe al campo del análisis numérico, sino que también es una herramienta útil en el diseño gráfico asistido por ordenador y se relaciona con la solución de problemas con valores en los bordes de ecuaciones diferenciales ordinarias y parciales. A continuación se tratan los tipos de esplín más recurrentes, y se invita al lector interesado a que se remita, por ejemplo, a [14], para ver otros tipos de interpolación a trozos como los esplines lineales y cuadráticos, interpolación cúbica de Bessel e interpolación de Akima, entre otros.

Esplín cúbico de Hermite

Este método de interpolación polinomial a trozos consiste en construir un polinomio de interpolación de Hermite en cada subintervalo generado por el conjunto de nodos. Nótese que, de acuerdo con el teorema 6, cada uno de estos polinomios debe ser de grado 3. Formalmente se da la siguiente:

Definición 2. Sea f una función continuamente diferenciable en el intervalo $[a,b]$, y sean $x_0, x_1,..., x_n, n + 1$ puntos del mismo, ordenados en forma creciente. Se define el esplín cúbico de Hermite como la aplicación $s \in C^1$

Rev. Ciencia en Desarrollo, Vol. 4 No. 1,
ISSN 0121-7488 - Diciembre de 2012, pp. 43-69

([a,b]), tal que para $j = 0,\ldots, n$, $s(x_j) = f(x_j) = f(x_j)$, y $s'(x_j) = f'(x_j)$, y además para $j = 0,\ldots,n+1$, $s_j \equiv s|_{[xj,xj+1]}$ es el polinomio de Hermite de grado 3 que interpola los nodos x_j y x_{j+1}.

La forma explícita de calcular el esplín cúbico de Hermite y su existencia, se establece en el siguiente teorema cuya sencilla demostración se basa en el hecho de considerar el esplín y su derivada, evaluados en x_j. Se propone, como un buen ejercicio para el lector, completar los detalles de la prueba.

Teorema 7. Bajo las condiciones de la definición anterior, existe un único esplín cúbico de Hermite cuya ecuación en cada subintervalo $[x_j, x_{j+1}]$ es:

$$s_j(x) = c_0 + c_1(x - x_j) + c_2(x - x_j)^2 + c_3(x - x_j)^3,$$

donde $h_j = x_{j+1} - x_j$

$$c_2 = 3\frac{f(x_{j+1}) - f(x_j)}{h_j^2} - \frac{f'(x_{j+1}) + 2f'(x_j)}{h_j}$$

$$c_3 = -\frac{f'(x_{j+1}) + f'(x_j)}{h_j^2} - 2\frac{f(x_{j+1}) - f(x_j)}{h_j^3}$$

Esplines cúbicos

Un esplín cúbico, en términos generales, es un conjunto de polinomios cúbicos, (posiblemente distintos), cuyos grafos están unidos o "pegados" por los puntos soporte relativos a los nodos interiores, de tal forma que se garantice "suavidad" de la curva resultante. La esencia de esta técnica es construir un polinomio cúbico sobre cada subintervalo determinado por la partición formada por los nodos, de tal manera que la función resultante sea diferenciable en cada uno de estos. Un completo y detallado tratamiento de los esplines puede encontrarse en el clásico [14]. Proponemos la siguiente:

Definición 3. Sea f una función definida sobre un intervalo $[a,b]$ y sea $\{a = x_0, x_1, \ldots, x_n = b\}$, $(x_i < x_j, \text{ si } i < j)$, una partición del intervalo. Una función $S: [a,b] \to \mathbb{R}$, se denomina un esplín (o trazador) cúbico interpolante si satisface:

Rev. Ciencia en Desarrollo, Vol 4 No. 1,
ISSN 0121-7488 - Diciembre de 2012, pp. 43-69

- $S \in C^2([a,b])$,
- $S(x_j) = f(x_j)$, para cada $j = 0,1,\dots,n-1$,
- $S|_{[x_j x_{j+1}]} \equiv S_j$ es un polinomio cúbico, para cada $j = 0,1,\dots,n-1$.

Bajo estas condiciones, la aplicación S no es única y su determinación dependerá del tipo de condiciones de frontera que se impongan. En particular, las condiciones más conocidas y convenientes son las siguientes:

1. $S''(a) = S''(b) = 0$
2. $S'(a) = f'(a)\ y\ S'(b) = f'(b)$
3. $S^k(a) = S^k(b),\ para\ k = 1,2,3.$
4. f''' es continua en x_1 y en x_{n-1}.

Satisfaciendo una de las condiciones (1), (2) o (3), S se denomina esplín natural, esplín sujeto a los bordes o esplín periódico, respectivamente. La condición (4) es conocida como "not a knot", y permite establecer que los pares $S_0(x),\ S_1(x)$ y $S_{n-2}(x),\ S_{n-1}(x)$, están descritos, respectivamente, por el mismo polinomio.

Respecto a las tres primeras condiciones, el esplín que determinan posee una propiedad de minoridad, en el sentido de que con la norma de $L_2([a,b])$ la segunda derivada tiene menor magnitud que cualquier otra aplicación en $C^2([a,b])$, que coincida con f en los nodos y que satisfaga alguna de las condiciones (1), (2) o (3). Para enunciar formalmente este hecho, se menciona el siguiente teorema cuya prueba puede verse, por ejemplo, en [11].

Teorema 8. Sea S un esplín que aproxima a f en el intervalo $[a,b]$, satisfaciendo una de las condiciones (1), (2) o (3), y sea $g \in C^2([a,b])$, otra función que coincide con f en los $n+1$ nodos dados, y que satisface la respectiva condición (1), (2) o (3). Entonces S satisface la propiedad de minoridad:

$$\int_a^b (S''(x))^2 dx \leq \int_a^b (g''(x))^2 dx.$$

El siguiente teorema garantiza la existencia de un único esplín para cada condición de borde expuesta anteriormente, e implícitamente, la forma de calcularlo.

