

# **Relatório do ep3 - MAC300**

**Decomposição QR para o cálculo de mínimos quadrados**

**Nomes:**

Fernanda de Camargo Magano 8536044

Eduardo Delgado Coloma Bier 8536148

**Nusp:**

**São Paulo**

**2014**

## Problema de mínimos quadrados, decomposição QR usando reflexões

- O uso de refletores é mais eficiente do que o uso de rotações. As matrizes estão sendo consideradas como retangulares, o que é mais abrangente, pois o algoritmo serve também para matrizes quadradas.
- Uma das coisas implementadas na busca da eficiência foi: a matriz R é guardada na própria A, enquanto a Q não será guardada. Não é necessário guardar a Q, apenas os vetores u de cada iteração e o gama. Com isso já se consegue calcular o valores de Q, sem guardá-la explicitamente. O u(1) é armazenado na primeira coluna de A, u(2) na segunda e assim sucessivamente. O tau também é armazenado na própria matriz A.
- Para evitar que ocorram overflows ou underflows foi implementado o seguinte: procurar o maior elemento da matriz e dividir todos os elementos de A e de b por esse valor máximo. Portanto, foi feito um reescalamento.
- O algoritmo é usado tanto para posto completo, quanto para incompleto. Para isso, é usado o pivoteamento de colunas. Isso garante a troca das mesmas, colocando na frente a que tiver maior norma. Assim, num determinado momento do algoritmo, se a matriz for de posto incompleto, a máxima norma das colunas será menor do que um certo epsilon. Esse é momento em que se descobre o posto da matriz e o algoritmo de decomposição QR termina.
- Ainda pensando em eficiência, o código foi implementado com orientação a linhas, mantendo a essência do algoritmo de ir zerando as colunas, colocando a norma de cada coluna no primeiro elemento da mesma. Mesmo o cálculo de normas foi feito orientado a colunas. Cada norma foi sendo calculada parcialmente, percorrendo cada linha para tal finalidade.
- Outro ponto importante para o desempenho do algoritmo foi que as normas calculadas para as colunas são aproveitadas a cada passo, isto é, tem o custo inicial fixo de calcular todas as normas. Contudo, a cada iteração, essas normas previamente calculadas são reaproveitadas. O custo com essa melhoria é de  $2(m-1)$ , enquanto antes era  $2(m-1)(n-1)$  se fosse calcular a cada vez.

### -----Formato do arquivo-----

Os arquivos para leitura devem ter o seguinte formato:

Na primeira linha estará o inteiro n (número de linhas) e m (número de colunas);

Depois haverá n X m linhas com três números (os dois primeiros são inteiros: a linha e a coluna da matriz e o terceiro é um double – o valor do elemento);

Por último, n linhas contendo um inteiro e um double, indicando a posição e o valor dos elementos do vetor b.

### -----Testes usando QR e polinômios:-----

Para montar o polinômio de grau 5 pode-se ter em mente que se A é uma matriz e p(x) é o polinômio, “substitui-se” os x's por A, chegando numa matriz modificada que será usada no programa para cálculo dos mínimos quadrados. Assim, testa-se com outros polinômios, fazendo o mesmo processo de cálculo (substituindo x por A nos polinômios usados para a aproximação). O cálculo do resíduo vai mostrar se o polinômio usado fez uma aproximação boa ou razoável da almejada.

