Um algoritmo triangular para interpolação osculatória

Mário M. Graça
Departamento de Matemática
Instituto Superior Técnico
Universidade Técnica de Lisboa
e-mail: mgraca@math.ist.utl.pt

Resumo: Tendo em vista uma abordagem uniforme dos problemas de interpolação simples e interpolação osculatória, define-se uma base polinomial aqui designada por base de Newton generalizada. É construído um certo sistema triangular inferior cuja solução, obtida recursivamente, produz os coeficientes do polinómio interpolador e/ou osculatório para um dado conjunto de n nós distintos. O algoritmo obtido é flexível, permitindo facilmente a actualização dos resultados após inclusão de um novo nodo na tabela de valores funcionais. Testa-se o problema clássico de Runge, respectivamente para nós equidistantes e nós de Chebyshev, usando aritmética exacta ou dados racionalizados em função de uma tolerância predefinida. Como ilustração, é construído um polinómio osculador da função de Runge, de coeficientes racionais e de grau 94. O respectivo integral é comparado com o que se obtém para outros polinómios interpoladores e osculatórios de grau inferior.

Abstract In order to model under a common framework the interpolating and oscullating polynomial problem, a generalized Newton's basis for polynomials is defined. A certain lower triangular linear system is constructed whose recursive solution gives the coefficients for the interpolating and/or oscullatory polynomials for a set of n distinct nodes. The algorithm is flexible, enabling an easy update of the computed results by inclusion of a new node in a table of functional values. The classical Runge problem is tested, respectively with equidistant or Chebyshev nodes, using exact arithmetic or rationalized data within a predefined tolerance. As an illustration, a 94 degree oscullating polynomial with rational coefficients is obtained for the Runge function, and the respective integral is compared with other interpolatory and oscullatory polynomials of lower degree.

palavras-chave: Polinómio interpolador, polinómio osculador, base de Newton generalizada, sistema linear triangular, função de Runge.
keywords: Interpolating polynomial, osculating polynomial, generalized Newton's basis, triangular linear system, Runge function.

1 Introdução

Dada a expressão analítica de uma função real f, definida num certo intervalo $[a, b]$, bem como as expressões das suas derivadas, pretendemos usar uma abordagem uniforme a fim de obter a solução para cada um dos seguintes problemas clássicos:
(i) Interpolação polinomial simples de uma tabela contendo n ($n \geq 1$) valores funcionais $\{t_i, f(t_i)\}_{i=1}^{i=n}$, dados n nós distintos $t_i \in [a, b]$;
(ii) Interpolação osculatória, ou seja, a construção de um polinómio que não só interpole os valores da tabela referida, como os valores das derivadas da função $f'(t_i)$ (problema habitualmente designado por interpolação de Hermite); interpolação de Hermite generalizada considerando derivadas de ordem superior a um.[1]
(iii) Construção de fórmulas de quadratura interpolatórias ou osculatórias.

Os problemas (i) a (iii) possuem grande relevância nas aplicações, podendo ser facilmente modelados mediante sistemas lineares. De facto, o uso de sistemas lineares neste contexto constitui uma abordagem natural e unificadora dos problemas em questão, evitando a construção de algoritmos particulares para cada caso específico. Porém, na literatura, essa via unificadora é habitualmente abandonada visto que os sistemas em causa são frequentemente mal condicionados à medida que o número de nós n aumenta, ou seja, tais sistemas são muito sensíveis a erros nos dados, sendo por conseguinte considerados computacionalmente inviáveis (ver por exemplo [1], [2], [3], [4], [6], [9], [10], [12]). Na origem do mau condicionamento desses sistemas está a base usualmente utilizada para o espaço dos polinómios reais, ou seja, a base canónica $\{1, x, x^2, \ldots, x^k\}$, para um certo inteiro k positivo fixo. Por exemplo, é bem conhecido que o problema (i) de interpolação de uma tabela de n nós pode ser modelado por um sistema de Vandermonde,[2] de n equações e n incógnitas. No entanto um tal sistema sofre do inconveniente duplo de, por um lado, possuir matriz densa e, por outro, ser geralmente mal condicionado mesmo para valores de n relativamente baixos. O facto da referida matriz de Vandermonde ser densa implica que um algoritmo para a obtenção da sua solução (mesmo que se efectuem cálculos exactos, e por conseguinte não se coloque o problema da estabilidade do algoritmo) seja de complexidade da ordem de n^3 operações aritméticas, justificando-se as-

[1] No presente trabalho utilizamos derivadas até segunda ordem, mas o algoritmo descrito é facilmente generalizável para ordens superiores.
[2] Alexandre Théophile Vandermonde (1735-1796).

sim que se construam algoritmos alternativos que sejam simultaneamente computacionalmente económicos e numericamente estáveis.

É objectivo deste trabalho mostrar que através da escolha apropriada de uma base polinomial (ver Definição 2.1), que designamos por *base de Newton generalizada*, a modelação daí resultante para os problemas (i) e (ii) através de sistemas *triangulares inferiores* habilita-nos a lidar de modo uniforme com os problemas em causa e a minorar os efeitos do eventual mau condicionamento do modelo. Uma abordagem semelhante para o problema (iii) pode ser usada (obtendo-se nesse caso sistemas triangulares superiores) cuja solução dá os chamados pesos de fórmulas de quadratura interpolatória. Este problema está tratado em [8]. Perseguimos assim um objectivo em conformidade com o *motto* de Richard Wesley Hamming [5] que está resumido na sua frase "the purpose of computing is insight not numbers".

O interesse de lidarmos exclusivamente com sistemas triangulares $n \times n$ é múltiplo. A solução de tais sistemas pode obter-se recursivamente com $\mathcal{O}(n^2)$ operações aritméticas, sendo a sua estabilidade numérica geralmente superior ao caso da resolução de um sistema de Vandermonde de matriz cheia. Acresce ainda o facto de sistemas triangulares se prestarem a computação *paralela* mediante algoritmos de complexidade $\mathcal{O}(\log n)$ (ver por exemplo [7]).