Rev. Ciencia en Desarrollo, Vol. 4 No. 1,
ISSN 0121-7488 - Diciembre de 2012, pp. 43-69

Teorema 9. Si S satisface alguna de las condiciones (1), (2), (3) o (4) de la definición anterior, este es único respectivamente, y para $j = 0,1,2,\dots,n-1$:

$$S|_{[x_j,x_{j+1}]} \equiv S_j(x) = a_j + b_j(x - x_j) + c_j(x - x_j)^2 + d_j(x - x_j)^3,$$

donde los coeficientes $\{a_i\}_{i=0}^{n}$ vienen dados por la ecuación $S(x_j) = a_j = f(x_j)$. Los coeficientes $\{c_i\}_{i=0}^{n}$ están determinados por el sistema $Hc = a$, con:

$$H = \begin{bmatrix} \mu_0 & \mu_1 & \mu_2 & \mu_3 & \cdots & \mu_{n-2} & \mu_{n-1} & \mu_n \\ h_0 & 2(h_0+h_1) & h_1 & 0 & \cdots & 0 & 0 & 0 \\ 0 & h_1 & 2(h_1+h_2) & h_2 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & h_{n-2} & 2(h_{n-1}+h_{n-2}) & h_{n-1} \\ \eta_0 & \eta_1 & \eta_2 & \eta_3 & \cdots & \eta_{n-2} & \eta_{n-1} & \eta_n \end{bmatrix}$$

$$\boldsymbol{c} = (c_0, c_1, \cdots, c_n)^T,$$

$$\boldsymbol{a} = \begin{bmatrix} \alpha_0 \\ \frac{3}{h_1}[a_2 - a_1] - \frac{3}{h_0}[a_1 - a_0] \\ \vdots \\ \frac{3}{h_{n-1}}[a_n - a_{n-1}] - \frac{3}{h_{n-2}}[a_{n-1} - a_{n-2}] \\ \alpha_n \end{bmatrix}$$

y con $h_j = x_{j+1} - x_j$. Entonces los coeficientes $\{b_i\}_{i=0}^{n-1}$ y , $\{d_i\}_{i=0}^{n-1}$ vienen dados por las ecuaciones:

$$b_j = \frac{a_{j-1} - a_j}{h_j} - \frac{h_j(2c_j + c_{j+1})}{3},$$

$$d_j = \frac{c_{j+1} - c_j}{3\frac{3}{h_1}[a_2 - a_1] - \frac{3}{h_0}[a_1 - a_0]}$$

En cada una de las cuatro condiciones de frontera se obtienen dos ecuaciones adicionales respectivamente, que determinan la primera y última fila de la matriz de coeficientes H: para la condición (1): $c_0 = c_n = 0$, por tanto $\mu_0 = \eta_n = 1$, $\mu_k = \eta_i = 0$, para $k = 1,\dots,n$, y $i = 0,\dots n-1$; además $\alpha_0 = \alpha_n = 0$.

Para la condición (2),

Rev. Ciencia en Desarrollo, Vol 4 No. 1,
ISSN 0121-7488 - Diciembre de 2012, pp. 43-69

$$2h_0c_0 + h_0c_1 = \frac{3}{h_0}[a_1 - a_0] - 3f'(x_0)$$

y por tanto

$$h_{n-1}c_{n-1} + 2h_{n-1}c_n = 3f'(x_n) - \frac{3}{h_{n-1}}[a_n - a_{n-1}]$$

y por tanto $\mu_0 = 2h_0,\ \mu_1 = h_0,\ \eta_{n-1} = h_{n-1},\ \eta_n = 2h_{n-1},\ \mu_k = \eta_i = 0$, para $k = 2, \dots n,\ y$ $i = 0, \dots, n-2;$ además $\alpha_0 = \frac{3}{h_0}(a_1 - a_0) - 3f'(x_0),\ \alpha_n = 3f'(x_n) - \frac{3}{h_{n-1}}(a_n - a_{n-1}).$

Para la condición (3),

$$c_0 = c_n$$

$$\frac{h_0}{3}(2c_0 + c_1) + \frac{h_{n-1}}{3}(2c_n + c_{n-1}) = \frac{3}{h_{n-1}}[a_n - a_{n-1}] - \frac{3}{h_0}[a_1 - a_0]$$

y entonces $\mu_0 = 1,\ \mu_n = -1,\ \mu_k = 0$ para $k = 1, \dots, n-1,\ \eta_0 = ((2h_0)/3),\ \eta_1 = ((h_0)/3),\ \eta_{n-1} = ((h_{n-1})/3),\ \eta_n = ((2h_{n-1}/3),\ \eta_i = 0$ para $i = 2, \dots, n-2;$ además $\alpha_0 = 0\ y\ \alpha_n = (3/(h_{n-1}))(a_n - a_{n-1}) - (3/(h_0))(a_1 - a_0).$

Para la condición (4),

$$h_1c_0 - (h_0 + h_1)c_1 + h_0c_2 = 0,$$

$$h_{n-2}c_{n-3} - (h_{n-2} + h_{n-1})c_{n-2} + h_{n-2}c_{n-1} = 0,$$

y por tanto $\mu_0 = h_1,\ \mu_1 = -(h_0 + h_1),\ \mu_2 = h_0,\ \mu_k = 0$ para $k = 3, \dots, n,\ \eta_{n-2} = h_{n-2},$ $\eta_{n-1} = -(h_{n-2} + h_{n-1}),\ \eta_n = h_{n-2},\ \eta_i = 0$ para $i = 0, \dots, n-3;$ además $\alpha_0 = \alpha_n = 0.$

