Manual de ajuste de curvas com Mínimos Quadrados

1 Introdução

O Método dos Mínimos Quadrados foi concebido no final do século XVIII. Bem no início do século XIX, o mundo viu sua primeira aplicação no estudo de trajetórias de corpos celestes. A idéia básica é: dado um conjunto de pontos, achar uma curva que melhor se ajusta a eles. Legendre foi o primeiro a publicar o método em 1805, em seu trabalho *Nouvelles méthodes pour la détermination des orbites des comètes*. Gauss publicou suas conclusões apenas em 1809, embora acredita-se que ele já conhecia o Método desde 1795 [9]. Esse método vem sendo ampliado, generalizado (vide seção 9.3) e largamente utilizado até hoje. Muitas vezes utilizamos uma calculadora ou um computador e nem sabemos que Método dos Mínimos Quadrados lá está, dentro de uma caixa preta. Nada de mau há nisso, a maioria das pessoas que dirigem um automóvel não sabem o princípio de funcionamento de seu motor. Já a maioria das pessoas que andam de bicicleta entendem o princípio de funcionamento dela, embora isso de nada lhes adianta. Nesse Manual pretendemos apenas que o leitor tenha maior autonomia com o Método dos Mínimos Quadrados e utilize-o no computador como uma caixa preta ou não.
O Método dos Mínimos Quadrados é um dos recursos estatísticos dos mais usados em ciências experimentais que pode ser visto, por exemplo, em [11]. Não entraremos no campo da Estatística, seremos bem mais singelos, utilizando o Método dos Mínimos Quadrados a fim de ajustar uma curva a um número finito de pontos, veja figura (9). Esse Manual tem como objetivo principal introduzir os alunos à utilização de softs gratuitos para execução do Método dos Mínimos Quadrados. Não estamos aqui preocupados com a eficiência desses softs, mas sim no encorajamento dos alunos na utilização do computador para tal fim. Para tanto utilizaremos softs amigáveis, nos quais os alunos terão também a facilidade na produção de gráficos, a fim de se ter uma visualização do Método. Além disso será fornecida uma rotina em Linguagem C de simples manipulação, caso o leitor necessite de velocidade de execução do Método.
Existam vários enfoques para se obter o Método dos Mínimos Quadrados. Talvez o mais popular seja aquele que utiliza o Cálculo Diferencial de Funções de Várias Variáveis que pode ser encontrado, por exemplo, em [6]. Outro enfoque utiliza matrizes que pode ser encontrado, por exemplo, em [1] ou [2] e que será discutido na seção 9.4. O enfoque aqui escolhido utiliza a noção intuitiva de distância de ponto a plano, generalizada para distância de ponto a subespaço vetorial, que pode ser encontrado, por exemplo em [4]. Portanto é necessário que leiro lenha noções básicas de produto interno e subespaço vetorial.
O caráter teórico dará ao leitor maior autonomia para interagir com o Método no computador. Por outro lado, o leitor poderá saltar a parte teórica e terá de forma independente um mero manual que utiliza softs amigáveis. Nesse caso, basta ler da seção 6 em diante, sem a necessidade de adentar no Apêndice que é a seção 9.

2 Distância de ponto a subespaço

Vamos, inicialmente, estudar a distância de um ponto a uma reta do ponto de vista vetorial. Para tanto consideremos dois vetores u e v como na Figura 1. Queremos achar a distância do extremo do vetor v à reta suporte do vetor u. Tomemos, então, um múltiplo do vetor u, por exemplo xu. Queremso que o

vetor $v - xu$ seja o menor possível. Definamos a função $f : \mathbb{R} \longrightarrow \mathbb{R}$, por $f(x) = ||v - xu||^2$. Utilizando propriedades de produto interno, podemos escrever f na forma $f(x) = ||u||^2x^2 - 2 < u, v > x + ||v||^2$, que é um polinômio do segundo grau em x, que terá um mínimo (por que?). Usando o cálculo diferencial temos, $f'(x) = 2||u||^2x - 2 < u, v >$. Tal mínimo ocorrerá, se $x = \dfrac{< v, u >}{||u||^2}$. Logo

$$xu = \frac{< v, u >}{||u||^2}u.$$

Esse último vetor é nosso conhecido e se chama projeção de v sobre u.

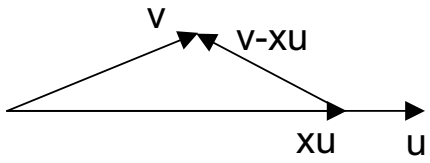


Figura 1: Como achar a diatância de ponto a reta vetorialmente

Uma maneira geométrica de resolver esse problema é observar que na Figura 1 o vetor $v - xu$ deve ser perpendicular ao vetor u conforme Figura 2 e, portanto, $< u, v - xu >= 0$, o que nos daria a mesma

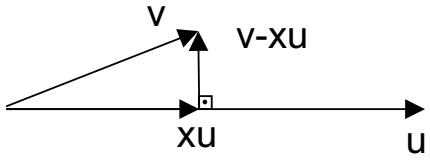


Figura 2: $v - xu$ perpendicular a u.

solução anterior. É esse caráter geométrico que queremos explorar, a fim generalizar a projeção de um vetor em um subespaço. Vamos iniciar com a projeção (ortogonal) de um vetor w no subespaço W gerado pelos vetores u e v, como na Figura 3. Observemos que a projeção deve ser uma combinação linear de u e v, isto é, tal projeção deve ser da forma $au + bv$. Generalizando o caráter geométrico introduzido anteriormente, temos que

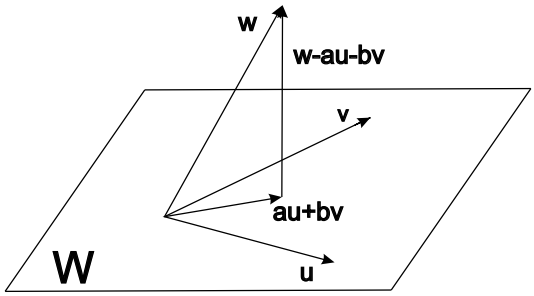


Figura 3: Distância de ponto a plano

$w - au - bv$ deve ser perpendicular a W, conforme Figura 3. Logo $w - au - bv$ deve ser simultaneamente perpendicular a u e a v, isto é,

$$\begin{cases} < w - au - bv, u >= 0 \\ < w - au - bv, v >= 0 \end{cases},$$

que é um sistema linear nas variáveis a e b.

$$\begin{cases} ||u||^2a & + & < u, v > b & = & < u, w > \\ < u, v > a & + & ||v||^2b & = & < v, w > \end{cases}$$

Notemos que a matriz dos coeficientes do sistema é simétrica, o que pode aumentar a velocidade de solução numérica no caso do subespaço W ter dimensão muito grande.
Vamos generalizar para dimensão superior a 2. Suponhamos que V seja um Espaço Vetorial e W um supespaço de V. Tomemos $\{u_1, u_2, \cdots, u_n\}$ uma base de W. Dado um vetor $v \in V$, queremos encontrar um vetor $a_1u_1 + a_2u_2 + \cdots + a_nu_n \in W$, que melhor se aproxima de v. Queremos, na realidade, que

$$||v - (a_1u_1 + a_2u_2 + \cdots + a_nu_n)||^2$$

seja o menor possível, o que nos daria o seguinte sistema linear de n equações e n incógnitas.

$$\begin{cases} < v - (a_1u_1 + a_2u_2 + \cdots + a_nu_n), u_1 >= 0 \\ < v - (a_1u_1 + a_2u_2 + \cdots + a_nu_n), u_2 >= 0 \\ \vdots \\ < v - (a_1u_1 + a_2u_2 + \cdots + a_nu_n), u_n >= 0 \end{cases}$$

ou

$$\begin{cases} < u_1, u_1 > a_1 + < u_2, u_1 > a_2 + \cdots + < u_n, u_1 > a_n =< v, u_1 > \\ < u_1, u_2 > a_1 + < u_2, u_2 > a_2 + \cdots + < u_n, u_2 > a_n =< v, u_2 > \\ \vdots \\ < u_1, u_n > a_1 + < u_2, u_n > a_2 + \cdots + < u_n, u_n > a_n =< v, u_n > \end{cases} . \quad (1)$$

Como a matriz desse sistema é simétrica pode-se usar Decomposição de Cholesky [5] para melhorar a eficiência computacional, mas isso está fora do nosso objetivo.

3 Espaço de Funções

Seja $I \subset \mathbb{R}$ um intervalo (eventualmente I poderá ser uma região de $\mathbb{R}^n$). Vamos considerar o conjunto $\mathcal{F}$ das funções $h : I \longrightarrow \mathbb{R}$, com as seguintes operações de adição e multiplicação por escalar

$$(f + g)(x) = f(x) + g(x) \quad \text{e} \quad (af)(x) = xf(x), \quad \forall f, g \in \mathcal{F} \text{ e } \forall a \in \mathbb{R}$$

Com essas operações $\mathcal{F}$ se torna um espaço vetorial real. Essas definições podem parecer estranhas, mas vamos dar exemplos para ver que isso não é tão abstrato quanto parece. Lembramos que uma função é, essencialmente, seu gráfico. Tomemos duas funções $f, g : R \longrightarrow \mathbb{R}$, definidas por $f(x) = 2x$ e $g(x) = x^2$. Vejamos a soma dessas funções representada nos gráficos da Figura 4.

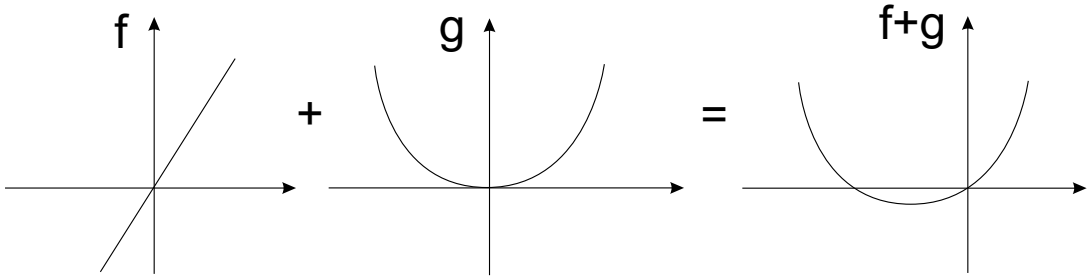


Figura 4: f+g=(f+g)

Observemos que a soma de f mais g produz uma terceira função $(f + g) : \mathbb{R} \longrightarrow \mathbb{R}$, dada por $(f + g)(x) = 2x + x^2$. De volta a Figura 4, o gráfico de f é uma reta passando pela origem, o gráfico de g é uma parábola de boca para cima e vértice na origem, enquanto $(f + g)$ é uma parábola de boca para cima que corta o eixo horizontal em 0 e -2.
Vejamos, agora, a multiplicação de uma função por um número (=escalar). Por exemplo, a função $\left(\frac{1}{4}f\right) : \mathbb{R} \longrightarrow \mathbb{R}$ é dada por $\left(\frac{1}{4}f\right)(x) = \frac{1}{4}f(x)$, isto é $\left(\frac{1}{4}f\right)(x) = \frac{1}{4}2x = \frac{x}{2}$. Veja Figura 5.