As derivadas dos elementos da base de Newton generalizada podem obter-se *recursivamente*, envolvendo um custo computacional da ordem das n operações, pelo que o algoritmo final para interpolação osculatória terá um custo computacional de $\mathcal{O}(n^2)$, admitindo como dados os valores funcionais $f(t_i)$ bem como as derivadas requerida pelo problema em causa. Tanto o cálculo recursivo das referidas derivadas como da solução do sistema triangular, que nos permite obter o polinómio interpolador ou osculatório que dá resposta aos problemas (i) e (ii), é efectuado neste trabalho recorrendo à definição recursiva *dinâmica* de funções, uma facilidade importante e eficaz que é disponibilizada pelo sistema *Mathematica* [13], e que utilizámos no código que desenvolvemos tendo em vista os exemplos ilustrativos apresentados na Secção 3.

Por exemplo, o sistema triangular de $2\,n$ equações que construimos para a interpolação osculatória de Hermite de uma tabela $\{t_i, f(t_i)\}_{i=1}^{i=n}$ permite-nos obter os coeficientes $(c_1,\, c_2,\, \ldots\, ,\, c_n,\, c_{n+1},\, c_{n+2},\, \ldots,\, c_{2n})$ do polinómio de grau menor ou igual a $2\,n-1$ que, nas abcissas t_i, toma os valores $f(t_i)$ e cuja derivada assume os valores $f'(t_i)$. As primeiras n componentes da solução dão-nos o polinómio interpolador da tabela, enquanto as restantes n

componentes fornecem a correcção ao polinómio interpolador de modo que o resultado seja o polinómio osculador de Hermite. A flexibilidade inerente ao sistema triangular em causa permite-nos realizar experiências numéricas que nos habilitam a comparar facilmente, por exemplo, as vantagens eventuais da interpolação osculatória de segunda ordem em comparação com a interpolação simples. Uma vez que actualmente estão disponíveis sistemas de computação simbólica capazes de obter as expressões das derivadas de funções f, a interpolação osculatória deverá ser encarada com o mesmo destaque que se deu no passado à interpolação simples. O processo recursivo para o cálculo da solução dos sistemas triangulares aqui discutidos facilitam esse desiderato.

A base de Newton generalizada aqui usada permite tratar o problema de interpolação osculatória (ii) como uma *correcção* do problema de interpolação (i). Além disso, a actualização de um polinómio interpolador ou osculatório por inclusão de mais um nó t_{n+1} ao suporte $t_1, \ldots, t_n$, pode fazer-se sem interferir com os cálculos anteriores, pelo que nomeadamente o problema prático da escolha conveniente do grau de um polinómio aproximante, ou a observação das suas propriedades osculatórias relativamente a uma dada função f podem realizar-se sem os inconvenientes associados a outras bases polinomiais, como seja a chamada base de Lagrange ou suas generalizações (ver [2], Ch. 4, para uma discussão sobre o interesse comparativo da escolha de bases polinomiais adoptadas na literatura corrente).
É bem conhecido que a base polinomial clássica de Newton [11] é interessante do ponto de vista computacional, sendo que a sua generalização (introduzida no parágrafo 2.1) permite alargar o âmbito da sua aplicação [8]. Alguns autores, como Dahlquist e Å. Björck ([2], Ch. 4), Heath ([6], Ch. 7), e Hamming [5] dão ênfase particular à base de Newton clássica e descrevem as suas múltiplas aplicações. Em particular, e dum modo mais abrangente, o notável texto de Hamming é inteiramente dedicado à exposição de métodos unificadores em matemática computacional. Note-se que já em 1973, ([5], p. 338), este autor previa o enorme progresso que a computação simbólica/numérica/gráfica viria a ter em matemática aplicada.

2 Interpolação polinomial simples e interpolação osculatória

Dado um conjunto de pontos reais distintos (designados por *abcissas* ou *nós*) $\mathcal{N}$, construimos uma base polinomial, a seguir definida em (1), dependendo

desses nós. Tal base é conhecida pela designação de *base de Newton* [11]. Essa base é completada da forma indicada em (2), em função da ordem de osculação pretendida. A uma base $\mathcal{B}_{\mathcal{N}}$ assim completada chamaremos *base de Newton generalizada*. Em particular, qualquer polinómio se pode escrever de forma única na base $\mathcal{B}_{\mathcal{N}}$.
Nos três problemas (i) a (iii) anteriormente referidos começamos por substituir uma dada função f por um certo polinómio gozando de propriedades interpolatórias e ou osculatórias no intervalo $[a, b]$ em questão. Por conseguinte, em todos os casos seremos levados a determinar os coeficientes de tal polinómio na respectiva base $\mathcal{B}_{\mathcal{N}}$, ou seja, a determinar a solução de um certo sistema triangular por substituição progressiva. Nas aplicações que aqui tratamos o sistema linear, além de ser triangular inferior, possui um padrão particular que resulta das expressões que se obtêm por derivação dos elementos da base de Newton generalizada. Uma vez que o determinante de um sistema triangular é igual ao produto dos elementos da diagonal, prova-se facilmente que, sob a hipótese de considerarmos nós distintos, esse determinante é único, donde se conclui que os polinómios que resolvem os problemas (i) a (ii) são também únicos.
O sistema triangular a resolver pode ser convenientemente seccionado em blocos de n equações. Do primeiro bloco resultam recursivamente os coeficientes do polinómio interpolador, do segundo bloco os coeficientes da respectiva correcção de Hermite; do terceiro bloco as correcções devidas à informação das derivadas de segunda ordem da função f nos nós, etc. No parágrafo 2.1 detalhamos o algoritmo para o cálculo dos coeficientes do polinómio p interpolador, do polinómio q de Hermite, e do polinómio r, osculador de segunda ordem de uma tabela dada de n nós distintos.