La demostración del anterior teorema radica en establecer la no singularidad de la matriz de coeficientes H, lo cual garantiza la existencia de única solución del sistema de $n+1 \times n+1$, $Hc = a$. Las dos ecuaciones adicionales que se obtienen en cada caso, y que permiten completar el sistema, se obtienen de manera relativamente sencilla usando las condiciones de borde respectivas. Nótese que para la condición (1) los polinomios cúbicos $S_1\ y\ S_{n-1}$ carecen de término cuadrático; en la condición (2) se requiere tener información sobre el valor de la derivada de la función f en el primer y último nodo; para (3) se debe

Rev. Ciencia en Desarrollo, Vol. 4 No. 1,
ISSN 0121-7488 - Diciembre de 2012, pp. 43-69

tener en cuenta que $f(x_0) = f(x_n)$, lo cual es producto de la primera de las tres hipótesis que se tienen en esta condición; para la condición "not a knot" forzar la continuidad de la tercera derivada de f implica que las imágenes de los nodos x_1 y x_{n-1}, es decir el primero y el último de los nodos interiores, no son la conexión o el empalme de dos polinomios cúbicos distintos, o en otras palabras, $S_0 = S_1$ y $S_{n-2} = S_{n-1}$. Se invita al lector interesado en este tipo de cuestiones, a que consulte las referencias [15, 14, 5, 3, 6], o [9], para ver los detalles de la prueba del teorema anterior, además para ver representaciones del error de interpolación que se comete en cada uno de los cuatro casos.

Equivalencia de las técnicas de interpolación

En principio, se recurre al conocido teorema fundamental del álgebra que garantiza la existencia de, a lo sumo, n ceros distintos para un polinomio de grado n. Su demostración puede verse, por ejemplo, en [16].

Teorema 10. Si P es un polinomio de grado $n \geq 0$, con coeficientes sobre un campo F, entonces P tiene a lo más n raíces distintas en F.

Ahora esbozamos la sencilla demostración del hecho que si dos polinomios de grado n tienen las mismas raíces, y si el coeficiente principal es el mismo, entonces los polinomios deben ser el mismo.

Teorema 11. Sean P y Q polinomios de grado n, con las mismas raíces $c_1, c_2, \dots, c_n$. Si el coeficiente principal de cada polinomio es el mismo entonces $P = Q$.

Demostración. Definiendo $R(x) = P(x) - Q(x)$, es claro que si el coeficiente principal de cada polinomio es el mismo entonces $grad(R) \leq n - 1$, y que $R(c_i) = 0$, para $i = 1,2,3, \dots n$. Se tiene entonces que $R(x)$, que es un polinomio de grado no mayor a $n - 1$, tiene n ceros. Como consecuencia del teorema anterior, necesariamente R debe ser idénticamente cero, y por tanto $P = Q$.

Ahora se procederá a demostrar el principal resultado de esta sección, el cual consiste en dar una prueba alternativa del hecho que, los polinomios de interpolación obtenidos por los métodos de Lagrange, Diferencias Divididas,

Rev. Ciencia en Desarrollo, Vol 4 No. 1,
ISSN 0121-7488 - Diciembre de 2012, pp. 43-69

Neville y Aitken, son el mismo. La principal demostración de este hecho es consecuencia de la unicidad del polinomio de interpolación. En efecto, enunciamos el siguiente:

Teorema 11. Dados los $n+1$ nodos $x_0, x_1, \ldots, x_n$, y los respectivos polinomios definidos en los teoremas 2, 3, 4 y 5, se tiene que para todo $x \in \mathbb{R}$, $L_n(x) = P_n^{[0,\cdots,n]}(x) = N_n(x) = P_{0\ldots n}(x)$.

Demostración. Se demostrará inicialmente, por inducción sobre n, que $L_n(x) = P_n^{[0,\cdots,n]}(x)$, para cualquier n. Cuando $n = 1$ el resultado es trivial. Se supone que dado un conjunto de r nodos $x_i, x_{i+1}, \ldots, x_{i+r-1}$, con $r \leq k$, se tiene que $L_{r-1}(x) = P_n^{[i,i+1,\cdots,i+r-1]}(x)$ Tenemos, por definición, y sin pérdida de generalidad, que

$$P_k^{[0,1,\cdots,k]}(x) = \frac{(x-x_k)P_{k-1}^{[0,\cdots,k-1]}(x)-(x-x_0)P_{k-1}^{[1,\cdots,k]}(x)}{(x_0-x_k)},$$

Usando la hipótesis de inducción, y dados los polinomios de interpolación de Lagrange de grado $k-1$:

$$L_{k-1}(x) = \sum_{i=0}^{k-1} \prod_{j=0,\ j\neq i}^{k-1} \frac{(x-x_j)}{(x_i-x_j)} y_i$$

$$L_{k-1}(x) = \sum_{i=1}^{k} \prod_{j=1,\ j\neq i}^{k} \frac{(x-x_j)}{(x_i-x_j)} y_i,$$

entonces:

$$P_k^{[0,1,\cdots,k]}(x) = \frac{(x-x_k)}{(x_0-x_k)} \sum_{i=0}^{k-1} \prod_{j=0,\ j\neq i}^{k-1} \frac{(x-x_j)}{(x_i-x_j)} y_i - \frac{(x-x_0)}{(x_0-x_k)} \sum_{i=1}^{k} \prod_{j=1,\ j\neq i}^{k} \frac{(x-x_j)}{(x_i-x_j)} y_i$$

$$= \frac{(x-x_k)}{(x_0-x_k)} \prod_{j=1}^{k-1} \frac{(x-x_j)}{(x_0-x_j)} y_0$$

$$+ \sum_{i=1}^{k-1} \left[\frac{(x-x_k)}{(x_0-x_k)} \prod_{j=0,\ j\neq i}^{k-1} \frac{(x-x_j)}{(x_i-x_j)} - \frac{(x-x_0)}{(x_0-x_k)} \prod_{j=1,\ j\neq i}^{k} \frac{(x-x_j)}{(x_i-x_j)} \right] y_i$$

$$- \frac{(x-x_0)}{(x_0-x_k)} \prod_{j=1}^{k-1} \frac{(x-x_j)}{(x_k-x_j)} y_k$$

Rev. Ciencia en Desarrollo, Vol. 4 No. 1,
ISSN 0121-7488 - Diciembre de 2012, pp. 43-69