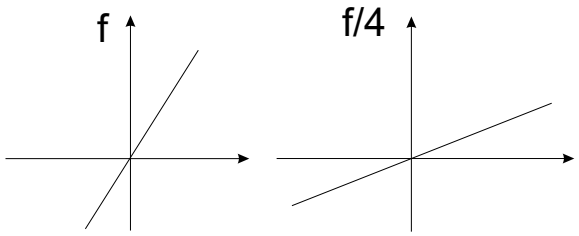


Figura 5: Multiplicação de escalar por função

4 O que pretende o Método dos Mínimos Quadrados

O Método dos Mínimos Quadrados é utilizado quando se modela um fenômeno por uma função f que é uma combinação linear de funções $f = a_1 f_1 + \cdots + a_n f_n$, em que os escalares a_i, para $n \in \{1, \ldots, n\}$ não são conhecidos. A princípio, isso pode parecer muito complicado. Vamos estudar um exemplo bem simples no qual teremos um circuito elétrico dado na Figura 6. Nesse circuito temos uma fonte com força

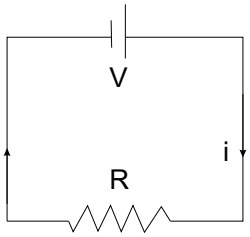


Figura 6: Circuito elétrico

eletromotriz V e um resistor de resistência R. O experimento consiste em variar V e medir a intensidade de corrente i. O modelo matemático assumido é $V = Ri$. Na realidade queremos ver V como função de i, $V(i) = Ri$, sendo R uma constante (=escalar) desconhecida. Suponha que foram feitas várias medições conforme a seguinte tabela.

i_1	i_2	$\cdots$	i_n
V_1	V_2	$\cdots$	V_n

Ao desenharmos os pontos (i_1, V_1), (i_2, V_2),...,(i_n, V_n), notaríamos que eles não estariam alinhados conforme Figura 7.

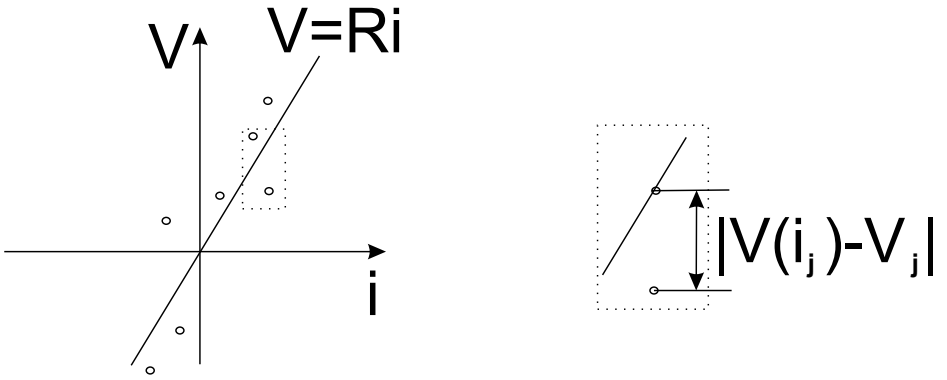


Figura 7: Mínimos Quadrados

O Método consiste em encontrarmos R de modo que a função $V(i) = Ri$ seja um bom modelo para o fenômeno, como na Figura 7. Queremos que a soma

$$(V(i_1) - V_1)^2 + (V(i_2) - V_2)^2 + \cdots + (V(i_n) - V_n)^2 \tag{2}$$

seja mínima. Daí o nome Mínimos Quadrados. De fato, o método consiste em minimizar a soma de quadrados em (2) como na Figura 7, daí alguns preferirem chamá-lo de Quadrados Mínimos, mas, provavelmente, o nome se deve a Legendre: *Méthode des Moindres Carrés*.

5 Elaboração do Algoritmo dos Mínimos Quadrados

Afirmamos na seção 3 que o espaço das funções $\mathcal{F}$ é um espaço vetorial. Queremos, agora, introduzir um produto interno em $\mathcal{F}$ de modo que o quadrado da norma seja da forma (2). Tomemos duas funções $f, g : I \longrightarrow \mathbb{R}$ em $\mathcal{F}$ e $x_1, x_2, \ldots, x_n \in I$ e definamos

$$< f, g >= f(x_1)g(x_1) + f(x_2)g(x_2) + \cdots + f(x_n)g(x_n). \tag{3}$$

Verifica-se facilmente que $< ., . >$ é um produto interno. Além disso

$$||f - g||^2 =< f - g, f - g >= (f(x_1) - g(x_1))^2 + (f(x_n) - g(x_n))^2 + \cdots + (f(x_n) - g(x_n))^2,$$

que representa a essência de (2). Para nos convencermos de tal fato, retornemos ao exemplo do circuito elétrico. Lá $f = V$, isto é, $f(i) = V(i) = Ri$, em que não se conhece R. Já g é uma função tal que $g(i_j) = v_j$. Observemos que, desse modo, $||f - g||^2$ seria exatamente (2). Portanto, R seria o escalar a ser determinado. Veja Figura 8.

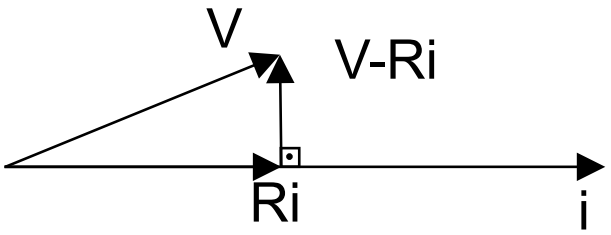
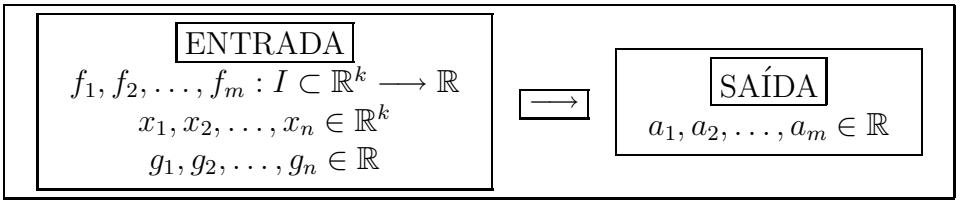


Figura 8: $V = Ri$

.

Agora vamos elaborar o algoritmo dos Mínimos Quadrados. Vejamos, inicialmente, os dados de entrada e saída.



O objetivo do algoritmo, simbolizado por $\boxed{\longrightarrow}$, é minimizar a soma dos quadrados

$$[f(x_1) - g_1]^2 + \cdots + [f(x_n) - g_n)]^2,$$

em que f é definida por $f(x) = a_1 f_1(x) + \cdots + a_m f_m(x)$. Conforme (1), temos que resolver o sistema linear $FA = B$.
A matriz dos coeficientes F tem o elemento da linha i e coluna j definido por

$$f_{ij} =< f_i, f_j >= \sum_{k=1}^{n} f_i(x_k) f_j(x_k), \forall i, j \in \{1, 2, \ldots, m\}.$$

A matriz dos termos independentes é dada por

$$B = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{pmatrix}.$$

Para definirmos B, tomemos uma função g, tal que $g(x_k) = g_k, \forall k \in \{1, 2, \ldots n\}$. De acordo com (1),

$$b_j =< g, f_j >= \sum_{k=1}^{n} g(x_k) f_j(x_k) = \sum_{k=1}^{n} g_k f_j(x_k), \forall j \in \{1, 2, \ldots, m\}.$$

Na prática, em alguns casos, a função g é totalmente dispensável, isto é, basta considerarmos a última soma na definição de b_j.
Finalmente, a matriz das incógnitas é dada por

$$A = \begin{pmatrix} a_1 \\ a_2 \\ \vdots \\ a_m \end{pmatrix}.$$

O algoritmo pode ser sintetizado em

ALGORITMO

$$f_{ij} = f_{ji} = f_i(x_1)f_j(x_1) + \cdots + f_i(x_n)f_j(x_n), \forall i, j \in \{1, 2, \ldots, m\}$$
$$b_j = g_1 f_j(x_1) + \cdots + g_n f_j(x_n), \forall j \in \{1, 2, \ldots, m\}$$
$$\begin{cases} f_{11}a_1 & + \cdots + & f_{1m}a_m = b_1 \\ \vdots & & \vdots \\ f_{m1}a_1 & + \cdots + & f_{mm}a_m = b_m \end{cases}$$

6 Resumo do algoritmo dos Mínimos Quadrados

ENTRADA

$$f_1, f_2, \ldots, f_m : I \subset \mathbb{R}^k \longrightarrow \mathbb{R}$$
$$x_1, x_2, \ldots, x_n \in I$$
$$g_1, g_2, \ldots, g_n \in \mathbb{R}$$

$\boxed{\longrightarrow}$

SAÍDA

$$a_1, a_2, \ldots, a_m \in \mathbb{R}$$

O objetivo do algoritmo, simbolizado por $\boxed{\longrightarrow}$, é minimizar a soma dos quadrados

$$[f(x_1) - g_1]^2 + \cdots + [f(x_n) - g_n)]^2, \tag{4}$$

em que f é definida por $f(x) = a_1 f_1(x) + \cdots + a_m f_m(x)$.

ALGORITMO

$$f_{ij} = f_{ji} = f_i(x_1)f_j(x_1) + \cdots + f_i(x_n)f_j(x_n), \forall i, j \in \{1, 2, \ldots, m\}$$
$$b_j = g_1 f_j(x_1) + \cdots + g_n f_j(x_n), \forall j \in \{1, 2, \ldots, m\}$$
$$\begin{cases} f_{11}a_1 & + \cdots + & f_{1m}a_m = b_1 \\ \vdots & & \vdots \\ f_{m1}a_1 & + \cdots + & f_{mm}a_m = b_m \end{cases}$$

Para entendermos o que o algoritmo resolve, tomemos $k = 1$ na ENTRADA. Temos, então, pares ordenados (x_1, g_1), (x_2, g_2),...,(x_n, g_n) representados no lado esquerda da Figura 9. Pretende-se determinar uma função $f(x) = a_1 f_1(x) + a_2 f_2(x) + \cdots + a_m f_m(x)$, representada no lado direito da Figura 9 que melhor se ajusta aos pares ordenados, no sentido dos mínimos quadrados explicitado em (4). Para se determinar os escalares a_1, a_2,...,a_m, resolve-se o sistema linear explicitado no ALGORITMO.

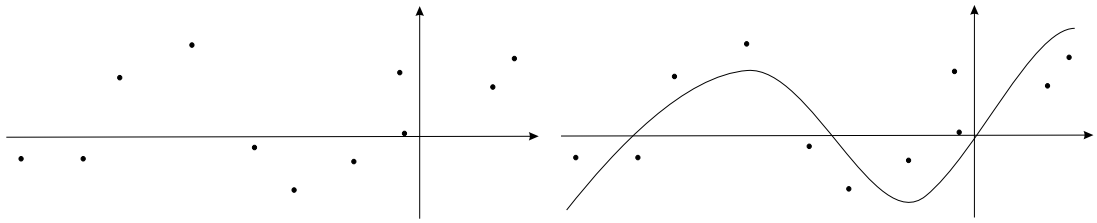


Figura 9: Ajuste de curva usando mínimos quadrados

7 Exemplos

7.1 Exemplo 1

Vamos determinar a reta de equação $y = ax + b$ que melhor se ajusta aos pontos $(x_1, y_1) = (-3, 6)$, $(x_2, y_2) = (0, 4)$, $(x_3, y_3) = (1, 0)$ e $(x_4, y_4) = (2, 2)$ no sentido dos mínimos quadrados, ou seja, tal que $\sum_{i=1}^{4}(y_i - ax_i - b)^2$ seja mínimo. Esses quatro pontos podem ser vistos na Figura 10.