Na Secção 3 ilustramos a aplicação do algoritmo triangular referido usando a função de Runge $f(x) = \dfrac{1}{1 + 25\,x^2}$, definida no intervalo $x \in [-1, 1]$, para valores tabelados com $n = 2, 4, 8, 16$ e $n = 32$ nós. Construimos os respectivos polinómios interpolador $p_{n-1}(x)$ (de grau $\leq n - 1$), o polinómio de Hermite $q_{2\,n-1}(x)$, e o polinómio osculador de segunda ordem $r_{3\,n-1}(x)$, utilizando respectivamente n nós equidistantes (ver parágrafo 3.1) ou os chamados nós de Chebyshev (ver parágrafo 3.2). Revisitamos assim o célebre fenómeno de Runge, o qual evidencia a incapacidade dos polinómios interpoladores para aproximar uniformemente certas funções, numa malha de nós equidistantes, como é o caso da função de Runge na vizinhança da origem. Por exemplo, para $n = 16$, observa-se que o polinómio interpolador $p_{15}(x)$, num suporte de abcissas equidistantes, se afasta muito da função f

nos extremos do intervalo $[-1,1]$ (ver Figura 2). Por contraste, o polinómio osculador de segunda ordem $r_{95}(x)$, de grau 94, obtido a partir da respectiva base de Newton generalizada associada a um suporte de 16 *nós de Chebyshev*, aproxima bem a função de Runge em todo o intervalo, permitindo-nos, por exemplo, calcular uma aproximação de $\int_{-1}^{1} f(x)\,dx$ com mais de 10 dígitos significativos correctos (ver Tabela 2).

2.1 Algoritmo triangular para interpolação osculatória

Definição 2.1. *Dado o conjunto $\mathcal{N}$ de nós distintos $t_1, t_2, \ldots, t_n$, os polinómios*

$$\begin{cases} \phi_0(x) = 1 \\ \phi_j = \phi_{j-1}(x) \times (x - t_j), \quad 1 \leq j \leq n-1 \end{cases} \tag{1}$$

(onde o polinómio $\phi_j(x)$ tem grau j) é uma base do espaço dos polinómios $\mathcal{P}_{n-1}$, de grau menor ou igual a $n-1$. Esta base é designada por base de Newton [11] associada ao suporte $\mathcal{N}$.
Para formar uma base de $\mathcal{P}_{n+k}$ (onde k será especificado em função do problema osculatório a resolver), completamos a base de Newton com os polinómios definidos por

$$\begin{array}{l} \phi_n(x) = \phi_{n-1}(x)\,(x - t_n) = (x - t_1)\,(x - t_2)\cdots(x - t_n) \\ \phi_j(x) = \phi_{j-1}(x)\,(x - t_r), \quad j \geq n+1, \ \textit{com } r \equiv j \pmod{n}. \end{array} \tag{2}$$

À base de Newton assim completada, $\mathcal{B}_{\mathcal{N}}$, daremos a designação de base de Newton generalizada.

Note que para qualquer $j \geq 1$, os zeros do polinómio $\phi_j(x)$ definidos em (1) e (2) são exactamente $t_1, t_2, \ldots, t_j$. Para o caso de interpolação simples a base é definida apenas por (1). No caso de interpolação de Hermite $k = n-1$, e para interpolação osculatória de segunda ordem $k = 2\,n-1$.

Fixado $n \geq 1$, dadas as abcissas $t_1 < t_2 < \ldots < t_n$ e os valores $f(t_i)$, $f'(t_i)$ e $f''(t_i)$, para $1 \leq i \leq n$, pretende-se determinar os coeficientes c_i do polinómio, de grau $\leq 3\,n-1$,

$$r_{3\,n-1}(x) = \sum_{i=1}^{3\,n} c_i\,\phi_{i-1}(x), \tag{3}$$

onde $\phi_{i-1}(x)$ designa o polinómio, de grau $i-1$, da respectiva base $\mathcal{B}_{\mathcal{N}}$. O polinómio $r_{3\,n-1}$ deverá satisfazer as seguintes $3\,n$ condições interpolatórias

e osculatórias de primeira e de segunda ordens:[3]

$$\begin{cases} r_{3\,n-1}(t_i) = f(t_i) \\ r'_{3\,n-1}(t_i) = f'(t_i), \\ r''_{3\,n-1}(t_i) = f''(t_i) \end{cases} \qquad i = 1, \ldots, n \tag{4}$$

Por exemplo, para $n = 4$ nós, fazendo $y^{(0)} = [y_1, y_2, y_3, y_4]^T = [f(t_1),\ f(t_2),\ f(t_3),\ f(t_4)]^T$, e atendendo ao modo como estão definidos os elementos da base de Newton generalizada, as primeiras 4 condições em (4) traduzem-se no seguinte sistema linear triangular inferior:

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & \phi_1(t_2) & 0 & 0 \\ 1 & \phi_1(t_3) & \phi_2(t_3) & 0 \\ 1 & \phi_1(t_4) & \phi_2(t_4) & \phi_3(t_4) \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ c_3 \\ c_4 \end{bmatrix} = \begin{bmatrix} f(t_1) \\ f(t_2) \\ f(t_3) \\ f(t_4) \end{bmatrix} \Leftrightarrow A^{(0)}\, c^{(0)} = b^{(0)} \tag{5}$$

Fazendo $y^{(1)} = [y_1, y_2, y_3, y_4, y_5, y_6, y_7, y_8]^T = [f(t_1), f(t_2), f(t_3), f(t_4),\ f'(t_1),\ f'(t_2),\ f'(t_3),\ f'(t_4)]^T$, as 4 condições de osculação de primeira ordem traduzem-se no sistema $A^{(1)}\, c^{(1)} = y^{(1)}$ a seguir:

$$\begin{bmatrix} 0 & 1 & \phi'_2(t_1) & \phi'_3(t_1) & \phi'_4(t_1) & 0 & 0 & 0 \\ 0 & 1 & \phi'_2(t_2) & \phi'_3(t_2) & \phi'_4(t_2) & \phi'_5(t_2) & 0 & 0 \\ 0 & 1 & \phi'_2(t_3) & \phi'_3(t_3) & \phi'_4(t_3) & \phi'_5(t_3) & \phi'_6(t_3) & 0 \\ 0 & 1 & \phi'_2(t_4) & \phi'_3(t_4) & \phi'_4(t_4) & \phi'_5(t_4) & \phi'_6(t_4) & \phi'_7(t_4) \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ c_3 \\ c_4 \\ c_5 \\ c_6 \\ c_7 \\ c_8 \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \\ y_7 \\ y_8 \end{bmatrix} \tag{6}$$

