Revista Brasileira de Ensino de Física, v. 29, n. 3, p. 481-483, (2007)
www.sbfisica.org.br

Notas e Discussões

Uma generalização do polinômio dos quadrados mínimos condicionado

(A generalization of the constrained least squares polynomial)

Oclide J. Dotto e Adalberto A. Dornelles Filho[1]

Departamento de Matemática e Estatística, Universidade de Caxias do Sul, Caxias do Sul, RS, Brasil
Recebido em 28/11/2006; Aceito em 19/3/2007

Este artigo descreve um programa de computador em MATLAB para a obtenção de um polinômio de ajuste dos quadrados mínimos, condicionado a passar por um conjunto arbitrário de nodos de ancoramento, contanto que ainda retenha um conveniente número de graus de liberdade.
Palavras-chave: ajustamento, quadrados mínimos, condicionamento.

The article describes a computer program in MATLAB for obtaining a least-squares polynomial fit, constrained to pass through an arbitrary set of anchorage nodes, provided a sufficient number of freedom degrees is retained.
Keywords: data fitting, least squares, constraint.

1. Introdução

Em recente artigo [1] mostramos que é possível determinar um polinômio de ajuste dos quadrados mínimos (QM) preestabelecendo que contenha *um* nodo de ancoramento (um ponto fixado; a origem, por exemplo). Chamamos a um polinômio de ajuste desse tipo de *polinômio dos quadrados mínimos condicionado* (PQMC) ou *ancorado*. Nos comentários finais do citado artigo dissemos que se poderia tentar desenvolver um algoritmo que obrigue o polinômio a passar por um conjunto arbitrário de nodos de ancoramento, contanto que ainda retenha um número de graus de liberdade suficientemente elevado. Recentemente conseguimos obter um algoritmo para o PQMC com um número t *qualquer* de nodos de ancoramento. A Fig. 1 ilustra um exemplo de ajustamento do PQMC de ordem $m = 4$ ancorado em $t = 2$ nodos.

2. Algoritmo

Segue o algoritmo, implementado na linguagem do MATLAB. Para auxiliar a compreensão, intercalamos explicações detalhadas de diversos passos desse algoritmo.

```
function p = PQMC(x, y, u, w, m)
% PQMC determina o polinômio dos QM Condicionado.
% Entrada:
%   x, y: vetores das abscissas e ordenadas.
%   u, w: vetores das abscissas e ordenadas dos
%         nodos de ancoramento.
%   m   : ordem do PQMC desejado.
% Saída:
%   p   : vetor dos coeficientes do PQMC.
```

O algoritmo tem, como dados de entrada, os vetores das abscissas $\mathbf{x} = [x_1, \ldots, x_n]^T$ e ordenadas $\mathbf{y} = [y_1, \ldots, y_n]^T$ dos n dados (dados experimentais, por exemplo), os vetores das abscissas $\mathbf{u} = [u_1, \ldots, u_t]^T$ e ordenadas $\mathbf{w} = [w_1, \ldots, w_t]^T$ dos t nodos de ancoramento (pontos que certamente pertencem à função teórica, por exemplo) e a ordem m do PQMC

$$P(x) = p_1 x^m + p_2 x^{m-1} + \cdots + p_m x + p_{m+1},$$

a determinar. Quase sempre a *ordem* coincide com o *grau* do polinômio, mas algumas vezes, muito raramente, o grau é menor que a ordem; isso acontece quando é nulo o coeficiente p_1.

```
% Dados e nodos de ancoramento
n = length(x)  % Número de pontos que constituem
               % os dados
t = length(u)  % Número de nodos de ancoramento
if m < t
    error('m DEVE ser no mínimo igual a t.')
end
```

No MATLAB, o comando `length(x)` determina o tamanho (número de componentes) do vetor **x**. Para que haja algum grau de liberdade do ajustamento polinomial ancorado, a ordem m do PQMC não deve ser

[1]E-mail: aadornef@ucs.br.

menor que o número t de nodos de ancoramento. Por exemplo, se $t = 2$, então o polinômio deve ter grau $m \geq 2$, pois $m = 1$ determinaria a reta passando pelos 2 nodos, sem liberdade para o ajustamento.