Pondo $f_1(x) = x$ e $f_2(x) = 1$, temos $f(x) = af_1(x) + bf_2(x) = ax + b$. Daí podemos calcular os seguintes coeficientes.

$$\begin{aligned} f_{11} &= f_1(x_1)f_1(x_1) + f_1(x_2)f_1(x_2) + f_1(x_3)f_1(x_3) + f_1(x_4)f_1(x_4) \\ &= (-3)(-3) + (0)(0) + (1)(1) + (2)(2) \\ &= 14 \end{aligned}$$

$$\begin{aligned} f_{12} = f_{21} &= f_1(x_1)f_2(x_2) + f_1(x_2)f_2(x_2) + f_1(x_3)f_2(x_3) + f_1(x_4)f_2(x_4) \\ &= (-3)(1) + (0)(1) + (1)(1) + (2)(1) \\ &= 0 \end{aligned}$$

$$\begin{aligned} f_{22} &= f_2(x_1)f_2(x_1) + f_2(x_2)f_2(x_2) + f_2(x_3)f_2(x_3) + f_2(x_4)f_2(x_4) \\ &= (1)(1) + (1)(1) + (1)(1) + (1)(1) \\ &= 4 \end{aligned}$$

Pondo $g(y) = y$ e $g_i = g(y_i) = y_i$, temos

$$\begin{aligned} b_1 &= g_1 f_1(x_1) + g_2 f_1(x_2) g_3 f_1(x_3) + g_4 f_1(x_4) \\ &= (6)(-3) + (4)(0) + (0)(1) + (2)(2) \\ &= -14 \end{aligned}$$

$$\begin{aligned} b_2 &= g_1 f_2(x_1) + g_2 f_2(x_2) g_3 f_2(x_3) + g_4 f_2(x_4) \\ &= (6)(1) + (4)(1) + (0)(1) + (2)(1) \\ &= 12 \end{aligned}$$

Daí temos o sistema linear

$$\left\{ \begin{array}{l} f_{11}a + f_{12}b = b_1 \\ f_{21}a + f_{22}b = b_2 \end{array} \right. \quad \text{ou} \quad \left\{ \begin{array}{l} 14a = -14 \\ 4b = 12 \end{array} \right. \quad \text{ou} \quad \left\{ \begin{array}{l} a = -1 \\ b = 3 \end{array} \right.$$

Logo a solução é a reta $y = 3 - x$, que pode ser vista na Figura 10.

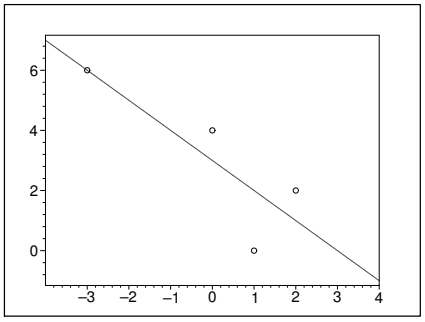


Figura 10: Ajuste de reta

7.2 Exemplo 2

Vamos determinar a parábola de equação $y = a_1x^2 + a_2x + a_3$ que melhor se ajusta aos pontos $(x_1, y_1) = (-2, 0)$, $(x_2, y_2) = (-1, 2)$, $(x_3, y_3) = (1, 2)$ e $(x_4, y_4) = (2, 10)$ (veja Figura 11) no sentido dos mínimos quadrados, ou seja, tal que $\sum_{i=1}^{4}(y_i - a_1x^2 - a_2x - a_3)^2$ seja mínimo. Pondo $f_1(x) = x^2$, $f_2(x) = x$ e $f_3(x) = 1$, temos $f(x) = a_1f_1(x) + a_2f_2(x) + a_3f_3(x) = a_1x^2 + a_2x + a_3$ e podemos calcular os seguintes coeficientes.

$$\begin{aligned}
f_{11} &= f_1(x_1)f_1(x_1) + f_1(x_2)f_1(x_2) + f_1(x_3)f_1(x_3) + f_1(x_4)f_1(x_4)\\
&= (-2)^2(-2)^2 + (-1)^2(-1)^2 + (1)^2(1)^2 + (2)^2(2)^2\\
&= 34
\end{aligned}$$

$$\begin{aligned}
f_{12} = f_{21} &= f_1(x_1)f_2(x_1) + f_1(x_2)f_2(x_2) + f_1(x_3)f_2(x_3) + f_1(x_4)f_2(x_4)\\
&= (-2)^2(-2) + (-1)^2(-1) + (1)^2(1) + (2)^2(2)\\
&= 0
\end{aligned}$$

$$\begin{aligned}
f_{13} = f_{31} &= f_1(x_1)f_3(x_1) + f_1(x_2)f_3(x_2) + f_1(x_3)f_3(x_3) + f_1(x_4)f_3(x_4)\\
&= (-2)^2(1) + (-1)^2(1) + (1)^2(1) + (2)^2(1)\\
&= 10
\end{aligned}$$

$$\begin{aligned}
f_{22} &= f_2(x_1)f_2(x_1) + f_2(x_2)f_2(x_2) + f_2(x_3)f_2(x_3) + f_2(x_4)f_2(x_4)\\
&= (-2)(-2) + (-1)(-1) + (1)(1) + (2)(2)\\
&= 10
\end{aligned}$$

$$\begin{aligned}
f_{23} = f_{32} &= f_2(x_1)f_3(x_1) + f_2(x_2)f_3(x_2) + f_2(x_3)f_3(x_3) + f_2(x_4)f_3(x_4)\\
&= (-2)(1) + (-1)(1) + (1)(1) + (2)(1)\\
&= 0
\end{aligned}$$

$$\begin{aligned}
f_{33} &= f_3(x_1)f_3(x_1) + f_3(x_2)f_3(x_2) + f_3(x_3)f_3(x_3) + f_3(x_4)f_3(x_4)\\
&= (1)(1) + (1)(1) + (1)(1) + (1)(1)\\
&= 4
\end{aligned}$$

Pondo $g(y) = y$ e $g_i = g(y_i) = y_i$, obtemos:

$$\begin{aligned}
b_1 &= g_1f_1(x_1) + g_2f_1(x_2)g_3f_1(x_3) + g_4f_1(x_4)\\
&= (0)(-2)^2 + (2)(-1)^2 + (2)(1)^2 + (10)(2)^4\\
&= 44
\end{aligned}$$

$$\begin{aligned}
b_2 &= g_1f_2(x_1) + g_2f_2(x_2)g_3f_2(x_3) + g_4f_2(x_4)\\
&= (0)(-2) + (2)(-1) + (2)(1) + (10)(2)\\
&= 20
\end{aligned}$$

$$\begin{aligned}
b_3 &= g_1f_3(x_1) + g_2f_3(x_2)g_3f_3(x_3) + g_4f_3(x_4)\\
&= (0)(1) + (2)(1) + (2)(1) + (10)(1)\\
&= 14
\end{aligned}$$

Temos assim o seguinte sistema.

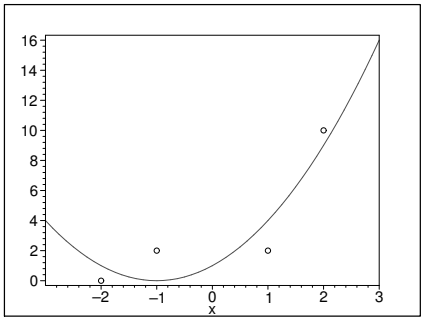


Figura 11: Ajuste de parábola

$$\begin{cases} f_{11}a_1 + f_{12}a_2 + f_{13}a_3 = b_1 \\ f_{21}a_1 + f_{22}a_2 + f_{23}a_3 = b_2 \\ f_{13}a_1 + f_{32}a_2 + f_{33}a_3 = b_3 \end{cases} \quad \text{ou} \quad \begin{cases} 34a_1 + 10a_3 = 44 \\ 10a_2 = 20 \\ 10a_1 + 4a_3 = 14 \end{cases}$$

Portanto, $a_1 = 1$, $a_2 = 2$ e $a_3 = 1$, o que nos dá a parábola $y = x^2 + 2x + 1$, que pode ser vista na Figura 11.

7.3 Exemplo 3

Vamos determinar o círculo de equação $x^2 + y^2 = a_1x + a_2y + a_3$ que melhor se ajusta aos pontos $(x_1, y_1) = (-2, 0)$, $(x_2, y_2) = (0, 2)$, $(x_3, y_3) = (1, -3)$ e $(x_4, y_4) = (3, 1)$ (veja Figura 12) no sentido dos mínimos quadrados, ou seja, tal que $\sum_{i=1}^{4}(x_i^2 + y_i^2 - a_1x^2 - a_2x - a_3)^2$ seja mínimo.

Pondo $f_1(x, y) = x$, $f_2(x, y) = y$ e $f_3(x, y) = 1$, temos

$$f(x, y) = a_1f_1(x, y) + a_2f_2(x, y) + a_3f_3(x, y) = a_1x + a_2y + a_3$$

e podemos calcular os seguintes coeficientes.

$$\begin{aligned} f_{11} &= f_1(x_1, y_1)f_1(x_1, y_1) + f_1(x_2, y_2)f_1(x_2, y_2) + f_1(x_3, y_3)f_1(x_3, y_3) + f_1(x_4, y_4)f_1(x_4, y_4) \\ &= (-2)(-2) + (0)(0) + (1)(1) + (3)(3) \\ &= 14 \end{aligned}$$

$$\begin{aligned} f_{12} = f_{21} &= f_1(x_1, y_1)f_2(x_1, y_1) + f_1(x_2, y_2)f_2(x_2, y_2) + f_1(x_3, y_3)f_2(x_3, y_3) + f_1(x_4, y_4)f_2(x_4, y_4) \\ &= (-2)(0) + (0)(2) + (1)(-3) + (3)(1) \\ &= 0 \end{aligned}$$

$$\begin{aligned} f_{13} = f_{21} &= f_1(x_1, y_1)f_3(x_1, y_1) + f_1(x_2, y_2)f_3(x_2, y_2) + f_1(x_3, y_3)f_3(x_3, y_3) + f_1(x_4, y_4)f_3(x_4, y_4) \\ &= (-2)(1) + (0)(1) + (1)(1) + (3)(1) \\ &= 2 \end{aligned}$$

$$\begin{aligned} f_{22} &= f_2(x_1, y_1)f_2(x_1, y_1) + f_2(x_1, y_2)f_2(x_1, y_2) + f_2(x_1, y_3)f_2(x_1, y_3) + f_2(x_1, y_4)f_2(x_1, y_4) \\ &= (-2)(-2) + (0)(0) + (1)(1) + (3)(3) \\ &= 14 \end{aligned}$$

$$\begin{aligned} f_{23} = f_{32} &= f_2(x_1, y_1)f_3(x_1, y_1) + f_2(x_2, y_2)f_3(x_2, y_2) + f_2(x_3, y_3)f_3(x_3, y_3) + f_2(x_4, y_4)f_3(x_4, y_4) \\ &= (-2)(0) + (0)(2) + (1)(-3) + (3)(1) \\ &= 0 \end{aligned}$$

$$\begin{aligned} f_{33} &= f_3(x_1, y_1)f_3(x_1, y_1) + f_3(x_2, y_2)f_3(x_1, y_2) + f_3(x_3, y_3)f_3(x_1, y_3) + f_3(x_4, y_4)f_3(x_1, y_4) \\ &= (1)(1) + (1)(1) + (1)(1) + (1)(1) \\ &= 4 \end{aligned}$$

Pondo $g(x, y) = x^2 + y^2$ e $g_i = g(x_i, y_i) = x_i^2 + y_i^2$, podemos calcular os termos independentes do sistema.

$$
\begin{aligned}
b_1 &= g_1 f_1(x_1, y_1) + g_2 f_1(x_2, y_2) + g_3 f_1(x_3, y_3) + g_4 f_1(x_4, y_4) \\
&= [(-2)^2 + 0^2](-2) + [0^2 + 2^2](0) + [1^2 + (-3)^2](1) + [3^3 + 1^2](3) \\
&= 32 \\
b_1 &= g_1 f_2(x_1, y_1) + g_2 f_2(x_2, y_2) + g_3 f_2(x_3, y_3) + g_4 f_2(x_4, y_4) \\
&= [(-2)^2 + 0^2](0) + [0^2 + 2^2](2) + [1^2 + (-3)^2](-3) + [3^3 + 1^2](1) \\
&= -12 \\
b_3 1 &= g_1 f_3(x_1, y_1) + g_2 f_3(x_2, y_2) + g_3 f_3(x_3, y_3) + g_4 f_3(x_4, y_4) \\
&= [(-2)^2 + 0^2](1) + [0^2 + 2^2](1) + [1^2 + (-3)^2](1) + [3^3 + 1^2](1) \\
&= 28
\end{aligned}
$$

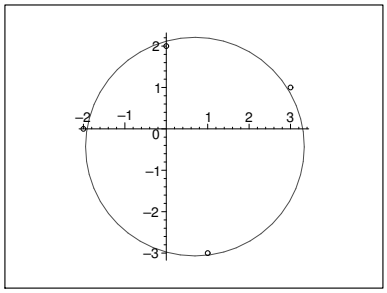


Figura 12: Ajustando um círculo

Resolvendo o seguinte sistema

$$
\begin{cases} f_{11}a_1 + f_{12}a_2 + f_{13}a_3 = b_1 \\ f_{21}a_1 + f_{22}a_2 + f_{23}a_3 = b_2 \\ f_{13}a_1 + f_{32}a_2 + f_{33}a_3 = b_3 \end{cases} \quad \text{ou} \quad \begin{cases} 14a_1 + 2a_3 = 32 \\ 14a_2 = -12 \\ 2a_1 + 4a_3 = 28 \end{cases},
$$

temos $a_1 = \dfrac{18}{13}$, $a_2 = \dfrac{-6}{7}$ e $a_3 = \dfrac{82}{13}$.