Finalmente, para $y^{(2)} = [y_1, \ldots, y_8,\ y_9,\ y_{10},\ y_{11},\ y_{12}]^T = [f(t_1), \ldots,\ f'(t_4),\ f''(t_1),\ f''(t_2), f''(t_3), f''(t_4)]^T$, o sistema triangular inferior cuja solução é constituída pelos coeficientes do polinómio $r_{11}(x)$, resulta da junção das equações em (5) e (6) com as 4 equações do sistema $A^{(2)}\, c^{(2)} = y^{(2)}$, onde $c^{(2)} = [c_1, \ldots, c_{12}]^T$ e

$$A^{(2)} = \begin{bmatrix} 0 & 0 & 2 & \phi''_3(t_1) & \ldots & \phi''_8(t_1) & 0 & 0 & 0 \\ 0 & 0 & 2 & \phi''_3(t_2) & \ldots & \phi''_8(t_2) & \phi''_9(t_2) & 0 & 0 \\ 0 & 0 & 2 & \phi''_3(t_3) & \ldots & \phi''_8(t_3) & \phi''_9(t_3) & \phi''_{10}(t_3) & 0 \\ 0 & 0 & 2 & \phi''_3(t_4) & \ldots & \phi''_8(t_4) & \phi''_9(t_4) & \phi''_{10}(t_4) & \phi''_{11}(t_4) \end{bmatrix}. \tag{7}$$

[3] O algoritmo a estabelecer pode ser facilmente estendido para ordens superiores.

Dado um qualquer número $n \geq 1$ de nós tabelados, atendendo ao padrão observado nos sistemas (5) a (7), facilmente se deduzem fórmulas recursivas para o cálculo da solução do sistema triangular inferior que resulta da aplicação das condições (4), levando em conta a definição dada da base de Newton generalizada para n nós. O sistema em causa é da forma $A\,c = y$, onde o segundo membro é obtido efectuando as seguintes atribuições, respectivamente para $k = 0, 1, 2$:

$$(A) \qquad y_i = f^{(k)}(t_{mod(i,n)}), \quad i = k\,n+1, k\,n+2, \ldots, k\,n+n \tag{8}$$

onde $f^{(0)}(x) = f(x)$.
(B) Cálculo dos coeficientes do polinómio interpolador, $p_{n-1}(x)$:

$$c_i = \frac{y_i - \sum_{j=1}^{i-1} \phi_{j-1}(t_i)\, c_j}{\phi_{i-1}(t_i)}, \quad i = 1, \ldots, n \tag{9}$$

(C) Cálculo dos coeficientes do polinómio osculador de primeira ordem (Hermite), $p_{2\,n-1}(x)$:

$$c_i = \frac{y_i - c_2 - \sum_{j=3}^{i-1} \phi'_{j-1}(t_{n-mod(2\,n,i)})\, c_j}{\phi'_{i-1}(t_{n-mod(2\,n,i)})}, \quad i = n+1, \ldots, 2\,n \tag{10}$$

(D) Cálculo dos coeficientes do polinómio osculador de segunda ordem, $p_{3\,n-1}(x)$:

$$c_i = \frac{y_i - 2\,c_3 - \sum_{j=4}^{i-1} \phi''_{j-1}(t_{n-mod(3\,n,i)})\, c_j}{\phi''_{i-1}(t_{n-mod(3\,n,i)})}, \quad i = 2\,n+1, \ldots, 3\,n \tag{11}$$

(E) A correcção $\Delta p_{n-1}(x)$ a adicionar ao polinómio interpolador $p_{n-1}(x)$, de modo a conter a informação quanto às primeiras derivadas de f nos nós tabelados, tem por expressão

$$\begin{array}{l} \Delta p_{n-1}(x) = \sum_{j=n+1}^{2\,n} c_j\, \phi_{j-1}\,(x), \\ \text{e} \\ q_{2\,n-1}(x) = p_{n-1}(x) + \Delta p_{n-1}(x) \end{array} \tag{12}$$

é o polinómio osculador de Hermite da tabela de dados.
(F) De modo análogo, a correcção a adicionar ao polinómio $q_{2\,n-1}(x)$, de modo a incluir a informação sobre as segundas derivadas de f, tem por expressão

$$\begin{array}{l} \Delta q_{2\,n-1}(x) = \sum_{j=2\,n+1}^{3\,n} c_j\, \phi_{j-1}\,(x), \\ \text{e} \\ r_{3\,n-1}(x) = q_{2\,n-1}(x) + \Delta q_{2\,n-1}(x) \end{array} \tag{13}$$

é o polinómio osculador de segunda ordem da tabela de dados.

O algoritmo triangular anterior, de passos (A) a (F), encontra-se traduzido em linguagem *Mathematica* no código transcrito no Anexo 4. Nesse código, a título de exemplo, fez-se $n = 4$ e usaram-se nós equidistantes utilizando a função de Runge no intervalo $[-1, 1]$. Convida-se o leitor a testar o código para outras funções e para outros nós escolhidos num certo intervalo.

2.2 Existência e unicidade de solução

Para mostrarmos que, fixado o número n de nós distintos, os polinómios p, q e r são únicos, iremos considerar os casos $1 \leq n \leq 3$. A partir do padrão obtido pode mostrar-se por indução matemática que a solução do sistema triangular associado à base $\mathcal{B}_{\mathcal{N}}$ é única e, portanto, são únicos os respectivos polinómios interpolador, e osculador de primeira e segunda ordens.