Considerando el segundo término en la última suma, este es equivalente a:

$$\sum_{i=1}^{k-1}\left[\prod_{j=1,\ j\neq i}^{k-1}\left[\frac{(x-x_j)}{(x_i-x_j)}\right]\left[\frac{(x-x_k)}{(x_0-x_k)}\frac{(x-x_0)}{(x_i-x_0)}-\frac{(x-x_0)}{(x_0-x_k)}\frac{(x-x_k)}{(x_i-x_k)}\right]\right]y_i$$

$$=\sum_{i=1}^{k-1}\left[\prod_{j=1,\ j\neq i}^{k-1}\left[\frac{(x-x_j)}{(x_i-x_j)}\right]\left[\frac{(x-x_k)}{(x_i-x_k)}\frac{(x-x_0)}{(x_i-x_0)}\right]\right]y_i$$

$$=\sum_{i=1}^{k-1}\left[\prod_{j=0,\ j\neq i}^{k}\left[\frac{(x-x_j)}{(x_i-x_j)}\right]\right]y_i$$

luego se obtiene finalmente

$$P_k^{[0,1,\cdots,k]}(x)=\prod_{j=1}^{k}\frac{(x-x_j)}{(x_0-x_j)}y_0+\sum_{i=1}^{k-1}\left[\prod_{j=0,\ j\neq i}^{k}\left[\frac{(x-x_j)}{(x_i-x_j)}\right]\right]y_i+\prod_{j=0}^{k-1}\frac{(x-x_j)}{(x_k-x_j)}y_k=$$

donde $L_k(x)$ es el polinomio de interpolación de Lagrange para los nodos $x_0, x_1, \ldots, x_k$. Así se obtiene el primer resultado.

Por otra parte, se demostrará que $L_n(x)=N_n(x)$. Nótese que el coeficiente principal de $L_n(x)$ es

$$\sum_{k=0}^{n}\left[\left(\prod_{j=0,\ j\neq k}^{n}(x_k-x_j)\right)^{-1}\right]y_k$$

y que el coeficiente principal de $N_n(x)$, de acuerdo con el teorema 5, es

$$a_n=f[x_0,x_1,\cdots,x_n]=\frac{f[x_1,\cdots,x_n]-f[x_0,\cdots,x_{n-1}]}{x_n-x_0}$$

Se puede demostrar por inducción [8] que

$$f[x_0,x_1,\cdots,x_n]=\sum_{k=0}^{n}\left[\left(\prod_{j=0,\ j\neq k}^{n}(x_k-x_j)\right)^{-1}\right]y_k$$

Rev. Ciencia en Desarrollo, Vol 4 No. 1,
ISSN 0121-7488 - Diciembre de 2012, pp. 43-69

así, los coeficiente principales de $L_n(x)$ y $N_n(x)$ son el mismo, y de acuerdo con sus definiciones, evidentemente sus raíces son las mismas, (los nodos). Usando el teorema anterior, se tiene que $L_n(x) = N_n(x)$.

También por inducción, mostraremos que $L_n(x) = P_{0\ldots n}(x)$ para todo n. Para el caso $n = 1$, el resultado es trivial. Ahora, dado un conjuntos de $r + 1$ nodos $x_i, x_{i+1}, \ldots, x_{i+r}$, con $r + 1 \leq k$, supongamos que $L_r(x) = P_{i\ldots r+i}(x)$. Sin pérdida de generalidad, tomemos los $k + 2$ nodos $x_0, x_1, \ldots, x_{k+1}$. Tenemos, por definición, que

$$P_{0\cdots k+1}(x) = \frac{(x - x_{k+1})P_{0\cdots k}(x) - (x - x_k)P_{0\cdots k-1,k+1}(x)}{x_k - x_{k+1}}$$
$$= \frac{(x - x_{k+1})}{(x_k - x_{k+1})}P_{0\cdots k}(x) - \frac{(x - x_k)}{(x_k - x_{k+1})}P_{0\cdots k-1,k+1}(x)$$

usando la hipótesis de inducción

$$P_{0\cdots k+1}(x) = \frac{(x - x_{k+1})}{(x_k - x_{k+1})}\sum_{j=0}^{k}\prod_{i=0,\ j\neq i}^{k}\frac{(x - x_i)}{(x_j - x_i)}y_j$$
$$-\frac{(x - x_k)}{(x_k - x_{k+1})}\sum_{j=0,\ j\neq k}^{k+1}\prod_{i=0,\ i\neq j,\ i\neq k}^{k+1}\frac{(x - x_i)}{(x_j - x_i)}y_j,$$

desarrollando el último término de cada suma, teniendo en cuenta el respectivo factor que las afecta, se obtiene:

$$P_{0\cdots k+1}(x) = \frac{(x - x_{k+1})}{(x_k - x_{k+1})}\sum_{j=0}^{k-1}\prod_{i=0,\ j\neq i}^{k}\frac{(x - x_i)}{(x_j - x_i)}y_j + \prod_{i=0,\ i\neq k}^{k+1}\frac{(x - x_i)}{(x_k - x_i)}y_k$$
$$-\frac{(x - x_k)}{(x_k - x_{k+1})}\sum_{j=0}^{k-1}\prod_{i=0,\ i\neq j,\ i\neq k}^{k+1}\frac{(x - x_i)}{(x_j - x_i)}y_j + \prod_{i=0,\ i\neq k+1}^{k+1}\frac{(x - x_i)}{(x_{k+1} - x_i)}y_{k+1},$$