Portanto, o círculo procurado tem centro $\left(\dfrac{9}{13}, \dfrac{-3}{7}\right)$ e raio $\dfrac{2\sqrt{14431}}{91} \approx 2,64$, que pode ser visto na Figura 12. Na realidade, no sentido dos mínimos quadrados, estamos ajustando o plano $z = \dfrac{18}{13}x - \dfrac{6}{7}y + \dfrac{82}{13}$ ao parabolóide $z = x^2 + y^2$, conforme Figura 13, na qual as verticais demarcam os pontos $(x_1, y_1) = (-2, 0)$, $(x_2, y_2) = (0, 2)$, $(x_3, y_3) = (1, -3)$ e $(x_4, y_4) = (3, 1)$.

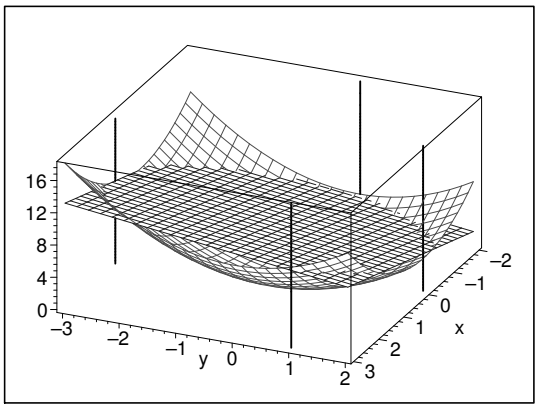


Figura 13: Ajuste de plano a parabolóide

Os exemplos aqui expostos foram copiados de [1] e [2]. Lá o autor usa a técnica de montar um sistema

linear impossível $AX = B$, em que o número de equações é maior que o número de incógnitas. A partir desse sitema ele monta um outro $A^tAX = A^tB$ que é o mesmo que encontramos nos nossos exemplos. De fato esse método é equivalente ao exposto no nosso Algoritmo cujas idéias podem ser encontradas em [4] e mais sucintamente em [3].

7.4 Exemplo 4

Até agora, dado um conjunto de pontos, estávamos ajustando uma curva no sentido dos mínimos quadrados. Vamos continuar fazendo isso, mas mudando um pouco o enfoque. Suponhamos que queiramos aproximar uma função trigonométrica por um polinômio. Por exemplo, aproximar $\text{sen}(x)$ para $x \in [0, \pi/2]$ por um polinômio do terceiro grau, $c_1x+c_2x^3$. Tomemos $(x_1, y_1) = (0, 0)$, $(x_2, y_2) = (\pi/6, 1/2)$, $(x_3, y_3) = (\pi/4, \sqrt{2}/2)$, $(x_4, y_4) = (\pi/3, \sqrt{3}/2)$ e $(x_5, y_5) = (\pi/2, 1)$. Pondo $f_1(x) = x$, $f_2(x) = x^3$, com $f(x) = c_1f_1(x) + c_2f_2(x) = c_1x + c_2x^3$ e $g(x) = \text{sen}(x)$, estamos querendo minimizar $||f - g||$, isto é, $\text{sen}(x) \approx c_1x + c_2x^3$, para $x \in [0, \pi/2]$. De posse desses dados, podemos construir a seguite tabela.

i	1	2	3	4	5
x_i	0 $0,0$	$\pi/6$ $0,52$	$\pi/4$ $0,78$	$\pi/3$ $1,05$	$\pi/2$ $1,57$
$g_i = g(x_i) = \text{sen}(x_i)$	0 $0,0$	$1/2$ $0,5$	$\sqrt{2}/2$ $0,71$	$\sqrt{3}/2$ $0,87$	1 $1,0$
$f_1(x_i) = x_i$	$0,0$	$0,52$	$0,78$	$1,05$	$1,57$
$f_2(x_i) = x_i^3$	$0,03$ $0,0$	$0,52^3$ $0,14$	$0,78^3$ $0,47$	$1,05^3$ $1,16$	$1,57^3$ $3,87$

De acordo com essa tabela, podemos calcular os seguintes coeficientes.

$$\begin{aligned} f_{11} &= f_1(x_1)f_1(x_1) + f_1(x_2)f_1(x_2) + f_1(x_3)f_1(x_3) + f_1(x_4)f_1(x_4) + f_1(x_5)f_1(x_5) \\ &= 0,0^2 + 0,52^2 + 0,78^2 + 1,05^2 + 1,57^2 \\ &= 4,44 \end{aligned}$$

$$\begin{aligned} f_{12} = f_{21} &= f_1(x_1)f_2(x_1) + f_1(x_2)f_2(x_2) + f_1(x_3)f_2(x_3) + f_1(x_4)f_2(x_4) + f_1(x_5)f_2(x_5) \\ &= (0,0)(0,0) + (0,52)(0,14) + (0,78)(0,47) + (1,05)(1,16) + (1,57)(3,87) \\ &= 7,73 \end{aligned}$$

$$\begin{aligned} f_{22} &= f_2(x_1)f_2(x_1) + f_2(x_2)f_2(x_2) + f_2(x_3)f_2(x_3) + f_2(x_4)f_2(x_4) + f_2(x_5)f_2(x_5) \\ &= 0,0^2 + 0,14^2 + 0,47^2 + 1,16^2 + 3,87^2 \\ &= 16,56 \end{aligned}$$

$$\begin{aligned} b_1 &= g_1f_1(x_1) + g_2f_1(x_2) + g_3f_1(x_3) + g_4f_1(x_4) + g_5f_1(x_5) \\ &= (0,0)(0,0) + (0,5)(0,52) + (0,71)(0,78) + (0,87)(1,05) + (1,0)(1,57) \\ &= 3,3 \end{aligned}$$

$$\begin{aligned} b_2 &= g_1f_2(x_1) + g_2f_2(x_2) + g_3f_2(x_3) + g_4f_2(x_4) + g_5f_2(x_5) \\ &= (0,0)(0,0) + (0,5)(0,14) + (0,71)(0,47) + (0,87)(1,16) + (1,0)(3,87) \\ &= 5,28 \end{aligned}$$

Resolvendo o sistema

$$\left\{ \begin{array}{l} f_{11}a + f_{12}b = b_1 \\ f_{21}a + f_{22}b = b_2 \end{array} \right. \quad \text{ou} \quad \left\{ \begin{array}{rcrcl} 4,44c_1 & + & 7,73c_2 & = & 3,3 \\ 7,73c_1 & + & 16,56c_2 & = & 5,28 \end{array} \right. ,$$

obtemos $c_1 = 1$ e $c_2 = -0,15$ e, portanto, $\text{sen}(x) \approx t - 0,15t^3$, cujos gráficos podem ser vistos na Figura 14.

8 Mínimos Quadrados no Computador

Utilizaremos os mesmos exemplos da seção 7.

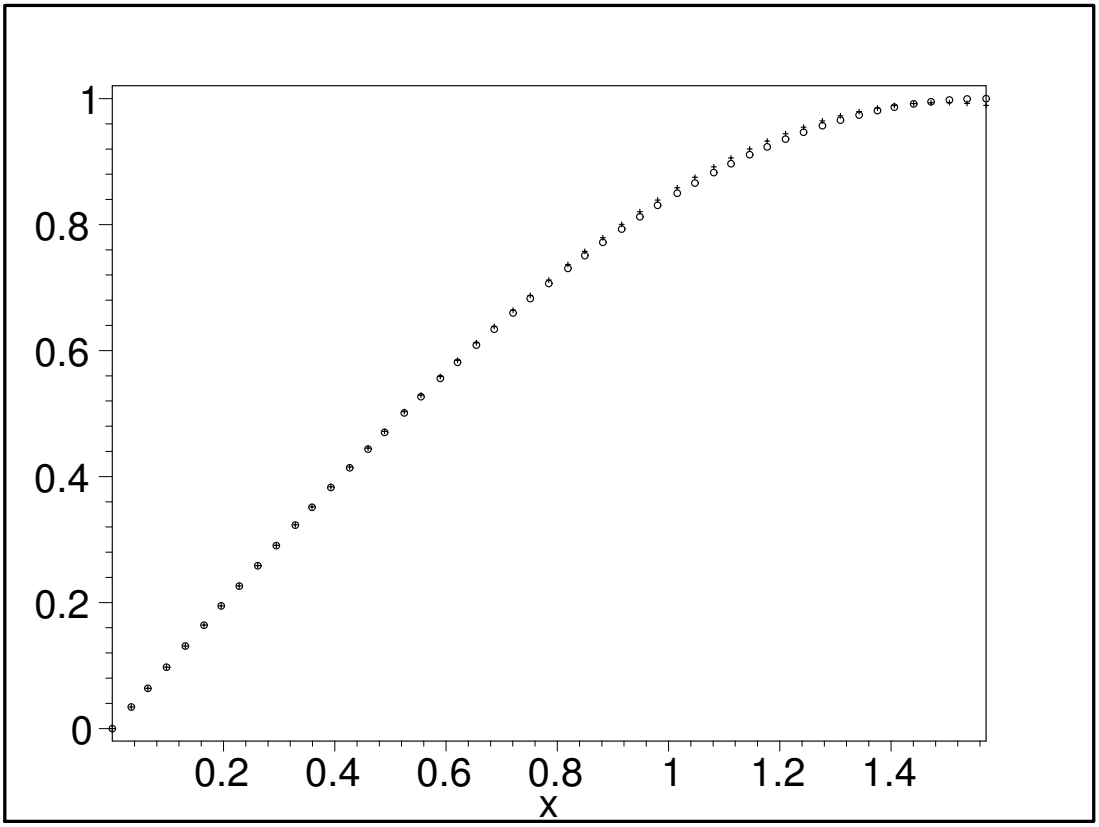


Figura 14: sen (símbolo o) - polinômio (símbolo +)

8.1 Usando o wxMaxima

Para usarmos o wxMaxima devemos acessar http://andrejv.github.com/wxmaxima e baixar o programa de instalação. A instalação é muito rápida e simples.
O wxMaxima é uma interface gráfica muito amigável para se utilizar um sistema de computacional simbólico chamado Maxima. Um sistema de computacional simbólico ou um sistema algébrico computacional (em inglês: computer algebra system) é um programa de computador que facilita o cálculo na matemática simbólica. Matemática simbólica é a utilização de computadores para manipular equações matemáticas e expressões simbólica, em oposição à mera manipulação de aproximações a quantidades numéricas específicas representadas por aqueles símbolos. Um tal sistema pode ser usado para integração ou diferenciação, substituição de uma expressão numa outra, simplificação de uma expressão, etc. A descrição do wxMaxima ora exposta foi retirada da Wikipédia. Não utilizaremos aqui a potencialidade da matemática simbólica do wxMaxima. Convidamos o leitor a explorar tal potencialidade, uma vez que o programa é muito amigável. Utilizaremos apenas comandos para resolver Mínimos Quadrados e para desenhar gráficos. Citamos outros dois programas populares para matemática simbólica: Maple e Mathematica. Nossa escolha pelo wsMaxima se deve, primordialmente , pelo fato pela sua gratuidade.