Seja $n = 1$ e $t_1 \in \mathbb{R}$ o nó único a considerar. A base de Newton generalizada é $\mathcal{B}_1 =< 1, x - t_1, (x - t_1)^2 >$. O sistema triangular a resolver, $A_1\, c = y$, tem a forma

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 2 \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ c_3 \end{bmatrix} = \begin{bmatrix} f(t_1) \\ f'(t_1) \\ f''(t_1) \end{bmatrix}.$$

Atendendo a que $det(A_1) = 2 \neq 0$, existe e é único o polinómio osculador

$$\begin{aligned} r_2(x) &= p_0(x) + \Delta p_0(x) = q_1(x) + \Delta q_1(x) \\ &= c_1 + c_2\,(x - t_1) + c_3\,(x - t_1)^2 \\ &= f(t_1) + f'(t_1)\,(x - t_1) + \frac{f''(t_1)}{2}\,(x - t_1)^2. \end{aligned}$$

Note-se que $r_2(x)$ coincide com o polinómio de Taylor da função f, de segunda ordem, em torno de $x = t_1$, como seria de esperar.

Considerando $n = 2$ nós, $t_1 \neq t_2$, o sistema triangular a resolver, $A_2\, c = y$, possui os seguintes $3\,n$ elementos na sua diagonal principal:

Para obter o polinómio interpolador $p_1(x)$:
$\phi_0(t_1) = 1$
$\phi_1(t_2) = t_2 - t_1$
Para obter $q_3(x)$:
$\phi_2'(t_1) = t_2 - t_1$
$\phi_3'(t_2) = (t_2 - t_1)^2$
Para obter $r_5(x)$:
$\phi_4''(t_1) = 2\,(t_2 - t_1)^2$
$\phi_5''(t_2) = 2\,(t_2 - t_1)^3$.

Assim, $det(A_2) = 2^2(t_2 - t_1)^9 \neq 0$. Por conseguinte, o sistema triangular correspondente possui solução única, logo existem e são únicos, os polinómios $p_1(x)$, $q_3(x)$ e $r_5(x)$.
Para o caso $n = 3$, são os seguintes os elementos da diagonal principal do respectivo sistema triangular $A_3\, c = y$:

Para obter o polinómio interpolador $p_2(x)$:
$\phi_0(t_1) = 1$
$\phi_1(t_2) = t_2 - t_1$
$\phi_2(t_3) = (t_3 - t_2)(t_3 - t_1)$
Para obter $q_5(x)$:
$\phi_3'(t_1) = (t_3 - t_1)\,(t_2 - t_1)$
$\phi_4'(t_2) = -(t_3 - t_2)\,(t_2 - t_1)^2$
$\phi_5'(t_3) = (t_3 - t_2)^2\,(t_3 - t_1)^2 = \phi_2^2(t_3)$
Para obter $r_7(x)$:
$\phi_6''(t_1) = 2\,(t_3 - t_1)^2\,(t_2 - t_1)^2$
$\phi_7''(t_2) = 2\,(t_2 - t_1)^3\,(t_3 - t_2)^2$
$\phi_8''(t_3) = 2\,(t_3 - t_2)^3\,(t_3 - t_1)^3 = 2\,\phi_2^3(t_3)$.

Assim, $det(A_3) = -2^3[(t_3 - t_2)\,(t_3 - t_1)\,(t_2 - t_1)]^9 \neq 0$, logo existem e são únicos os polinómios $p_2(x)$, $q_5(x)$ e $r_7(x)$. Por indução, pode mostrar-se a seguinte

Proposição 2.1. *Dado um suporte $\mathcal{N}$ de n nós reais e distintos, o sistema triangular de $3\,n$ equações e $3\,n$ incógnitas, $A_n\, c = y$, associado à base polinomial $\mathcal{B}_{\mathcal{N}}$, possui solução única e*

$$det(A_1) = 2$$

$$det(A_n) = (-1)^n\, 2^n \left[\prod_{\substack{i\,=\,n,\,n-1,\,\ldots,\,1 \\ 1 \,\leq\, j \,\leq\, i-1}} (t_i - t_j) \right]^9 \neq 0, \qquad n \geq 2.$$

Corolário 2.1. *Dado um suporte $\mathcal{N}$, constituído por n nós distintos num intervalo $[a, b]$, os polinómios interpolador $p_{n-1}(x)$, osculador de primeira ordem $q_{2\,n-1}(x)$ e osculador de segunda ordem $r_{3\,n-1}(x)$ são únicos. Os respectivos coeficientes obtêm-se recursivamente usando o algoritmo (A)-(F).*

Observação 2.1. Supondo que a função dada f, é suficientemente diferenciável num intervalo $[a, b]$ contendo $\mathcal{N}$, é possível utilizar a via aqui discutida usando exclusivamente os sistemas triangulares associados à base $\mathcal{B}_{\mathcal{N}}$ para

deduzir (de modo uniforme) fórmulas do erro de interpolação ou osculação $f(x)-\Psi(x)$, para $x \in [a,b]$, onde Ψ é o polinómio p_{n-1}, $q_{2\,n-1}$ ou $r_{3\,n-1}$. Não faremos essa dedução aqui, mas o leitor é convidado a consultar as demonstrações clássicas para o erro de interpolação ou osculação que encontra nas referências. Poderá constatar que nenhuma das demonstrações aí apresentadas (provas essas por vezes longas e complicadas) abarca simultaneamente os casos interpolatório e osculatório. Como ponto de partida dessas provas encontrará o referido polinómio interpolador de Lagrange ou suas extensões para o caso osculatório, levando inevitavelmente à elaboração de demonstrações particulares para cada caso. Pelo contrário, na abordagem adoptada neste trabalho procurou-se uma via unificadora que facilite a construção de um algoritmo comum, bem como o tratamento da questão do respectivo erro de interpolação ou osculação, sem sair do quadro da resolução de sistemas triangulares associados à base de Newton generalizada.