y compactando en una sola suma, se tiene:

$$P_{0\cdots k+1}(x) = \sum_{j=0}^{k-1}\left[\frac{(x - x_{k+1})}{(x_k - x_{k+1})}\prod_{i=0,\ j\neq i}^{k}\frac{(x - x_i)}{(x_j - x_i)}y_j - \frac{(x - x_k)}{(x_k - x_{k+1})}\prod_{i=0,\ i\neq j,\ i\neq k}^{k+1}\frac{(x - x_i)}{(x_j - x_i)}y_j\right]$$
$$+\prod_{i=0,\ i\neq k}^{k+1}\frac{(x - x_i)}{(x_k - x_i)}y_k + \prod_{i=0,\ i\neq k+1}^{k+1}\frac{(x - x_i)}{(x_{k+1} - x_i)}y_{k+1}.$$

Rev. Ciencia en Desarrollo, Vol. 4 No. 1,
ISSN 0121-7488 - Diciembre de 2012, pp. 43-69

Dentro de la suma, multiplicando por $\frac{(x_j - x_{n+1})}{(x_j - x_{n+1})}$ y $\frac{(x_j - x_n)}{(x_j - x_n)}$, respectivamente en cada uno de los términos dentro de la suma, se obtiene:

$$\begin{aligned} P_{0\cdots k+1}(x) &= \sum_{j=0}^{k-1} \prod_{i=0,\ j\neq i}^{k} \frac{(x-x_i)}{(x_j-x_l)} \left(\frac{(x_j - x_{k+1})}{(x_k - x_{k+1})} + \frac{(x_j - x_k)}{(x_{k+1} - x_k)} \right) y_j + \prod_{i=0,\ i\neq k}^{k+1} \frac{(x-x_i)}{(x_k-x_i)} y_k \\ &+ \prod_{i=0,\ i\neq k+1}^{k+1} \frac{(x-x_i)}{(x_{k+1}-x_i)} y_{k+1} \\ &= \sum_{j=0}^{k-1} \prod_{i=0,\ j\neq i}^{k} \frac{(x-x_i)}{(x_j-x_i)} y_j + \prod_{i=0,\ i\neq k}^{k+1} \frac{(x-x_i)}{(x_k-x_l)} y_k + \prod_{i=0}^{k} \frac{(x-x_i)}{(x_{k+1}-x_l)} y_{k+1} \\ &= L_{k+1}(x), \end{aligned}$$

como se quería demostrar.

Programación de las técnicas de interpolación

Ahora se presenta la codificación de las distintas técnicas que se han tratado a lo largo de este escrito, recordando que el software Matlab dispone de rutinas propias para manipular la mayoría de tales métodos. La intención, en principio, es dar una alternativa mucho más sencilla desde el punto de vista del uso de sentencias básicas propias de ese lenguaje de programación, las cuales resultarán familiares para cualquiera que haya tenido la oportunidad de haber conocido los fundamentos de programación en Matlab. Por otra parte, se quiere compartir el deleite de programar con aquellos que también disfrutan de este arte como herramienta fundamental para exhibir el poder del análisis numérico. Se recomienda tener en cuenta que en todos los códigos que se presentan, el número de nodos que se trabajan es n, (y no $n+1$), y que el subíndice del primero de ellos es 1, (y no 0).

Polinomio de interpolación de Lagrange

El algoritmo construye y almacena las funciones coordenadas en las filas de una matriz cuadrada A de orden n, y luego la opera, desde el punto de vista matricial, con el vector Y que contiene en sus componentes las imágenes de los nodos. Al correr el código, este arroja los coeficientes del polinomio de interpolación de Lagrange, almacenados en el vector L, además de presentar el grafo del polinomio.

Rev. Ciencia en Desarrollo, Vol 4 No. 1,
ISSN 0121-7488 - Diciembre de 2012, pp. 43-69

```
format long
n=input('¿numero de nodos? ');
X=zeros(1,n); Y=zeros(n,1); for i=1:n,
X(1,i)=input('inserte nodo '); Y(i,1)=input('inserte
imagen del nodo '); end
A=zeros(n,n);
for i=1:n, B=[1]; for j=1:n, if i==j, continue, else,
B=(1/(X(1,i)-X(1,j)))*conv([1 -X(1,j)],B); end,
end, A(i,:)=B; end
L=zeros(1,n); for i=1:n, L=L+Y(i,1)*A(i,:); end
disp('Los coeficientes del polinomio de interpolacion
de Lagrange son');
L %muestra coeficientes del polinomio de interpolacion
x=[min(X)-1:0.01:max(X)+1];
for i=1:n %grafica de los puntos a interpolar
        plot(X(1,i),Y(i,1),'h'),grid on, hold on
end, plot(x,polyval(L,x),'r');
```

Método de Newton

En las líneas de código se calculan los coeficientes de Newton, en principio, usando una matriz cuadrada de orden n, D, que en su columna j-ésima almacena las j-ésimas diferencias divididas de tal manera que al terminar el proceso de cálculos arroja una matriz triangular superior representada por D, donde será explícita la manera iterativa en que progresa el cálculo de tales diferencias. La primera fila de la matriz representa de manera ordenada, los coeficientes de Newton. Luego el código presenta el cálculo explícito de los coeficientes del polinomio de interpolación basado en la fórmula descrita en el teorema 5, almacenando cada término polinómico de la forma $a_k \prod_{j=1}^{k}(x - x_{j-1})$, en las filas de la matriz cuadrada P de orden n, y sumando estas filas para arrojar el vector Df que contiene los coeficientes del polinomio de interpolación. Adicionalmente se presenta la gráfica del polinomio interpolante.

```
n=input('inserte numero de nodos '); format long
X=zeros(1,n); Y=zeros(1,n);
for i=1:n, X(1,i)=input('inserte nodo ');
```