Utilizaremos o wxMaxima para resolver os exemplos da seção 7.
Os símbolos (%i) e (%o), que apareceram em seguida, significam, respectivamente, entrada (*input*) e saída (*output*). Ao se digitar uma linha de comando, devemos pressionar as teclas [*shit*] seguida de [*enter*].

```
(%i1)  /* zerar conteúdo das variáveis */ reset();kill(all);
```

(%o1) $[tr-unique, lispdisp, features, labels, \_]$

(%o0) *done*

Exemplo da seção 7.1: Determinar a reta $y = ax + b$ que melhor se ajusta aos pontos $(-3, 6)$, $(0, 4)$, $(1, 0)$ e $(2, 2)$.

Resolução do Exemplo da secção 7.1:

Passo 1: Com os pontos construir a matriz

```
(%i1)  M1:matrix([-3,6],[0,4],[1,0],[2,2]);
```

(%o1) $$\begin{pmatrix} -3 & 6 \\ 0 & 4 \\ 1 & 0 \\ 2 & 2 \end{pmatrix}$$

Passo 2: Carregar o pacote "lsquares".

```
(%i2)  load("lsquares");
```

; *Compiler warnings for "C : /PROGRA 2/MAXIMA 1.0/share/maxima/5.25.0/share/lbfgs/lb1.lisp" :*
; *In LB1 : Unused lexical variable I*
(%o2) *C : /PROGRA 2/MAXIMA 1.0/share/maxima/5.25.0/share/contrib/lsquares.mac*

Passo 3: Executar "lsquare_estimates"para determinar os parâmetros a e b.

```
(%i3)  lsquares_estimates(M1,[x,y],y=a*x+b,[a,b]);
```

(%o3) $[[a = -1, b = 3]]$

Exemplo da seção 7.2: Determinar a parábola $y = ax^2 + bx + c$ que melhor se ajusta aos pontos $(-2, 0)$, $(-1, 2)$, $(1, 2)$ e $(2, 10)$.

Resolução do Exemplo da seção 7.2:

Passo 1: Com os pontos construir a matriz

```
(%i4)  M2: matrix([-2,0],[-1,2],[1,2],[2,10]);
```

(%o4) $$\begin{pmatrix} -2 & 0 \\ -1 & 2 \\ 1 & 2 \\ 2 & 10 \end{pmatrix}$$

Passo 2: Como "lsquares"já foi carregado no Exemplo 1, não será carregado.

Passo 3: Executar "lsquare_estimates"para determinar os parâmetros a, b e c.

```
(%i5)  lsquares_estimates(M2,[x,y],y=a*x^2+b*x+c,[a,b,c]);
```

(%o5) $[[a = 1, b = 2, c = 1]]$

Exemplo da seção 7.3: Determinar ao círculo $x^2 + y^2 = ax + by + c$ que melhor se ajusta aos pontos $(-2, 0)$, $(0, 2)$, $(1, -3)$ e $(3, 1)$.

Resolução do Exemplo da seção 7.3:

Passo 1: Com os pontos construir a matriz

```
(%i6)  M3: matrix(
         [-2,0],
         [0,2],
         [1,-3],
         [3,1]
       );
```

$$(\%o6) \quad \begin{pmatrix} -2 & 0 \\ 0 & 2 \\ 1 & -3 \\ 3 & 1 \end{pmatrix}$$

Passo 2: Como "lsquares"já foi carregado no Exemplo 1, não será carregado.

Passo 3: Executar "lsquare_estimates"para determinar os parâmetros a, b e c.

```
(%i7)  lsquares_estimates(M3,[x,y],x^2+y^2=a*x+b*y+c,[a,b,c]);
```

$$(\%o7) \quad [[a = \frac{18}{13}, b = -\frac{6}{7}, c = \frac{82}{13}]]$$

Gráficos: Para desenhar curvas planas basta executar "wxplot2d".

```
(%i8)  /* Gráfico do Exemplo 1 */ wxplot2d(3-x,[x,-4,4]);
```

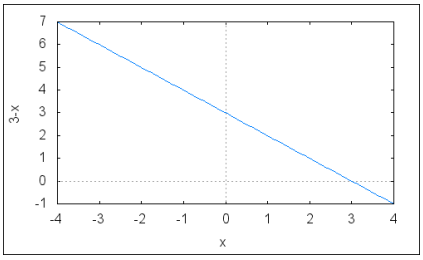


Figura 15: Gráfico do Exemplo 1

```
(%i9)  /* Gráfico do Exemplo 2 */ wxplot2d(x^2+2*x+1,[x,-3,3]);
```

```
(%i2)  /* Gráfico do Exemplo 3 */ wxplot2d([parametric,9/13+2.64*cos(t),-3/7+2.64*sin(t),
       [t,-%pi,%pi]],[gnuplot_preamble, "set size ratio -1"]);
```

Estes gráficos podem ser vistos nas figuras 15, 16 e 17.
Vamos, agora, acrescentar os pontos dados aos gráficos:

```
(%i11) wxplot2d([3-x,[discrete,[[-3,6],[0,4],[1,0],[2,2]]],[x,-4,4],
       [style,[lines,1,1], [points,1,2]],[legend,"cor da reta","cor do ponto"]);
```

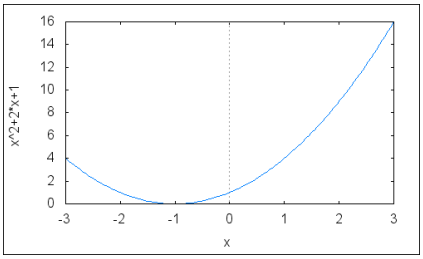


Figura 16: Gráfico do Exemplo 2

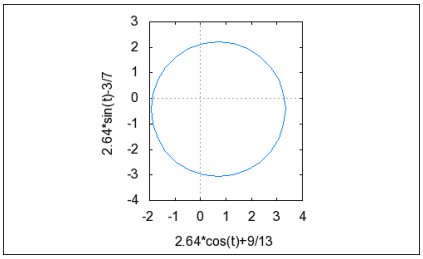


Figura 17: Gráfico do Exemplo 3

```
(%i12) wxplot2d([x^2+2*x+1,[discrete,[[-1,2],[-1,2],[1,2],[2,10]]]],[x,-3,3],
       [style,[lines,1,1], [points,1,2]],[legend,"cor da parábola","cor do ponto"]);

(%i13) wxplot2d([[parametric,9/13+2.64*cos(t),-3/7+2.64*sin(t),[t,-%pi,%pi]],
       [discrete,[[-2,0],[0,2],[1,-3],[3,1]]]],[x,-4,4],[style,[lines,1,1],
       [points,1,2]],[legend,"cor do círculo","cor do ponto"],
       [gnuplot_preamble, "set size ratio -1"]);
```

Estes gráficos podem ser vistos nas figuras 18, 19 e 20.

8.2 Usando o Scilab

Para usarmos o Scilab devemos acessar http://www.scilab.org/ e baixar o programa de instalação. A instalação também é muito rápida e simples.
O Scilab é um software científico para computação numérica semelhante ao Matlab que fornece um poderoso ambiente computacional aberto para aplicações científicas (informações retiradas da Wikipédia). A grande vantagem do Scilab é a sua gratuidade, além de sua eficiência em métodos numéricos e sua utilização industrial.

A fim de ilustrar o uso do Scilab, faremos um Exemplo de seção 7.

Exemplo da seção 7.2: Vamos determinar a parábola de equação $y = a_1x^2 + a_2x + a_3$ que melhor se ajusta aos pontos $(x_1, y_1) = (-2, 0)$, $(x_2, y_2) = (-1, 2)$, $(x_3, y_3) = (1, 2)$ e $(x_4, y_4) = (2, 10)$ (veja Figura 11) no sentido dos mínimos quadrados, ou seja, tal que $\sum_{i=1}^{4}(y_i - a_1x^2 - a_2x - a_3)^2$ seja mínimo.
Pondo $f_1(x) = x^2$, $f_2(x) = x$ e $f_3(x) = 1$, temos $f(x) = a_1f_1(x) + a_2f_2(x) + a_3f_3(x)$ e podemos calcular os seguintes coeficientes f_{ij} e gerar a seguinte matriz.

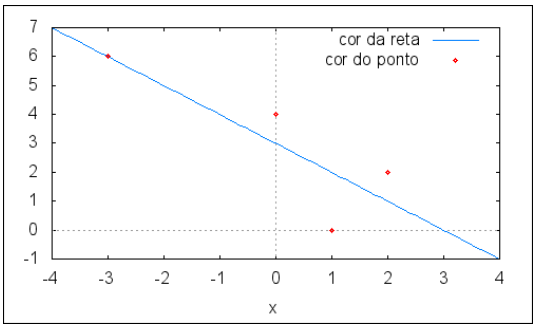


Figura 18: Ajuste de reta pelo Maxima

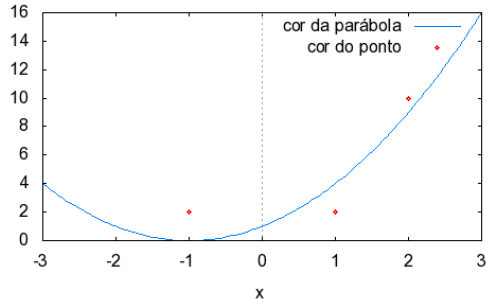


Figura 19: Ajuste de parábola pelo Maxima

$$A = \begin{pmatrix} f_{11} & f_{12} & f_{13} \\ f_{21} & f_{22} & f_{23} \\ f_{13} & f_{32} & f_{33} \end{pmatrix} = \begin{pmatrix} 4 & -2 & 1 \\ 1 & -1 & 1 \\ 4 & 2 & 1 \end{pmatrix}$$

Passo 1: Introduzir a matrix A no Scilab.

```
-->A=[4 -2 1;1 -1 1;1 1 1;4 2 1]
 A  =

    4.  - 2.    1.
    1.  - 1.    1.
    1.    1.    1.
    4.    2.    1.
```

No passo seguinte devemos criar a matriz

$$B = \begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{pmatrix} = \begin{pmatrix} 0 \\ 2 \\ 2 \\ 10 \end{pmatrix}.$$

Passo 2: Introduzir a matrix B no Scilab.

```
-->B=[0;2;2;10]
 B  =
```

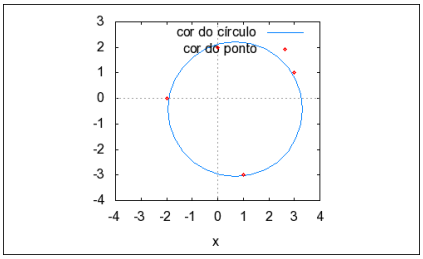


Figura 20: Ajuste de círculo

```
    0.
    2.
    2.
    10.
```

Passo 3: Usar o comando **lsq** para determinar os coeficientes a_1, a_2 e a_3.

```
-->X=lsq(A,B)
 X  =

    1.
    2.
    1.
```

Para desenharmos a parábola $y = x^2 + 2x + 1$, devemos inicialmente introduzirmos uma sequência de pontos no eixo x.

```
-->x=-3:0.2:3
 x  =


         column  1 to 22

  - 3.  - 2.8  - 2.6  - 2.4  - 2.2  - 2.  - 1.8  - 1.6  - 1.4  - 1.2  - 1.  - 0.8  - 0.6  -

         column 23 to 31

    1.4    1.6    1.8    2.    2.2    2.4    2.6    2.8    3.
```

e desenhar o gráfico utilizando o comando **plot**.

```
-->plot(x^2+2*x+1)
```

O Scilab abrirá uma janela com a Figura 21.