3 Exemplos

O célebre fenómeno de Runge mostra que nem sempre é possível aproximar uniformemente uma função contínua f, definida num intervalo $[a,b]$, usando *polinómios interpoladores* com nós equidistantes. Em particular, não é verdade que um polinómio interpolador de grau elevado aproxime sempre melhor uma dada função, em todo o intervalo, do que um polinómio interpolador de grau inferior. Como veremos nos parágrafos 3.1 e 3.2, ao comparar os resultados que se obtêm, por exemplo, para a função de Runge $f(x) = \dfrac{1}{1+25\,x^2}$, no intervalo $[-1,1]$, respectivamente usando n nós equidistantes nesse intervalo, verifica-se que o polinómio interpolador para nós equidistantes se afasta da função na proximidade dos extremos do intervalo, e esse afastamento é tanto maior quanto maior for n. Verifica-se ainda que esse afastamento é mais notório quando passamos de um polinómio interpolador a um polinómio osculador.

A esta circunstância decepcionante relativamente ao uso de polinómios interpoladores para efeitos de aproximação de funções dá-se habitualmente a designação de fenómeno de Runge. No entanto se, em vez de nós equidistantes, usarmos os chamados nós de Chebyshev, o polinómio interpolador em n nós, $p_n(x)$, aproxima-se da função de Runge quando n aumenta, conforme aqui verificamos experimentalmente para $2 \leq n \leq 32$. O exemplo de Runge mostra a importância da escolha do conjunto de suporte $\mathcal{N}$ no que toca a minorar o erro de um polinómio interpolador ou osculatório.

Começámos por calcular o polinómios interpolador $p_n(x)$, usando aritmética exacta, respectivamente para $n = 2, 4, 8$ e 16 nós equidistantes,

$$t_i = -1 + \frac{2\,i}{n-1}, \quad i = 0, \ldots, n-1.$$

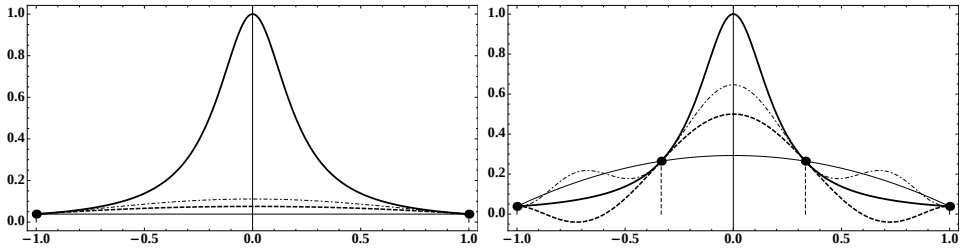

Figura 1: À esquerda $n = 2$ e à direita $n = 4$ nós equidistantes.

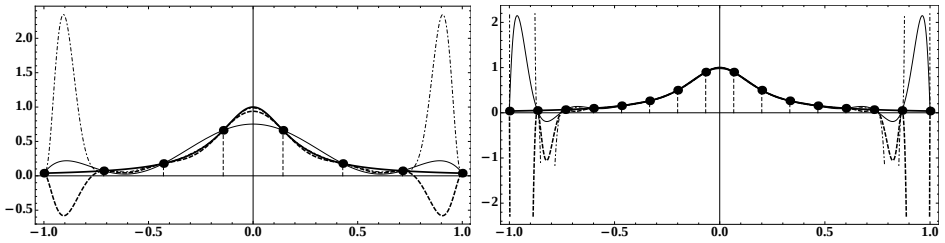

Figura 2: À esquerda $n = 8$ e à direita $n = 16$ nós equidistantes.

Comparámos $p_{n-1}(x)$ com o polinómio de Hermite $q_{2\,n-1}(x)$ e o polinómio osculador de segunda ordem $r_{3\,n-1}(x)$. O fenómeno de Runge é ainda mais pronunciado para os casos de $q_{2\,n-1}(x)$ e $r_{3\,n-1}(x)$ (ver Figuras 1 e 2). Uma vez que iremos usar os polinómios p, q e r para obtermos aproximações do integral $I(f) = \int_{-1}^{1} f(x)\,dx$, concluimos que a interpolação com pontos equidistantes não poderá ser usada para esta finalidade.

Na Tabela 1 comparam-se os erros de integração para cada caso. Por exemplo, verifica-se que para $n = 4$ o erro de quadratura resultante de integrarmos o polinómio osculador $q_{11}(x)$ é muito inferior quando comparado com o que resulta dos polinómios interpolador $p_3(x)$ e de Hermite $q_7(x)$. No entanto, para $n \geq 8$ os erros de quadratura são cada vez maiores à medida que n aumenta. O leitor poderá refazer os cálculos aplicando o código *Mathematica* fornecido no Anexo 4.

n	$e(p_{n-1})$	$e(q_{2\,n-1})$	$e(r_{3\,n-1})$
2	0.472	0.423	0.385
4	0.133	0.217	0.00085
8	−0.0304	0.202	−0.503
16	−0.282	12.0	−501

Tabela 1: Para n nós equidistantes, tabela dos erros de quadratura (arredondados a 3 dígitos decimais), $e(p_{n-1}) = \int_{-1}^{1} f(x) - p_{n-1}(x)\,dx$, $e(q_{2\,n-1}) = \int_{-1}^{1} f(x) - q_{2\,n-1}(x)\,dx$ e $e(r_{3\,n-1}) = \int_{-1}^{1} f(x) - r_{3\,n-1}(x)\,dx$.

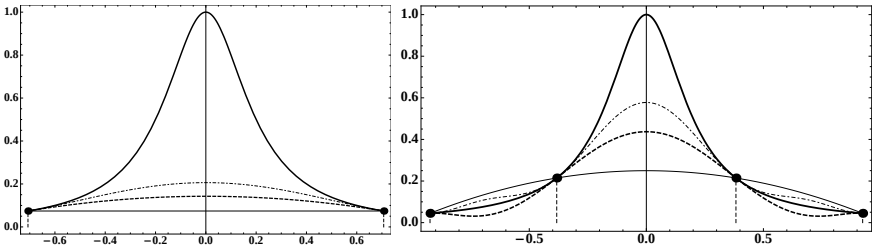

Figura 3: À esquerda $n = 2$ nós; À direita $n = 4$ nós de Chebyshev.