## -----Cálculo do polinômio:-----

- Imagine um polinômio de grau cinco. Então escolhemos 15 pontos do mesmo. Sejam eles: (x1, y1) ...(xm, ym), sendo m=15. Deseja-se conseguir aproximações usando mínimos quadrados
- O objetivo é encontrar polinômios de grau p que melhor se ajuste a esses dados. Eles são da forma  $y_i = c_p \cdot x_i^p + c_{(p-1)} \cdot x_i^{(p-1)} \dots + c_1 x_1 + c_0 x_0 + \text{resíduo}$

$$b = \begin{bmatrix} y_1 \\ . \\ . \\ . \\ y_n \end{bmatrix} \quad A = \begin{bmatrix} x_1^p & \dots & x_1 & 1 \\ x_2^p & \dots & x_2 & 1 \\ : & & : & \\ : & & : & \\ x_m^p & \dots & x_m & 1 \end{bmatrix}$$

- Monta-se a matriz A e b e resolve por QR
- A matriz A, como indicada, é construída usando potências de xi. É uma forma de conseguir uma base que melhor aproxima. Mais adequada do que a canônica, por exemplo.
- Os x's obtidos pelo QR serão usados na montagem do polinômio desejado
- Depois calcula-se o resíduo e é verificado se a aproximação foi boa ou não.
- Seja um polinômio:  $P(x) = x^5 + 2x^4 + 1x^3 + 3x^2 - 7x + 6$
- Considere os seguintes quinze pontos abaixo:

$$P(x) = x^5 + 2x^4 + 1x^3 + 3x^2 - 7x + 6$$

Pontos:

Então o b é:

(0, 6)

**b:**

(1, 6)

6

(-1, 16)

6

(1,5 ; 23,34375)

16

(2 ; 76)

23.34375

(2,3 ; 138,26863)

76

(2,8 ; 326,90688)

138.26863

(3,3 ; 680,04513)

326.90688

(4 ; 1626)

680.04513

(5 ; 4546)

1626

(5.5 ; 7087,59375)

4546

(6; 10656)

7087.59375

(7 ; 22056)

10656

(8; 41614)

22056

(10 ; 121,236)

41614

121236

O A será montado de acordo com o grau do polinômio que se deseja aproximar.

Observação: 2.25 está equivalendo a 2,25 (foi usado o ponto no lugar da vírgula, já que o computador segue o sistema americano).

Por exemplo, se quisermos usar **um polinômio de grau cinco** para fazer a aproximação, o A será igual a:

0	0	0	0	0	1
1	1	1	1	1	1
-1	1	-1	1	-1	1
7.59375	5.0625	3.375	2.25	1.5	1
32	16	8	4	2	1
64.36343	27.9841	12.167	5.29	2.3	1
172.10368	61.4656	21.952	7.84	2.8	1
391.35393	118.5921	35.937	10.89	3.3	1
1024	256	64	16	4	1
3125	625	125	25	5	1
5032.84375	915.0625	166.375	30.25	5.5	1
7776	1296	216	36	6	1
16807	2401	343	49	7	1
32768	4096	512	64	8	1
100000	10000	1000	100	10	1

**Polinômio de grau 5 -buscando uma boa aproximação:**

$$1.221742x^5 + 0.028770x^4 - 0.048396x^3 + 6.777989x^2 + 28.027789x + 97.246516$$

$$\text{Resíduo} = 2652.378247$$

**Matriz A – estimando por polinômio de grau 4:**

0	0	0	0	1
1	1	1	1	1
1	-1	1	-1	1
5.0625	3.375	2.25	1.5	1
16	8	4	2	1
27.9841	12.167	5.29	2.3	1
61.4656	21.952	7.84	2.8	1
118.5921	35.937	10.89	3.3	1
256	64	16	4	1
625	125	25	5	1
915.0625	166.375	30.25	5.5	1
1296	216	36	6	1
2401	343	49	7	1
4096	512	64	8	1
10000	1000	100	10	1

**Polinômio de grau 4:**

$$11.625817x^4 - 1.432606x^3 - 15.072538x^2 - 115.923644x - 355.044347$$

$$\text{Resíduo} = 130782.450037$$

### Matriz A – estimando por polinômio de grau 3:

0	0	0	1
1	1	1	1
-1	1	-1	1
3.375	2.25	1.5	1
8	4	2	1
12.167	5.29	2.3	1
21.952	7.84	2.8	1
35.937	10.89	3.3	1
64	16	4	1
125	25	5	1
166.375	30.25	5.5	1
216	36	6	1
343	49	7	1
512	64	8	1
1000	100	10	1