Observação 1: Os cáculos das matrizezs A e B poderiam ser feitas automaticamente, pois o Scilab possui um ambiente de programação bem amigável, mas que não será aqui explorado.
Observação 2: A idéia da utilização das matrizes A e B para solucionar o Problema dos Mínimos Quadrados pode ser vista em [1] e [2]. Tal idéia é, simplesmente, resolver o sistema $A^tAX = A^tB$.

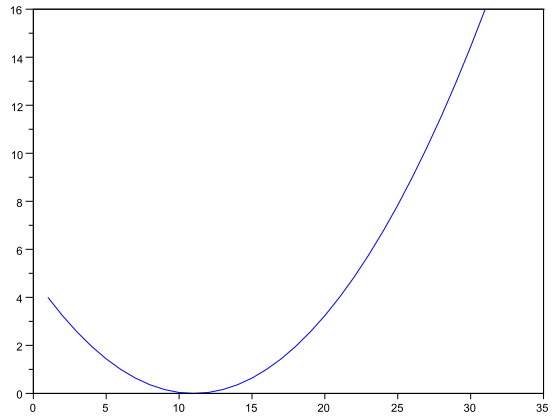


Figura 21: Parábola do Exemplo 2 feita pelo Scilab

8.3 Usando C

A linguagem C foi criada em 1972 por Ritchie & Thompson e é bem menos amigável que os ambientes wxMaxima e Scilab apresentados nas seções 8.1 e 8.2, respectivamente. O wxMaxima e o Scilab possuem ambientes de programação bastante amigáveis, mas diferem do C, pois este é uma linguagem compilada enquanto aqueles são interpretados. O que significa isso? Numa linguagem interpretada cada comando é interpretado pelo programa e, então, executado individualmente. Numa linguagem compilada, é gerado um código em linguagem de máquina que executa o programa integralmente o que lhe confere maior celeridade. Além disso, os programa em C tendem a ser bem compactos e de rápida execução.
Para trabalharmos com o C podemos usar o programa DevC++ , que pode ser baixado em http://www.bloodshed.net gratuitamente. Vários outros programas nos permitem trabalhar com o C. Escolhemos o DevC++, que além de sua gratuidade é sugerido por [7] que nos parece uma ótima referência para o estudo de C.

A fim de ilustrar o uso do C, faremos um Exemplo de seção 7.

Exemplo da seção 7.2: Vamos determinar a parábola de equação $y = a_1x^2 + a_2x + a_3$ que melhor se ajusta aos pontos $(x_1, y_1) = (-2, 0)$, $(x_2, y_2) = (-1, 2)$, $(x_3, y_3) = (1, 2)$ e $(x_4, y_4) = (2, 10)$ (veja Figura 11) no sentido dos mínimos quadrados, ou seja, tal que $\sum_{i=1}^{4}(y_i - a_1x^2 - a_2x - a_3)^2$ seja mínimo.
Da seção 7.2, sabemos que devemos criar três funções: $f_1(x) = x^2$, $f_2(x) = x$ e $f_3(x) = 1$. Para criar tais funções em C, podemos fazer o seguinte código.

```
/*********************************************************/
#include<stdio.h> /* controle de entrada e saída */
#include<stdlib.h> /* biblioteca padrão */

float F(int i,float x) /* funções f1(x), f2(x),...,fm(x)  */
{
   switch(i)
   {
     case 0:
       return(pow(x,2)); /* F(0,x)=x^2 */
```

```
      break;
    case 1:
      return(x);  /* F(1,x)=x */
      break;
    case 2:
      return(1); /* F(2,x)=1 */
      break;
  }
}

int main()
{
  printf("%f\n",F(0,3)); /* escreve F(0,3) */
  printf("%f\n",F(1,2016)); /* escreve F(2,2016) */
  printf("%f\n",F(2,2016)); /* escreve F(2,2016) */
  system("PAUSE");
}
/********************************************************/
```

A função F(i,x) definida em float F(int i,float x) corresponde a função $f_{i+1}(x)$, isto é, $f_1(x) = F(0,x)$, $f_2(x) = F(1,x)$ e $f_3(x) = F(2,x)$. O motivo de começarmos com $i = 0$ em F se deve ao fato de que a primeira coordenada de um vetor em C tem índice 0 e não 1 como é o usual. Como os livros didáticos de C usam como primeiro índice o 0, preferimos manter esta estratégia que é também padrão entre os programadores. Portanto a saída desse programa é

```
9.000000
2016.000000
1.000000
Precione qualquer tecla para continuar. . .
```

Para calcularmos os coeficientes f_{ij} da seção 7.2, temos o seguinte código

```
/************************************************************/
#include <stdio.h> /* entrada e saída */
#include<stdlib.h> /* biblioteca padrão */

#define m 3 /* número de coeficiente a serem determinados */
#define n 4 /* número de pontos fornecidos */

float F(int i,float x) /* funções f1(x), f2(x),...,fm(x)  */
{
   switch(i)
   {
     case 0:
       return(pow(x,2)); /* F(0,x)=x^2 */
       break;
     case 1:
       return(x);  /* F(1,x)=x */
       break;
     case 2:
```

```
       return(1); /* F(2,x)=1 */
       break;
   }
}

void fij(float f[m][m],float x[n]) /* retorna os coeficientes fij */
{
  int i,j,k;
  float soma;
  for(i=0;i<=m-1;i++)
  {
     for(j=i;j<=m-1;j++)
     {
       soma=0;
       for(k=0;k<=n-1;k++) /* fij=fi(x1)fj(x1)+...+fi(xn)fj(xn) */
       {
         soma=soma+F(i,x[k])*F(j,x[k]);
       }
       f[i][j]=soma;
       f[j][i]=f[i][j];
     }
  }
}

int main()
{
  float x[n]={-2,-1,1,2}, /* x1=-2, x2=-1, x3=1, x4=2 */
        y[n]={0,2,2,10}, /* y1=0, y2=2, y3=2, y4=10 */
        f[m][m],b[m];
  int i;
  fij(f,x); /* calcula a matriz f */
  printf("%f\n",f[0][2]);
  system("PAUSE");
}
/*****************************************************************/
```

A saída é

```
10.000000
Precione qualquer tecla para continuar. . .
```

o que está coerente com a seção 7.2, uma vez que $f_{13} = f[0][2] = 10$. Lembremos que os ídices de vetores e matrizes começam do 0 e, portanto, $f_{ij} = f[i-1][j-1]$.

Vamos, agora, calcular os termos independentes b_1, b_2 e b_3.

```
/*************************************************************/
#include <stdio.h>
#include<stdlib.h>

#define m 3 /* número de coeficiente a serem determinados */
```

```
#define n 4 /* número de pontos fornecidos */

float F(int i,float x) /* funções f1(x), f2(x),...,fm(x)  */
{
   switch(i)
   {
     case 0:
       return(pow(x,2)); /* F(0,x)=x^2 */
       break;
     case 1:
       return(x);  /* F(1,x)=x */
       break;
     case 2:
       return(1); /* F(2,x)=1 */
       break;
   }
}

void bi(float b[m],float x[n],float y[n]) /* retorna os termos independentes bi */
{
  int i,k;
  float soma;
  for(i=0;i<=m-1;i++)
  {
     soma=0;
     for(k=0;k<=n-1;k++) /* y1fi(x1)+...+ynf(xn) */
     {
       soma=soma+y[k]*F(i,x[k]);
     }
     b[i]=soma;
  }
}

int main()
{
  float x[n]={-2,-1,1,2},
        y[n]={0,2,2,10},
        f[m][m],b[m];
  int i;
  bi(b,x,y);
  printf("%f\n",b[2]);
  system("PAUSE");
}
/*************************************************************/
```

A saída é

```
14.000000
Precione qualquer tecla para continuar. . .
```

o que está correto, uma vez que $b_3 = b[2] = 14$, que está coerente com a seção 7.2.

Finalmente, vamos exibir o programa completo introduzindo uma rotina para resolver um sistema linear retirada de [7].

```
/*************************************************************************
Determinação da parábola y=a_1x^2+a_2x+a_3 que melhor se ajusta aos pontos
(-2,0), (-1,2), (1,2) e (2,10),
no sentido dos MÍNIMOS QUADRADOS.
output: a_1, a_2 e a_3
*************************************************************************/

#include <stdio.h> /* entrada e saída */
#include<stdlib.h> /* biblioteca padrão */

#define m 3 /* número de coeficiente a serem determinados */
#define n 4 /* número de pontos fornecidos */

float F(int i,float x) /* funções f1(x), f2(x),...,fm(x)  */
{
   switch(i)
   {
     case 0:
       return(pow(x,2)); /* F(0,x)=x^2 */
       break;
     case 1:
       return(x);  /* F(1,x)=x */
       break;
     case 2:
       return(1); /* F(2,x)=1 */
       break;
   }
}

void fij(float f[m][m],float x[n]) /* retorna os coeficientes fij */
{
  int i,j,k;
  float soma;
  for(i=0;i<=m-1;i++)
  {
     for(j=i;j<=m-1;j++)
     {
       soma=0;
       for(k=0;k<=n-1;k++)
       {
         soma=soma+F(i,x[k])*F(j,x[k]);
       }
       f[i][j]=soma;
       f[j][i]=f[i][j];
     }
  }
```

```
}

void bi(float b[m],float x[n],float y[n]) /* retorna os termos independentes bi */
{
  int i,k;
  float soma;
  for(i=0;i<=m-1;i++)
  {
     soma=0;
     for(k=0;k<=n-1;k++)
     {
       soma=soma+y[k]*F(i,x[k]);
     }
     b[i]=soma;
  }
}

void solve(float[m][m],float[m]); /* rerolve o sistema linear */

int main()
{
  float x[n]={-2,-1,1,2}, /* x1=-2, x2=-1, x3=1, x4=2 */
        y[n]={0,2,2,10}, /* y1=0, y2=2, y3=2, y4=10 */
        f[m][m],b[m];
  int i;
  fij(f,x);
  bi(b,x,y);
  solve(f,b);
  for(i=0;i<m;i++)
    printf("%f\n",b[i]);
  system("PAUSE");
}

void solve(float f[m][m],float b[m])
{
register int i,j,k;
float w[m];

for (k=0; k<m; k++) {
for (i=0; i<k; i++) {
w[i] = f[k][i];
for (j=0; j<i; j++)
w[i] -= w[j] * f[i][j];
        f[k][i] = w[i] * f[i][i];
}
for (i=0; i<k; i++)
f[k][k] -= w[i] * f[k][i];
f[k][k] = 1.0 /f[k][k];
}
```

```
for (k=1; k<m; k++)
for (i=0; i<k; i++)
b[k] -= f[k][i] * b[i];
for (k=m-1; k>=0; k--) {
b[k] *= f[k][k];
for (i=k+1; i<m; i++)
b[k] -= f[i][k] * b[i];
}
}
/*************************************************************************/
```

A saída é

```
1.000000
2.000000
1.000000
Precione qualquer tecla para continuar. . .
```

o que está coerente, uma vez que, pela seção 7.2 $a_1 = 1$, $a_2 = 2$ e $a_3 = 1$.
Como não é nosso objetivo a solução de sistema linear, não será dada ênfase a esse tópico. Aliás, a rotina de [7] foi modificada, propositalmente, retirando-se a alocação dinâmica [8] a fim de facilitar a modificação do código pelo leitor não familiarizado com programaçãm em C. A íntegra da rotina [7] pode ser encontrada em 9.1.