Rev. Ciencia en Desarrollo, Vol. 4 No. 1,
ISSN 0121-7488 - Diciembre de 2012, pp. 43-69

```
Y(1,i)=input('inserte imagen del nodo '); end
D=zeros(n,n); for i=1:n, D(i,1)=Y(1,i); end
for i=2:n, for j=1:n+1-i, D(j,i)=(D(j+1,i-1)-
D(j,i-1))/(X(1,j+i-1)-X(1,j)); end, end
D %matriz triangular superior de diferencias
divididas
disp('Los coeficientes de Newton son'); D(1,:)
%coeficientes de Newton
P=zeros(n,n);
P(1,n)=D(1,1);
for i=2:n,  B=1;
for j=1:i-1,  B=conv([1 -X(1,j)],B); end
            B=D(1,i)*B; k=length(B);
for m=1:k, P(i,n+1-m)=B(1,k+1-m); end
end, Df=zeros(1,n); for i=1:n, Df=Df+P(i,:); end
disp('Los coeficientes del polinomio de
interpolacion son'); Df
x=[min(X)-1:0.01:max(X)+1];
for i=1:n, plot(X(1,i),Y(1,i),'h'),grid on, hold
on, end
plot(x,polyval(Df,x),'r'); %grafica del polinomio
```

Método de Neville

Se propone trabajar con una matriz B que varía de dimensión de acuerdo con el grado de los polinomios base que necesita. Inicialmente es una matriz de orden $(n-1)\times 2$, para almacenar en cada fila los polinomios de grado uno dados por la fórmula $P_0^{[i]}(x)$, y luego usando estos polinomios, según el método de Neville, aumentar el número de columnas según como aumenta el grado de los polinomios que se calculan y reducir el número de filas conforme el número de polinomio base por calcular se reduce, hasta llegar a una matriz $1\times n$ que contiene los coeficientes del polinomio interpolante. Dado que las dimensiones de la matriz B van cambiando, al finalizar en cada paso su construcción, antes de cambiar las dimensiones, esta toma la denominación de A, que inicialmente es un vector que contiene las imágenes de los nodos y estas son de hecho cada $P_0^{[i]}(x)$, los cuales son el primer grupo de datos que requieren los $P_1^{[j,j+1]}(x)$.

Rev. Ciencia en Desarrollo, Vol 4 No. 1,
ISSN 0121-7488 - Diciembre de 2012, pp. 43-69

```
format long
n=input('inserte numero de nodos'); X=zeros(1,n);
A=zeros(n,1);
for i=1:n, X(1,i)=input('inserte nodo');
A(i,1)=input('inserte imagen de nodo'); end
for i=1:n, plot(X(1,i),A(i,1),'p'), grid on, hold
on, end
for k=1:n-1, B=zeros(n-k,k+1);
for i=1:n-k,  B(i,:)=(1/(X(1,i)-
X(1,i+k)))*(conv([1 -X(i+k)],A(i,:))-conv([1
-X(i)],A(i+1,:)));  end
A=B;
end
A, T=[min(X)-2:0.01:max(X)];,
plot(T,polyval(A,T));
```

Método de Aitken

La idea base del diseño del código es la misma que el método de Neville, teniendo en cuenta que al aumentar el número de columnas y reducir el número de filas, siempre se debe usar el primer polinomio, es decir, la primera fila de la matriz del paso anterior para construir todos los polinomios que completan la matriz actual.

```
format long, n=input('inserte numero de nodos');
X=zeros(1,n); A=zeros(n,1);
for i=1:n, X(1,i)=input('inserte nodo');
A(i,1)=input('inserte imagen de nodo'); end
for i=1:n, plot(X(1,i),A(i,1),'p'), grid on, hold
on, end
for k=1:n-1
B=zeros(n-k,k+1);
for i=1:n-k
     B(i,:)=(1/(X(1,k)-X(1,i+k)))*(conv([1
     -X(i+k)],A(1,:))-conv([1 -X(k)],A(i+1,:)));
end
A=B;
```

Rev. Ciencia en Desarrollo, Vol. 4 No. 1,
ISSN 0121-7488 - Diciembre de 2012, pp. 43-69

```
end, A, T=[min(X)-2:0.01:max(X)];
plot(T,polyval(A,T));
```

Polinomio de Hermite

De acuerdo con su construcción, aquí se requiere un vector adicional que almacene los valores de la derivada de la función en cada nodo. Primero se almacena cada polinomio coordenado en una matriz de orden n, A, y luego se usa para construir los polinomios $P_i(x) = [l_i(x)]^2[1 - 2(x - x_i)l_i'(x_i)]$, y $Q_i(x) = [l_i(x)]^2 (x - x_i)$, que paralelamente se van operando por $f(x_i)$ y $f'(x_i)$ respectivamente y sumando en la medida en que se van construyendo. Tales sumas totales son P y Q, cuya suma finalmente es H, y ese es el resultado que arroja, al igual que su grafo.

```
n=input('numero de nodos  ');
X=zeros(1,n); Y=zeros(1,n); Z=zeros(1,n);
for i=1:n
  X(1,i)=input('inserte nodo '); Y(1,i)=input('inserte
imagen del nodo  ');
    Z(1,i)=input('inserte valor de la derivada de la
funcion en el nodo  ');
end, A=zeros(n,n);
for i=1:n
    B=[1];
    for j=1:n
        if i==j
            continue
        else
            B=(1/(X(1,i)-X(1,j)))*conv([1 -X(1,j)],B);
        end
    end
    A(i,:)=B;
end
P=zeros(1,2*n); Q=zeros(1,2*n);
for i=1:n
    P=P+Y(1,i)*conv(conv(A(i,:),A(i,:)),[-2*polyval(p
olyder(A(i,:)),X(1,i))  2*X(1,i)*polyval(polyder(A(i,:
```