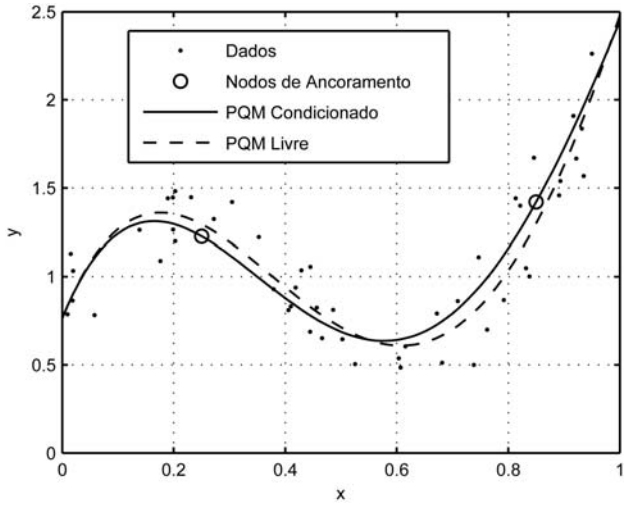

Figura 1 - Ajustamento de um PQMC de ordem 4 ancorado em 2 nodos.

O PQMC procurado pode ser escrito na forma

$$P(x) = Q(x)R(x) + S(x),$$

onde

$$S(x) = s_1 x^{t-1} + s_2 x^{t-2} + \cdots + s_{t-1}x + s_t$$

é o polinômio *interpolador* pelos nodos de ancoramento com ordem $t-1$ e

$$R(x) = (x-u_1)(x-u_2)\cdots(x-u_t)$$

é um polinômio *auxiliar*. Já

$$Q(x) = q_1 x^{m-t} + q_2 x^{m-t-1} + \cdots + q_{m-t}x + q_{m-t+1}$$

é um polinômio de *ajuste* dos QM *livre* de ordem $m-t$.

Por construção, tanto $R(x)$ quanto $S(x)$ não possuem graus de liberdade, pois estão *ancorados* aos nodos $u_1, \ldots, u_t$. Já o polinômio de ajuste $Q(x)$ possui $m-t+1$ graus de liberdade que são, de fato, os graus de liberdade de $P(x)$. Se, para $i = 1, \ldots, t$, temos $R(u_i) = 0$, então $P(u_i) = S(u_i)$, de forma que $P(x)$ fica também *ancorado* aos nodos $u_1, \ldots, u_t$.

```
% S(x): Polinômio interpolador.
V = vander(u);
s = V\w;
```

Inicialmente, determinamos os coeficientes $s_1, \ldots, s_t$ do polinômio interpolador $S(x)$. No MATLAB, o comando `vander(u)` constrói, a partir do vetor $\mathbf{u}$, a matriz de Vandermonde

$$\mathbf{V} = \begin{bmatrix} u_1^{t-1} & \cdots & u_1^2 & u_1 & 1 \\ u_2^{t-1} & \cdots & u_2^2 & u_2 & 1 \\ \vdots & & \vdots & \vdots & \vdots \\ u_t^{t-1} & \cdots & u_t^2 & u_t & 1 \end{bmatrix},$$

e o comando `s = V\w` fornece a solução[2] (clássica) do sistema linear $\mathbf{Vs} = \mathbf{w}$.

```
% Q(x): Polinômio dos QM livre
for i = 1 : n

% R(x): Polinômio auxiliar
R(i) = 1;
for k = 1 : t
  R(i) = R(i) * (x(i) - u(k));
end

% Matriz dos coeficientes
A(i, m-t+1) = R(i);
for j = m-t : -1 : 1
  A(i,j) = A(i, j+1) * x(i);
end

% Termo independente
  b(i,1) = y(i) - polyval(s, x(i));
end
q = A\b;
```

Determinamos os coeficientes $q_1, \ldots, q_{m-t+1}$ do polinômio dos QM livre $Q(x)$ observando que

$$Q(x_i)R(x_i) + S(x_i) = y_i, \quad i = 1, \cdots, n,$$

e que esse conjunto de equações constitui um sistema linear sobredeterminado que deve ser resolvido no sentido dos QM. Ele pode ser reescrito como

$$Q(x_i)R(x_i) = y_i - S(x_i), \quad i = 1, \cdots, n,$$

ou ainda,

$$(q_1 x_i^{m-t} + q_2 x_i^{m-t-1} + \cdots + q_{m-t}x_i + q_{m-t})R(x_i) = y_i - S(x_i) \quad i = 1, \cdots, n.$$

[2] No MATLAB, o comando "\" fornece solução dos QM de um sistema linear $\mathbf{Ax} = \mathbf{b}$. Um sistema linear *pode* não ter solução clássica, mas *sempre* tem solução dos QM. A solução dos QM é obtida pela resolução da equação normal $\mathbf{A}^T\mathbf{Ax} = \mathbf{A}^T\mathbf{b}$. Essa solução dos QM é *única* quando $\mathbf{A}$ tem colunas linearmente independentes (seja $\mathbf{A}$ quadrada ou não). No caso de infinitas soluções de QM, a solução encontrada pelo MATLAB é a de menor norma. Importante: *A solução dos QM coincide com a solução clássica quando esta existe.* O sistema linear $\mathbf{Vs} = \mathbf{w}$ é determinado, isto é, possui uma única solução (clássica). Já o sistema linear $\mathbf{aq} = \mathbf{b}$ é sobredeterminado, isto é, possui apenas solução dos QM.