Convidamos o leitor a modificar o último código de modo a proceder o ajuste da reta da seção 7.1. Já uma tarefa mais desafiadora é modificar o código a fim proceder ao ajuste do círculo conforme seção 7.3. Deixamos a solução deste último problema na seção 9.2. Ao leitor que não aceitar o desafio, sugerimos que tente entender a coerência das modificações realizadas an seção 9.2

9 Apêndice

9.1 Código em C para resolução de sistemas $AX = B$, em que A é simétrica

O seguinte código foi retirado de [7]. Acrescentamos 7 linhas ao código original, indicadas por pelo símbolo $< --$, para facilitar o entendimento da execução do programa.

```
/***************************
* Modified Cholesky Method *
* (C) 1993 by K. Ikeda     *
***************************/
#include <stdio.h>
#include<stdlib.h> /* <-- */

#define A(i,j) a[(i)*(i+1)/2+j]

void decomp(double a[], int n)
{
  register int i,j,k;
  double *w = (double *)malloc(n*sizeof(double));
```

```
  for (k=0; k<n; k++) {
    for (i=0; i<k; i++) {
      w[i] = A(k,i);
      for (j=0; j<i; j++)
      w[i] -= w[j] * A(i,j);
      A(k,i) = w[i] * A(i,i);
    }
    for (i=0; i<k; i++)
      A(k,k) -= w[i] * A(k,i);
    A(k,k) = 1.0 / A(k,k);
  }
  free((void *)w);
}

void solve(double a[], double b[], int n)
{
  register int i,k;

  for (k=1; k<n; k++)
    for (i=0; i<k; i++)
      b[k] -= A(k,i) * b[i];
    for (k=n-1; k>=0; k--) {
      b[k] *= A(k,k);
      for (i=k+1; i<n; i++)
        b[k] -= A(i,k) * b[i];
    }
}

int main()
{
  double *a, *b;
  int i,j,n;

  printf("tamanho do sistema="); /* <-- */
  scanf("%d", &n);
  a = (double *)malloc(n*(n+1)/2*sizeof(double));
  b = (double *)malloc(n*sizeof(double));
  printf("entre com a matriz do sistema\n"); /* <-- */
  for (i=0; i<n; i++)
    for (j=0; j<=i; j++) {
      printf("f%d%d=f%d%d=",i+1,j+1,j+1,i+1); /* <-- */
      scanf("%lf", &A(i,j));
    }
  printf("termos independentes\n"); /* <-- */
  for (i=0; i<n; i++) {
    printf("b%d=",i+1);  /* <-- */
    scanf("%lf", &b[i]);
  }
  decomp(a,n);
```

```
  solve(a,b,n);
  for (i=0; i<n; i++)
    printf("%lf\n", b[i]);
  free((void *)a); free((void *)b);
  system("PAUSE"); /* <-- */
}
```

Ao executarmos este programa,temos:

```
tamanho do sistema=3
f11=f11=1
f21=f12=0
f22=f22=2
f31=f13=0
f32=f23=0
f33=f33=3
termos independentes
b1=1
b2=1
b3=1
1.000000
0.500000
0.333333
Precione qualquer tecla para continuar. . .
```

Fornecemos o seguinte sistema

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 3 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix},$$

cuja solução é

```
1.000000
0.500000
0.333333.
```

9.2 Ajuste de círculo em C

No seguinte código foi necessário acrescentarmos uma função G que faz o papel de g na seção 7.3.

```
/*********************************************************************
Determinação do círculo x^2+y^2=a_1x^2+a_2x+a_3 que melhor se ajusta aos pontos
(-2,0), (0,2), (1,-3) e (3,1)
no sentido dos MÍNIMOS QUADRADOS.
output: a_1, a_2 e a_3
*********************************************************************/

#include <stdio.h> /* entrada e saída */
#include<stdlib.h> /* biblioteca padrão */

#define m 3 /* número de coeficiente a serem determinados */
#define n 4 /* número de pontos fornecidos */
```

```
float F(int i,float x,float y) /* funções f1(x), f2(x),...,fm(x)  */
{
   switch(i)
   {
     case 0:
       return(x); /* F(0,x,y)=x */
       break;
     case 1:
       return(y);  /* F(1,x,y)=y */
       break;
     case 2:
       return(1); /* F(2,x,y)=1 */
       break;
   }
}

void fij(float f[m][m],float x[n],float y[n]) /* retorna os coeficientes fij */
{
  int i,j,k;
  float soma;
  for(i=0;i<=m-1;i++)
  {
     for(j=i;j<=m-1;j++)
     {
       soma=0;
       for(k=0;k<=n-1;k++)
       {
         soma=soma+F(i,x[k],y[k])*F(j,x[k],y[k]);
       }
       f[i][j]=soma;
       f[j][i]=f[i][j];
     }
  }
}

float G(float x,float y) /* função g dada */
{
   return(x*x+y*y); /* G(x,y)=x^2+y^2 */
}

void bi(float b[m],float x[n],float y[n]) /* retorna os termos independentes bi */
{
  int i,k;
  float soma;
  for(i=0;i<=m-1;i++)
  {
     soma=0;
     for(k=0;k<=n-1;k++)
```

```
    {
      soma=soma+G(x[k],y[k])*F(i,x[k],y[k]);
    }
    b[i]=soma;
  }
}

void solve(float[m][m],float[m]); /* rerolve o sistema linear */

int main()
{
  float x[n]={-2,0,1,3}, /* x1=-2, x2=0, x3=1, x4=3 */
        y[n]={0,2,-3,1}, /* y1=0, y2=2, y3=-3, y4=1 */
        f[m][m],b[m];
  int i;
  fij(f,x,y);
  bi(b,x,y);
  solve(f,b);
  for(i=0;i<m;i++)
    printf("%f\n",b[i]);
  system("PAUSE");
}

void solve(float f[m][m],float b[m])
{
register int i,j,k;
float w[m];

for (k=0; k<m; k++) {
for (i=0; i<k; i++) {
w[i] = f[k][i];
for (j=0; j<i; j++)
w[i] -= w[j] * f[i][j];
        f[k][i] = w[i] * f[i][i];
}
for (i=0; i<k; i++)
f[k][k] -= w[i] * f[k][i];
f[k][k] = 1.0 /f[k][k];
}

for (k=1; k<m; k++)
for (i=0; i<k; i++)
b[k] -= f[k][i] * b[i];
for (k=m-1; k>=0; k--) {
b[k] *= f[k][k];
for (i=k+1; i<m; i++)
b[k] -= f[i][k] * b[i];
}
}
```

```
/*************************************************************************/
```

A saída é

```
1.384615
-0.857143
6.307693
Precione quaquer tecla para continuar. . .
```

O que está de acordo com a seção 7.3: $a_1 = \dfrac{18}{13} \approx 1.384615$, $a_2 = \dfrac{-6}{7} \approx -0.857143$ e $a_3 = \dfrac{82}{13} \approx 6.307693$.

9.3 Trocando o Produto Interno

Uma generalização do Método dos Mínimos Quadrados pode ser feito trocando o produto interno (3), por

$$< f, g >= \int_a^b f(x)g(x)\,dx, \tag{5}$$

sendo $f, g : [a, b] \longrightarrow \mathbb{R}$. Na realidade, estamos considerando $\mathcal{F}$ o espaço das funções contínuas em $[a, b]$. Tomando $f, g \in \mathcal{F}$, verifica-se que (5) é um produto interno em $\mathcal{F}$.

9.3.1 Aproximação por polinômios

Dada uma função $g : [a, b] \longrightarrow \mathbb{R}$, queremos aproximá-la de um polinômio

$$f(x) = a_0 + a_1x + a_2x^2 + \cdots + a_nx^n.$$

Do ponto de vista dos mínimos quadrados, desejamos minimizar

$$||g - f||^2 = \int_a^b [g(x) - f(x)]^2 dx. \tag{6}$$

De fato, queremos encontrar os coeficientes $a_0 + a_1 + a_2 + \cdots + a_n$ que minimizem (6).
Façamos, então, o Exemplo da seção 7.4, com o produto interno da integral (5). Quemos aproximar $g(x) = \text{sen}(x)$ do polinômio $f(x) = a_1x + a_2x^3$, em $[0, \pi/2]$. Pondo $f_1(x) = x$ e $f_2(x) = x^3$, de acordo com (1), devemos resolver o seguite sistema.

$$\begin{cases} < f_1, f_1 > a_1 + < f_1, f_2 > a_2 =< g, f_1 > \\ < f_2, f_1 > a_1 + < f_2, f_2 > a_2 =< g, f_2 > \end{cases} \tag{7}$$

Podemos calcular a matriz dos coeficientes de (7).

$$< f_1, f_1 >= \int_0^{\pi/2} x^2 dx = \frac{(\pi/2)^3}{3}$$

$$< f_1, f_2 >=< f_2, f_2 >= \int_0^{\pi/2} x^4 dx = \frac{(\pi/2)^5}{5}$$

$$< f_2, f_2 >= \int_0^{\pi/2} x^6 dx = \frac{(\pi/2)^7}{7}$$

Lembrado que $g(x) = \text{sen}(x)$, podemos calcular os termos do segundo membro de (7).

$$< g, f_1 >= \int_0^{\pi/2} x\,\text{sen}(x)\,dx = 1$$

$$< g, f_2 >= \int_0^{\pi/2} x^3\,\text{sen}(x)\,dx = \frac{3\pi^2 - 24}{4}$$

Resolvendo (7), obtemos

$a_1 = \dfrac{150}{\pi^3} - \dfrac{210(3\pi^2 - 24)}{\pi^5} \approx 0,98879223305331$

$a_2 = \dfrac{1400(3\pi^2 - 24)}{\pi^7} - \dfrac{840}{\pi^5} \approx -0,14506181330687$

O gráfico de $\text{sen}(x)$ e de $c_1x + c_2x^3$ feito pelo wxMaxima pode ser visto na Figura 22. Observemos que do ponto de vista visual não se nota diferença.

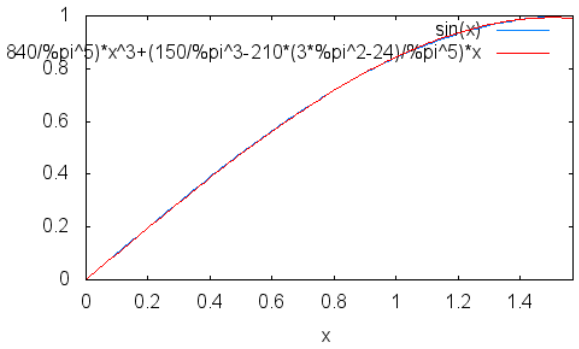


Figura 22: $\text{sen}(x) \approx c_1x + c_2x^3$

Do ponto de vista do produto interno (5), podemos avaliar o erro cometido:

$$||f - g||^2 = \int_0^{\pi/2} [c_1x + c_2x^3 - \text{sen}(x)]^2 dx \approx 8,2153141931748169.10^{-4}$$

ou

$$||f - g|| \approx 0,028662369394687.$$

O valor $||f - g||^2$ é chamado Erro Quadrático Médio.