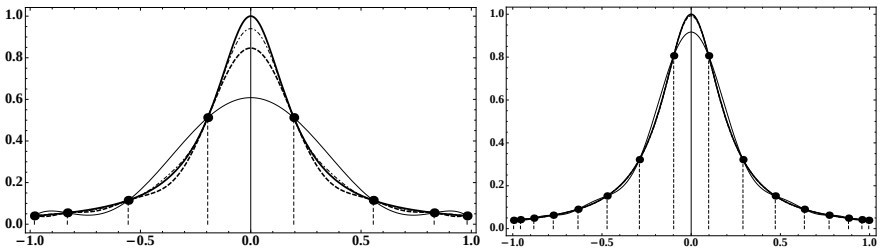

Figura 4: À esquerda $n = 8$ nós. À direita $n = 16$ nós de Chebyshev.

Os nós de Chebyshev ([2], p. 101) localizam-se simetricamente no interior do intervalo $[-1, 1]$, e definem-se (se ordenados de modo crescente) como

$$t_i = -\cos\left(\frac{2\,i-1}{2\,n}\,\pi\right), \qquad i = 1, 2, \ldots, n.$$

As Figuras 3 e 4 mostram que tanto os polinómios $p_{n-1}(x)$, como os osculadores $q_{2\,n-1}(x)$ e $r_{3\,n-1}(x)$ aproximam cada vez melhor a função em todo o intervalo, à medida que n aumenta, sendo que o polinómio osculador

n	$e(p_{n-1})$	$e(q_{2\,n-1})$	$e(r_{3\,n-1})$
2	0.413	0.349	0.301
4	0.208	0.189	0.094
8	0.050	0.044	0.006
16	0.002	0.002	0.00001
32	3.36×10^{-6}	3.30×10^{-6}	3.07×10^{-11}

Tabela 2: Para n nós de Chebyshev, tabela dos erros de quadratura $e(p_{n-1}) = \int_{-1}^{1} f(x) - p_{n-1}(x)\,dx$, $e(q_{2\,n-1}) = \int_{-1}^{1} f(x) - q_{2\,n-1}(x)\,dx$ e $e(r_{3\,n-1}) = \int_{-1}^{1} f(x) - r_{3\,n-1}(x)\,dx$. Para $n < 32$ usaram-se abcissas racionalizadas, com tolerância $\delta = 10^{-8}$. Para $n = 32$, fez-se $\delta = 10^{-16}$.

de segunda ordem é claramente superior aos restantes. Consequentemente, o integral $\int_{-1}^{1} r_{3\,n-1}(x)\,dx$ deverá aproximar o valor exacto do integral da função $f(x)$ melhor do que no caso do respectivo polinómios interpolador e de Hermite, conforme se evidencia na Tabela 2.

3.1 Exemplos para nós equidistantes

Nas figuras apresentadas neste parágrafo e no parágrafo 3.2 estão desenhados a traço cheio o gráfico de $f(x)$ e a traço fino o polinómio interpolador $p_{n-1}(x)$. A tracejado representa-se o polinómio osculador de primeira ordem $q_{2\,n-1}(x)$, enquanto a ponteado é apresentado o polinómio osculador de segunda ordem $r_{3\,n-1}(x)$.
No caso de nós equidistantes $t_i \in [-1, 1]$, o cálculo dos polinómios osculadores mediante aplicação do algoritmo descrito no parágrafo 2.1 foi efectuado com aritmética exacta, por aplicação do código dado no Anexo 4. Por exemplo, para $n = 4$, obtém-se:

$$p(x) = 1/26 + 75(1+x)/221 - 225/884(1/3+x)(1+x).$$

$$\begin{aligned} q(x) \;&= p(x) + 5625(-1+x)(-1/3+x)(1/3+x)(1+x)/22984 \\ &\quad +421875(-1+x)(-1/3+x)(1/3+x)(1+x)^2/195364 \\ &\quad -1265625(-1+x)(-1/3+x)(1/3+x)^2(1+x)^2/781456, \end{aligned}$$

e

$$\begin{aligned} r(x) \;&= q(x) + 31640625(-1+x)^2(-1/3+x)^2(1/3+x)^2(1+x)^2/20317856 \\ &\quad +2373046875(-1+x)^2(-1/3+x)^2(1/3+x)^2(1+x)^3/172701776 \\ &\quad -7119140625(-1+x)^2(-1/3+x)^2(1/3+x)^3(1+x)^3/690807104. \end{aligned}$$

Note que a função f é par e que os polinómios p, q e r também o são.

3.2 Exemplos para nós de Chebyshev

Por definição, os nós de Chebyshev não são números racionais. No entanto, usámos o algoritmo anteriormente descrito efectuando os cálculos exactamente, após racionalização dos dados t_i. Para o efeito, utilizámos a função $Rationalize[t, \delta]$ do sistema *Mathematica*, a qual produz como resultado o número racional de menor denominador que se encontra a uma distância de t não superior a δ. Nos exemplos que aqui apresentamos, em geral fez-se $\delta = 10^{-prec}$, com $prec = 8$ ou $prec = 16$. Por conseguinte, os resultados obtidos são solução exacta para um problema de interpolação osculatória tal que as abcissas utilizadas diferem das abcissas exactas com um erro absoluto não superior à tolerância δ adoptada. Todos os polinómios calculados possuem coeficientes racionais. Resolvemos assim exactamente um problema perturbado. Uma vez que os cálculos efectuados através do algoritmo que adoptámos são exactos, não se coloca a questão da estabilidade numérica do referido algoritmo.

Agradecimentos
Este trabalho foi apoiado pelo Instituto de Mecânica-IDMEC-LAETA/IST, Centro de Projecto Mecânico, através da FCT (Portugal)/programa POCTI.