### Polinômio de grau 3:

$$104.552228x^3 - 43.062088x^2 + 147.358334x - 2619.018745$$

$$\text{Resíduo} = 29088.741949$$

## -----Perturbações em matrizes -----

- Foi testada uma matriz dois por dois, para ser melhor de visualizado o que estava acontecendo;
- A ideia foi perturbar o vetor b em apenas 0.01, mas como A é mal-condicionada, perturbar na direção do maxmag pode causar estragos na solução;
- A matriz A escolhida foi a seguinte:  
25 28  
10 12
- Essa matriz foi construída de modo que as colunas são quase LD. Portanto, é mal-condicionada. A pequena perturbação em b é feita na direção do maxmag de A que é  $[1 \ 1]^t$
- A norma infinita de  $\|\delta b\| / \|b\| = 0.01/53 = 1.887 \times 10^{-4}$
- A norma infinita de  $\|\delta x\| / \|x\| = 1.28/2.767241 = 0.4625$
- Fica visível de que a alteração em b é pequena, mas o erro relativo em x tem proporção maior (enquanto no b é da ordem de  $10^{-4}$  a alteração em x é de  $10^0$ );
- Os arquivos em que esses testes são realizados são: **ill\_conditioned** e **ill\_conditioned\_perturbado**
- A solução de x por mínimos quadrados do primeiro arquivo foi:  
**-1.000000**  
**2.767241**  
A do segundo:  
**-2.280000**  
**3.886810**  
Portanto, o delta x é  $[-1.28 \ 1.119569]$  transposto

- Esse exemplo mostra a interferência do número de condição num sistema e como mudanças escolhidas cuidadosamente podem causar alterações em  $x$ , principalmente se for na direção de máxima magnificação da matriz.

### -----Observações-----

- O programa foi feito em C;
- Alguns testes estão nos arquivos: `identidade`, `ill_conditioned`, `ill_conditioned_perturbado`, `triang`, `estima_pol_5`, `a_2x1`, `a_3x2`, `a_2x2`, `nula`, `testa_eps`;
- Para compilar e testar os arquivos basta digitar:  
**`gcc -lm EP3.c -o EP3 <nome de um dos arquivos acima> -v`**  
 Lembrando-se de que a opção `-v` é opcional (deve ser colocada apenas se quiser imprimir várias informações, como a matriz  $R$ , a matriz reescalada, normas, entre outros).
- O arquivo `testa_eps` é um teste de um caso extremo: a matriz não é nula, mas seus valores são menores que `eps`, então se comporta e é detectado como se fosse uma matriz nula.
- Os arquivos **`ill_conditioned`** e **`ill_conditioned_perturbado`** testam pequena perturbação em  $b$  e seu efeito em  $x$ .
- O teste do arquivo `a_2x1` é interessante, pois foi escolhido propositalmente com dimensões pequenas (é um vetor de duas linhas) para ficar mais evidente o que está ocorrendo no programa. Mostra o caso de posto incompleto e como o QR calcula os mínimos quadrados. Num vetor que seja  $\begin{bmatrix} 1 & 1 \end{bmatrix}$  transposto e  $b$  assuma os valores 5 e 9 (que é o que ocorre no arquivo), uma solução lógica esperada mentalmente seria 7, que é o que de fato acontece.
- Para compilar o arquivo que calcula os resíduos deve-se colocar:  
**`gcc -lm calcula_residuo.c -o calcula_residuo`** e para rodar:  
**`./calcula_residuo <arquivo contendo A e b> <arquivo contendo x aproximado>`**  
 em que  $x$  aproximado foi estimado a partir de um polinômio e que deseja-se calcular o resíduo para verificar se a aproximação foi boa.