9.3.2 Aproximação por Funções Trigonométricas - Coeficientes de Fourier

Em algumas situações necessitamos de proximar funções de polinômios trigonométricos. Isso pode ocorrer quando estamos estudando fenômenos ondulatórios como, por exemplo, a propagação do som [10].
Suponhamos que quiséssemos aproximar uma função contínua $g : [-L, L] \longrightarrow \mathbb{R}$ por uma função da forma:

$$\begin{aligned} f(x) &= a_0 + a_1\cos\frac{\pi x}{L} + b_1\text{sen}\frac{\pi x}{L} + a_2\cos\frac{2\pi x}{L} + b_2\text{sen}\frac{2\pi x}{L} + \cdots + a_n\cos\frac{n\pi x}{L} + b_n\text{sen}\frac{n\pi x}{L} \\ &= a_0 + a_1\cos\frac{\pi x}{L} + a_2\cos\frac{2\pi x}{L} + \cdots + a_n\cos\frac{n\pi x}{L} + b_1\text{sen}\frac{\pi x}{L} + b_2\text{sen}\frac{2\pi x}{L} + \cdots + b_n\text{sen}\frac{n\pi x}{L}. \end{aligned} \quad (8)$$

Pondo $c_i(x) = \cos\frac{i\pi x}{L}$ e $s_i(x) = \text{sen}\frac{i\pi x}{L}$, obtemos

$$f(x) = a_0c_0(x) + a_1c_1(x) + a_2c_2(x) + \cdots a_nc_n(x) + b_1s_1(x) + b_2s_2(x) + \cdots + b_ns_n(x)$$

ou

$$f = a_0c_0 + a_1c_1 + a_2c_2 + \cdots a_nc_n + b_1s_1 + b_2s_2 + \cdots + b_ns_n, \quad (9)$$

o que significa, em termos de Álgebra Linear, que f é combinação linear de $c_0,\ c_1\, c_2 \cdots, c_n$ e $s_1\, s_2 \cdots, s_n$. Uma função do tipo (8) é chamacha de polinômio trigonométrico de grau n. Portanto, desejamos aproximar g de um polinômio trigonométrico de grau n, isto é, queremos calcular $a_0,\ a_1\, a_2 \cdots, a_n$ e $b_1\, b_2 \cdots, b_n$ de modo que

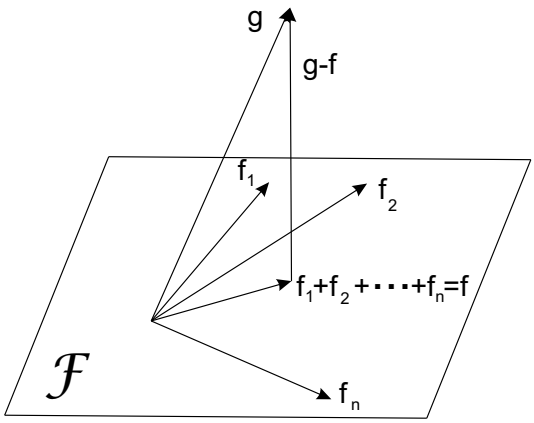


Figura 23: Aproximação de f a g

$$||g-f||^2 = \int_{-L}^{L} [g(x)-f(x)]^2 dx$$

seja mínimo. Denotemos por $\mathcal{F}$ o espaço gerado por $\{c_0,\, c_1\, c_2 \cdots, c_n$ e $s_1\, s_2 \cdots, s_n\}$. Representemos, simbolicamente, $\mathcal{F}$ por um plano como na Figura 23. Para minimizar $||g-f||$, temos que $g-f$ deve ser perpendicular a $\mathcal{F}$, como na figura 23 e, consequentemente, $< g-f, c_i >= 0$ e $< g-f, s_i >= 0$, $\forall i$ ou

$$< c_i, f >=< g, c_i > \text{ e } < s_i, f >=< g, s_i >, \forall i \tag{10}$$

Usando-se técnicas de integração, verifica-se que:

$< c_i, c_j >= \int_{-L}^{L} \cos\frac{i\pi x}{L}\cos\frac{j\pi x}{L}dx = 0$, para $i \neq j$,

$< s_i, s_j >= \int_{-L}^{L} \text{sen}\frac{i\pi x}{L}\text{sen}\frac{j\pi x}{L}dx = 0$, para $i \neq j$,

$< c_i, s_j >= \int_{-L}^{L} \cos\frac{i\pi x}{L}\text{sen}\frac{j\pi x}{L}dx = 0$, para $i \neq j$ e para $i = j$,

$< c_i, c_i >= \int_{-L}^{L} \cos\frac{i\pi x}{L}\cos\frac{i\pi x}{L}dx = L$, para $i > 0$,

$< s_i, s_i >= \int_{-L}^{L} \text{sen}\frac{i\pi x}{L}\text{sen}\frac{i\pi x}{L}dx = L$, para $i > 0$ e

$< c_0, c_0 >= \int_{-L}^{L} dx = 2L.$

Concluímos que o conjunto $\{c_0,\, c_1\, c_2 \cdots, s_1\, s_2 \cdots, s_n\}$ é ortogonal e que $||c_0||^2 = 2L$, $||c_i||^2 = ||s_i||^2 = L$, para $i \neq 0$. De (9) e (10), concluímos que

$$< c_i, f >=< g, c_i > \Leftrightarrow < c_i, a_0c_0 + a_1c_1 + \cdots a_nc_n >= \int_{-L}^{L} g(x)c_i(x)dx$$

ou

$$||c_i||^2 =< c_i, c_i >= \int_{-L}^{L} g(x)\cos\frac{i\pi x}{L}dx.$$

Para $i = 0$, $a_0 = \dfrac{1}{2L}\displaystyle\int_{-L}^{L} g(x)dx.$

Para $i \neq 0$, $a_i = \dfrac{1}{L}\displaystyle\int_{-L}^{L} g(x)\cos\frac{i\pi x}{L}dx.$

Analogamente,

$b_i = \dfrac{1}{L}\displaystyle\int_{-L}^{L} g(x)\text{sen}\dfrac{i\pi x}{L}dx.$

Resumindo, dado $g : [-L, L] \longrightarrow \mathbb{R}$, podemos aproximar g de um polinômio trigonométrico

$$g(x) \approx a_0 + a_1\cos\frac{\pi x}{L} + b_1\text{sen}\frac{\pi x}{L} + a_2\cos\frac{2\pi x}{L} + b_2\text{sen}\frac{2\pi x}{L} + \cdots + a_n\cos\frac{n\pi x}{L} + b_n\text{sen}\frac{n\pi x}{L},$$

no sentido dos mínimos quadrados. Para tanto, basta tomarmos

$$\boxed{\boxed{a_0 = \frac{1}{2L}\int_{-L}^{L} g(x)dx} \quad \boxed{a_i = \frac{1}{L}\int_{-L}^{L} g(x)\cos\frac{i\pi x}{L}, \forall i \neq 0} \quad \boxed{b_i = \frac{1}{L}\int_{-L}^{L} g(x)\text{sen}\frac{i\pi x}{L}}},$$

que são chamados de coeficientes de Fourier de g. Alguns autores preferem trocar a_0 por $\dfrac{a_0}{2}$ em (8) e eliminar a primeira igualdade acima.

Para sermos mais preciso, enunciemos o seguite resultado.

Teorema 9.1 *Se $g : [-L, L] \longrightarrow \mathbb{R}$ é contínua, então o polinômio trigonométrico*

$$f(x) = \frac{a_0}{2} + a_1\cos\frac{\pi x}{L} + b_1\text{sen}\frac{\pi x}{L} + a_2\cos\frac{2\pi x}{L} + b_2\text{sen}\frac{2\pi x}{L} + \cdots + a_n\cos\frac{n\pi x}{L} + b_n\text{sen}\frac{n\pi x}{L}$$

que minimiza

$$||g - f||^2 = \int_{-L}^{L}[g(x) - f(x)]^2dx$$

tem coeficientes

$$a_i = \frac{1}{L}\int_{-L}^{L} g(x)\cos\frac{i\pi x}{L}dx \text{ e } b_i = \frac{1}{L}\int_{-L}^{L} g(x)\text{sen}\frac{i\pi x}{L}dx.$$

9.4 Enfoque Matricial

O Enfoque matricial é bastante popular e pode ser visto, por exemplo, em [1] ou [2].

Lembramos que dados $f_1, \ldots, f_m$; $x_1, \ldots, x_n$ e $g_1, \ldots, g_n$, queremos encontrar

$$f(x) = a_1f_1(x) + \cdots + a_mf_m(x)$$

que melhor se se ajusta aos pontos (x_1, g_1),...,(x_n, g_n), no sentido dos mínimos quadrados, isto é, tal que $[f(x_1) - g_1]^2 + \cdots + [f(x_n) - g_n]^2$ seja mínimo. Se fosse possível encontrar f cujo gráfico contivesse os pontos (x_1, g_1),...,(x_n, g_n), deveríamos resolver o sistema

$$\begin{cases} f_1(x_1)a_1 + \cdots + f_m(x_1)a_m = g_1 \\ \cdots \\ f_1(x_1)a_1 + \cdots + f_m(x_n)a_m = g_n \end{cases}. \tag{11}$$

Definimos $d_{ij} = f_j(x_i)$ e $D = (d_{ij})_{n\times m}$,

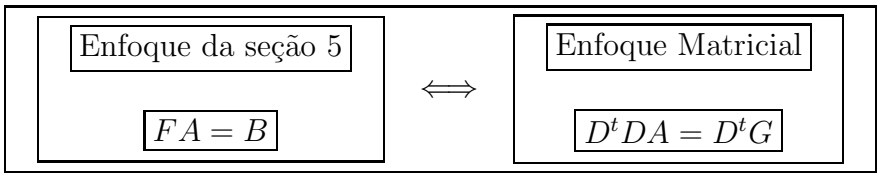
$$A = \begin{pmatrix} a_1 \\ \vdots \\ a_m \end{pmatrix} \text{ e } G = \begin{pmatrix} g_1 \\ \vdots \\ g_n \end{pmatrix}.$$

Desta forma o sistema (11) pode ser escrito na forma $DA = G$. Na prática sabemos que tal sistema é impossível, então o problema passa a ser encontrar A tal que $||DA - G||$ seja mínimo. Aqui $||.||$ é

proveniente do produto interno usual de $\mathbb{R}^n$. Verifica-se que $||DA-G||^2 = [f(x_1)-g_1]^2+\cdots+[f(x_n)-g_n]^2$, portanto minimizar $||DA-G||$ é resolver o problema dos mínimos quadrados. Podemos ver, por exemplo, em [1] ou [2] que minimizar $||DA-G||$ é equivalente a resolver o sistema $D^tDA = D^tG$, que é o enfoque matricial desejado.
Vamos, agora, verificar que resolver o sistema $D^tDA = D^tG$ é exatamente o que se fêz na seção 5. Definindo $F = D^tD$, temos que o elemento da linha i e coluna j de F é $\sum_{r=1}^{n} a_{ri}a_{r_j} = \sum_{r=1}^{n} f_i(x_r)f_j(x_r)$, que foi exatamente a definição de F da seção 5.
Definindo

$$B = D^tG = \begin{pmatrix} b_1 \\ \vdots \\ b_n \end{pmatrix},$$

temos $b_i = \sum_{r=1}^{n} d_{ri}g_r = \sum_{r=1}^{n} f_i(x_r)g_r$, que é exatamente a definição de b_i da seção 5.
Para finalizar, lembramos que a seção 5 propõe como solução dos mínimos quadrados a resolução de $FA = B$, que é exatamente o sistema $D^tDA = D^tB$.



Referências

[1] Santos, Reginaldo J., *Introdução à Álgrabra Linear*, UFMG, 2004. Disponível em

`http://www.mat.ufmg.br/~regi.`

[2] Santos, Reginaldo J. , *Álgebra Linear e Aplicações*, UFMG, 2006. Disponível em

`http://www.mat.ufmg.br/~regi.`

[3] Callioli, Carlos A. & Domingues, Hygino H. & Costa, Roberto C., *Álgebra Linear e Aplicações*, Atual Editora, sexta edição reformulada, 2010.

[4] Boldrini, José L. & Costa, Sueli I. R. & Figueiredo, Vera L. & Waltzler, Henry G., *Álgebra Linear*, Editora Harbra Ltda, São Paulo, terceira edição, 1986.

[5] Bueno, H., *Algebra Linear: um Segundo Curso*, Sociedade Brasileira de Matematica, Rio de Janeiro, 2006.

[6] Leithold, Louis, *O Cálculo com Geometria Analítica*, Vol. 2, Editora Harbra Ltda, terceira edição, 1990.

[7] Ikeda, K., *Modified Cholesky Method*, http://www-b2.is.tokushima-u.ac.jp/ ikeda/num/cholesky.c, 1993.

[8] Mizrahi, V. V., *Treinamento em Linguagem C*, segunda edição, Editora Pearson Prentice Hall, 2008.

[9] Plackett, R. L., *The discovery of the method of least squares*, Biometrika 59, pp 239-251; 1972.

[10] Aton, H & Rorres, C., *Álgebra Linear com Aplicações*, Editora Bookman, oitava edição, 2004.

[11] Helene, Otaviano, *Método dos Mínimos Quadrados com formalismo matricial*, Editora Livraria da Física, 2006.

Conteúdo

Índice

Francisco Satuf
UFT - Campus de Gurupi
satuf@uft.edu.br
http://satuf.net