Rev. Ciencia en Desarrollo, Vol 4 No. 1,
ISSN 0121-7488 - Diciembre de 2012, pp. 43-69

```
)),X(1,i))+1]);
    Q=Q+Z(1,i)*conv(conv(A(i,:),A(i,:)),[1 -X(1,i)]);
end
H=P+Q,    x=[X(1,1)-1:0.01:X(1,n)+1];    for    i=1:n,
plot(X(1,i),Y(1,i),'p'),hold on; end
plot(x,polyval(H,x),'r');
```

Esplín cúbico de Hermite

La idea del código anterior es usada aquí también, solo que se construirán tantos polinomios de Hermite de grado 3 como subintervalos generados por los nodos. El resultado es una matriz G de orden $n \times 4$, n indica el número de subintervalos, y el número de columnas muestra los coeficientes del polinomio de Hermite en cada fila.

```
n=input('numero de nodos '); X=zeros(1,n);
Y=zeros(1,n); Z=zeros(1,n);
disp('recuerde insertar los nodos ordenados en forma
creciente')
for i=1:n
X(1,i)=input('inserte nodo '); Y(1,i)=input('inserte
imagen del nodo ');
Z(1,i)=input('inserte valor de la derivada de la
funcion en el nodo ');
end, for i=1:n, plot(X(1,i),Y(1,i),'p'), grid on,
hold on; end
h=zeros(1,n-1); for i=1:n-1, h(1,i)=X(1,i+1)-X(1,i);
end, G=zeros(n-1,4);
for m=1:n-1
A=zeros(2,2);
for i=m:m+1, B=[1];
      for j=m:m+1
            if i==j, continue, else, B=(1/(X(1,i)-
            X(1,j)))*conv([1 -X(1,j)],B); end
            end, A(i-m+1,:)=B;
end
P=zeros(1,4); Q=zeros(1,4);
```

Rev. Ciencia en Desarrollo, Vol. 4 No. 1,
ISSN 0121-7488 - Diciembre de 2012, pp. 43-69

```
for i=1:2
            P=P+Y(1,m+i-
            1)*conv(conv(A(i,:),A(i,:)),[-2*polyval
            (polyder(A(i,:)),X(1,m+i-1)) 2*X(1,m+i-
            1)*polyval(polyder(A(i,:)),X(1,m
            +i-1))+1]);
            Q=Q+Z(1,m+i-
            1)*conv(conv(A(i,:),A(i,:)),[1
            -X(1,m+i-1)]);
        end
      H=P+Q; x=[X(1,m):0.01:X(1,m+1)];
      plot(x,polyval(H,x)); G(m,:)=H;
 end, G
```

Esplines cúbicos

Como se ha discutido, el cambio de condiciones de borde en el esplín, permite completar un sistema de igual número de ecuaciones e incógnitas, agregando dos ecuaciones más al sistema, las demás ecuaciones son las mismas sin importar la condición de borde. Por tanto, lo único que variará en la codificación de cualquiera de los tipos de esplín, según la condición de borde, será la manera de introducir los datos respectivos en la primera y última fila de la matriz de coeficientes H, y en la primera y última componentes del vector a. A continuación se muestra la codificación de un esplín natural, (tenga en cuenta que la numeración presente en las líneas de código no hace parte del mismo, solo tiene fines descriptivos). El vector h contiene, en sus $n-1$ componentes, los valores h_i de la definición, y la matriz de coeficientes H es una matriz cuadrada de orden n. En las líneas 6 y 9 se hace la introducción de las sentencias relativas al esplín natural y las dos ecuaciones que completan el sistema. Es sobre esas líneas que se hará la variación cuando se quiera cambiar de condiciones de borde.

```
1  format long, n=input('numero de nodos ');
X=zeros(1,n); A=zeros(1,n);
2  for i=1:n, X(1,i)=input('inserte nodo ');
A(1,i)=input('inserte imagen del nodo '); end
3  for i=1:n, plot(X(1,i),A(1,i),'p'), hold on;
```

Rev. Ciencia en Desarrollo, Vol 4 No. 1,
ISSN 0121-7488 - Diciembre de 2012, pp. 43-69

```
end
4  h=zeros(1,n-1);
5  for i=1:n-1, h(1,i)=abs(X(1,i+1)-X(1,i)); end,
H=zeros(n,n);
6  H(1,1)=1; H(n,n)=1;
7  for i=2:n-1, H(i,i-1)=h(1,i-1);
H(i,i)=2*(h(1,i-1)+h(1,i)); H(i,i+1)=h(1,i); end
8  a=zeros(n,1);
9  a(1,1)=0; a(n,1)=0;
10 for i=2:n-1, a(i,1)=(3/h(1,i))*(A(1,i+1)-
A(1,i))-(3/h(1,i-1))*(A(1,i)-A(1,i-1)); end
11 T=rref([H a]); C=(T(:,n+1))'; B=zeros(1,n-1);
D=zeros(1,n-1);
12 for i=1:n-1, B(1,i)=(A(1,i+1)-A(1,i))/h(1,i)-
h(1,i)*(2*C(1,i)+C(1,i+1))/3; end
13 for i=1:n-1, D(1,i)=(C(1,i+1)-C(1,i))/
(3*h(1,i)); end
14 for i=1:n-1
15 Y=[X(1,i):0.1:X(1,i+1)];
16 P=[D(1,i) C(1,i)-3*D(1,i)*X(1,i)
3*D(1,i)*(X(1,i))^2-2*C(1,i)*X(1,i)+B(1,i)
C(1,i)*(X(1,i))^2-D(1,i)*(X(1,i))^3-
B(1,i)*X(1,i)+A(1,i)];
17 plot(Y,polyval(P,Y),'r'); hold on,
18 end
19 M=zeros(4,n-1); for i=1:n-1, M(1,i)=A(1,i);
M(2,i)=B(1,i); M(3,i)=C(1,i); M(4,i)=D(1,i); end,
M'
```