Com o comando `polyval(s, x(i))` efetuamos o cálculo de $S(x_i)$. Então montamos o sistema linear $\mathbf{Aq} = \mathbf{b}$, onde

$$\mathbf{A} = \begin{bmatrix} R(x_1)x_1^{m-t} & R(x_1)x_1^{m-t-1} & \cdots & R(x_1)x_1 & R(x_1) \\ R(x_2)x_2^{m-t} & R(x_2)x_2^{m-t-1} & \cdots & R(x_2)x_2 & R(x_2) \\ \vdots & \vdots & & \vdots & \vdots \\ R(x_{n-1})x_{n-1}^{m-t} & R(x_{n-1})x_{n-1}^{m-t-1} & \cdots & R(x_{n-1})x_{n-1} & R(x_{n-1}) \\ R(x_n)x_n^{m-t} & R(x_n)x_n^{m-t-1} & \cdots & R(x_n)x_n & R(x_n) \end{bmatrix}$$

e

$$\mathbf{q} = \begin{bmatrix} q_1 \\ q_2 \\ \vdots \\ q_{-t} \\ q_{m-t+1} \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} y_1 - S(x_1) \\ y_2 - S(x_2) \\ \vdots \\ y_{n-1} - S(x_{n-1}) \\ y_n - S(x_n) \end{bmatrix}.$$

O comando `q = A\b` obtém a solução dos QM do sistema linear $\mathbf{Aq} = \mathbf{b}$.

```
% P(x): Polinômio dos QM condicionado.
p = q;
for i = 1 : t
  p = conv(p, [1; -u(i)]);
end
for i = 1 : t
  p(m - t + 1 + i) = p(m - t + 1 + i) + s(i);
end
```

Finalmente obtemos os coeficientes $p_1, \ldots, p_{m+1}$ de $P(x)$. Inicialmente, pela convolução dos vetores $\mathbf{p}$ e $[1, -u_i]^T$ executada pelo comando `conv(p, [1; -u(i)])` e, em seguida, pela soma (devidamente deslocada) do vetor $\mathbf{s}$. Isso é equivalente às operações polinomiais em $Q(x)R(x) + S(x)$.

3. Ancoramento em um único nodo

O método descrito acima é geral. Como caso particular, temos o ajustamento condicionado a conter um *único* nodo (u_1, w_1); nesse caso o PQMC é da forma

$$P(x) = Q(x)(x - u_1) + w_1,$$

onde $Q(x) = q_1x^{m-1} + q_2x^{m-2} + \cdots + q_{m-1}x + q_m$ é um polinômio de ajuste de ordem $m-1$. Obtemos os coeficientes $q_1, \ldots, q_m$ de $Q(x)$ pela resolução (no sentido dos QM) do sistema linear $\mathbf{Aq} = \mathbf{b}$, onde

$$\mathbf{A} = \begin{bmatrix} (x_1 - u_1)x_1^{m-1} & (x_1 - u_1)x_1^{m-2} & \cdots & (x_1 - u_1)x_1 & (x_1 - u_1) \\ (x_2 - u_1)x_2^{m-1} & (x_2 - u_1)x_2^{m-2} & \cdots & (x_2 - u_1)x_2 & (x_2 - u_1) \\ \vdots & \vdots & & \vdots & \vdots \\ (x_{n-1} - u_1)x_{n-1}^{m-1} & (x_{n-1} - u_1)x_{n-1}^{m-2} & \cdots & (x_{n-1} - u_1)x_{n-1} & (x_{n-1} - u_1) \\ (x_n - u_1)x_n^{m-1} & (x_n - u_1)x_n^{m-2} & \cdots & (x_n - u_1)x_n & (x_n - u_1) \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} y_1 - w_1 \\ y_2 - w_1 \\ \vdots \\ y_{n-1} - w_1 \\ y_n - w_1 \end{bmatrix}.$$

4. Conclusão

O programa que acabamos de descrever é melhor que o programa `polqmcond` do referido artigo pelo fato de ser uma generalização deste, e, além disso, usar apenas cálculos numéricos (enquanto o último usa também cálculos simbólicos), o que o torna mais rápido e portável, isto é, pode ser convertido em outra linguagem mais facilmente. Os comandos específicos do MATLAB podem ser implementados pelo usuário, ou mesmo, podem já estar disponíveis em bibliotecas (LAPACK, por exemplo).

Referências

[1] Oclide J. Dotto e Adalberto A. Dornelles Filho, Revista Brasileira de Ensino de Física **28**, 397 (2006).