4 Anexo (código *Mathematica*)

$n = 4;$
$prec = 8;$
$f[x\_] := 1/(1 + 25\,x^2);$
(* nós equidistantes exactos *)
$xi = -1 + Table[(2\,i)/(n-1), \{i, 0, n-1\}];$
(* extremos do intervalo *)
$aa = Min[xi]; bb = Max[xi];$
(* valores funcionais e derivadas *)
$yi = Map[f, xi];$
$yLi = Map[D[f[x], x] /. x \to \texttt{\#\&}, xi];$
$yLLi = Map[D[f[x], \{x, 2\}] /. x \to \texttt{\#\&}, xi];$
$yi = Join[yi, yLi, yLLi];$

(* definição recursiva e dinâmica de base de Newton generalizada $\phi[i,t]$ *)
$\phi[0, t\_] := 1;$
$\phi[i\_ /;\ 1 \le i \le n, t\_] := \phi[i,t] = Product[t - xi[[j]], \{j, 1, i\}];$
$\phi[i\_ /;\ Mod[i, n] == 0, t\_] := \phi[i,t] = \phi[i-1, t] * (t - xi[[n]]);$
$\phi[i\_ /;\ n+1 \le i \le (3\,n-2), t\_] := \phi[i,t] = \phi[i-1, t] * (t - xi[[Mod[i, n]]])$

(* definição recursiva e dinâmica de derivadas *)
(* de primeira e segunda ordens *)
$d\phi[0, x\_, 1] := d\phi[0, x, 1] = 0;$
$d\phi[0, x\_, 2] := d\phi[0, x, 2] = 0;$
$d\phi[i\_ /; Mod[i, n] != 0, x\_, 1] := d\phi[i, x, 1] =$
$\quad d\phi[i-1, x, 1] * (x - xi[[Mod[i, n]]]) + \phi[i-1, x];$
$d\phi[i\_ /; Mod[i, n] != 0, x\_, 2] := d\phi[i, x, 2] =$
$\quad d\phi[i-1, x, 2] * (x - xi[[Mod[i, n]]]) + 2 * d\phi[i-1, x, 1];$
$d\phi[i\_ /; Mod[i, n] == 0, x\_, 1] := d\phi[i, x, 1] =$
$\quad d\phi[i-1, x, 1] * (x - xi[[n]]) + \phi[i-1, x];$
$d\phi[i\_ /; Mod[i, n] == 0, x\_, 2] := d\phi[i, x, 2] =$
$\quad d\phi[i-1, x, 2] * (x - xi[[n]]) + 2 * d\phi[i-1, x, 1];$

(* definição recursiva da solução c[i] *)
$c[i\_ /; 1 \leq i \leq n] := c[i] = Chop[(yi[[i]] -$
$\quad Sum[\phi[j-1, xi[[i]]] * c[j], \{j, 1, i-1\}])/\phi[i-1, xi[[i]]]];$

$c[n+1] := Chop[(yi[[n+1]] - c[2] -$
$\quad Sum[(d\phi[j-1, xi[[1]], 1]) * c[j],\ \{j, 3, n\}])/d\phi[n, xi[[1]], 1]];$

$c[i\_ /; n+2 \leq i \leq 2*n] := c[i] = (\, m2n = n - Mod[2*n, i];$
$Chop[(yi[[i]] - c[2] -$
$\quad Sum[(d\phi[j-1, xi[[m2n]], 1]) * c[j],$
$\quad\quad \{j, 3, i-1\}])/d\phi[i-1, xi[[m2n]], 1]]\,);$
$c[i\_ /; 2\,n+1 \leq i \leq 3\,n] := c[i] = (\, m3n = n - Mod[3*n, i];$
$Chop[(yi[[i]] - 2c[3] -$
$\quad Sum[(d\phi][j-1, xi[[m3n]], 2]) * c[j],$
$\quad\quad \{j, 4, i-1\})/d\phi[i-1, xi[[m3n]], 2]]\,);$

(* polinómio interpolador *)
$p[x\_] := Sum[c[j] * \phi[j-1, x], \{j, 1, n\}];$
(* correcção a p(x) *)
$\Delta p[x\_] := Sum[c[j] * \phi[j-1, x], \{j, n+1, 2\,n\}];$

(* polinómio de Hermite *)
$q[x\_] := q[x] = p[x] + \Delta p[x];$

(* correcção a q(x) *)
$\Delta q[x\_] := Sum[c[j] * \phi[j-1, x], \{j, 2n+1, 3\,n\}];$
(* polinómio osculador de segunda ordem *)
$r[x\_] := q[x] + \Delta q[x];$

Referências

[1] K. E. Atkinson, *An introduction to numerical analysis*, John Wiley and Sons, 2nd ed., 1989.

[2] G. Dahlquist and Å. Björck, *Numerical Methods in Scientific Computing*, Vol. I., SIAM, 2008.

[3] R. L. Burden and J. D. Faires, *Numerical Methods*, Thomson Brooks Cole, third ed., 2003.

[4] W. Gautschi, *Numerical Analysis - An Introduction*, Birkhäuser, 1997.

[5] R. W. Hamming, *Numerical Methods for Scientists and Engineers*, Dover, New York, 2nd ed., 1986.

[6] M. T. Heath, *Scientific Computing - An Introductory Survey*, McGraw-Hill, 1997.

[7] N. J. Higham, "Stability of parallel triangular system solvers", *SIAM J. Sci. Comput.*, 16(2) (1995), pp. 400-413.

[8] M. M. Graça, "Quadrature as a least-squares and minimax problem", arXiv:1206.0281v1 [math.NA], 1 Jun 2012, pp. 1-22.

[9] R. Kress, *Numerical Analysis*, Springer, 1998.

[10] P. Moin, *Fundamentals of Engineering Numerical Analysis*, Cambridge University Press, 2nd ed., 2010.

[11] J. F. Steffensen, *Interpolation*, Dover, 2nd ed., Boston, 2006.

[12] E. Süli and D. Mayers, *An Introduction to Numerical Analysis*, Cambridge University Press, 2003.

[13] S. Wolfram, *The Mathematica Book*, Wolfram Media, fifth ed., 2003.