En este código, las líneas 7 y 10 muestran el uso de las $n-2$ ecuaciones que permanecen fijas a pesar del cambio de condiciones de borde. De la línea 14 a 18 se construye y grafica cada uno de los polinomios cúbicos en cada subintervalo. Los vectores A, B, C y D almacenan respectivamente los coeficientes $\{a_j\}_{j=1}^{n-1}$, $\{b_j\}_{j=1}^{n-1}$, $\{c_j\}_{j=1}^{n-1}$, y $\{d_j\}_{j=1}^{n-1}$. Tales vectores, en ese orden, son arrojados después de ejecutar el código como las cuatro columnas, en tal orden, de una matriz M, de $n-1$ filas. Si el esplín con el que se trabaja es sujeto a los bordes, se debe agregar la línea:

Rev. Ciencia en Desarrollo, Vol. 4 No. 1,
ISSN 0121-7488 - Diciembre de 2012, pp. 43-69

```
da=input('inserte el valor de la derivada en
primer nodo');
db=input('inserte el valor de la derivada en
ultimo nodo');
```

al principio del código, por ejemplo en la cuarta línea. Y para almacenar las dos nuevas ecuaciones que completan el sistema, se deben cambiar las líneas 6 y 9 por las siguientes:

```
6  H(1,1)=2*h(1,1); H(1,2)=h(1,1); H(n,n-
1)=h(1,n-1); H(n,n)=2*h(1,n-1);
9  a(1,1)=(3/h(1,1))*(A(1,2)-A(1,1))-3*da;
a(n,1)=3*db-(3/h(1,n-1))*(A(1,n)-A(1,n-1));
```

Si el esplín es periódico, recuerde que las imágenes del primer y último nodo deben ser las mismas, por lo tanto podrían modificarse levemente las líneas en las que se almacenan los nodos y sus imágenes, por ejemplo asignando al valor de la imagen del último nodo, el valor de la imagen del primero, una vez se haya introducido este último. En este caso, las líneas que reemplazan a la 6 y 9 son:

```
6  H(1,1)=1; H(1,n)=-1; H(n,1)=2*h(1,1)/3;
H(n,2)=h(1,1)/3; H(n,n-1)=h(1,n-1)/3;
H(n,2)=2*h(1,n-1)/3;
9  a(1,1)=0; a(n,1)=(3/h(1,n-1))*(A(1,n)-A(1,n-
1))-(3/h(1,1))*(A(1,2)-A(1,1));
```

Para la condición not a knot, las líneas 6 y 9 se cambian por las siguientes:

```
6  H(1,1)=h(1,2); H(1,2)=-(h(1,1)+h(1,2));
H(1,3)=h(1,1); H(n,n-2)=h(1,n-1); H(n,n-1)=-
(h(1,n-1)+h(1,n-2)); H(n,n)=h(1,n-2);
9  a(1,1)=0; a(n,1)=(3/h(1,n-1))*(A(1,n)-A(1,n-
1))-(3/h(1,1))*(A(1,2)-A(1,1));
```

Agradecimientos

El autor agradece al par evaluador, la cuidadosa lectura del manuscrito, pues sus comentarios han contribuido a mejorar la presentación del mismo.

Referencias

[1] R. Bartle, The elements of real analysis, New York: Jhon Wiley & Sons, 1976.

[2] W. Rudin, Principles of mathematical analysis, New York: McGraw-Hill, 1976.

[3] W. Cao, T. Chung, J. Morris & W. Yang, Applied numerical methods using Matlab, Hoboken, New Jersey: John Wiley & Sons, 2005.

[4] J. Kiusalaas, Numerical methods in Engineering with Matlab, New York: Cambridge University Press, 2005.

[5] R. Burden & J. Faires, Numerical Analysis, Pacific Grove, CA: Brooks/Cole Thomson, 2001.

[6] R. Canale & S. Chapra, Numerical methods for engineers: with Software and Programming Applications, New York: McGraw-Hil, 2002.

[7] F. Mayer & E. Süli, An introduction to numerical analysis, UK: Cambridge University Press, 2003.

[8] C. Zarowski, An introduction to numerical analysis for electrical and computer engineers, Hoboken, New Jersey: John Wiley & Sons, 2004.

[9] J. Stoer & R. Bulirsch, Introduction to numerical analysis, New York: Springer-Verlag, 1980.

[10] A. Aitken, "On interpolation by iteration of proportional parts, without the use of differences", Proc. Edinburgh Math. Soc., pp. 56-76, 1932.

[11] F. B. Hildebrand, Introduction to numerical analysis, New York: McGraw-Hill, 1974.

[12] T. Fort, Finite differences and difference equations in the real domain, Fair Lawn, N.J: Oxford University Press, 1948.

[13] M. J. D. Powell, Approximation theory and methods, Cambridge: Cambridge University Press, 1981.

[14] C. De Boor, A practical guide to splines, Berlín: Springer-Verlag, 1978.

[15] G. Birkhoff & C. De Boor, "Error bounds for spline interpolation", Journal of Mathematics and Mechanics, pp. 13, 827-836, 1964.

[16] N. Jacobson, Lectures in abstract Algebra. Vol I. Basic Concepts, Princeton, New Jersey: Van Nostrand, 1951.

Rev. Ciencia en Desarrollo, Vol. 4 No. 1,
ISSN 0121-7488 - Diciembre de 2012, pp. 43-69