**Análise Numérica – Prática Laboratorial Avaliada 2**

**Respostas devem ser entregues no inquérito Moodle aberto**

**desde 04–05–2020 até 10–05–2020 23:59h**

Procure nas páginas a seguir a sua versão de trabalho prático (2 páginas identificadas com o seu nome e número de aluno). Estude as questões consoante as orientações dadas na Prática Laboratorial de programação e as noções contidas na sebenta teórica, e discutidas nas aulas Teórico-Práticas. Depois de resolvidas as questões, entre no Moodle da disciplina para responder o inquérito/relatório da prática.
É recomendado usar ferramentas computacionais: calculadoras científicas, programas de cálculo numérico ou simbólico, instruções Matlab/Octave `linsolve`, `inv`, `cond`, `lu`, `chol`, ou códigos dos métodos diretos fornecidos no Moodle da disciplina `gaussred.m`, `subinv.m`, `choles.m`
É recomendada a consulta de dúvidas com os docentes e companheiros da disciplina, através dos fóruns abertos no Moodle, ou das atividades síncronas disponíveis no Microsoft Teams
**O Inquérito online deve ser respondido depois de estudar todas as questões, podendo-se alterar respostas só até a data limite**

**Versão:**196

**Trabalho Análise Numérica 04–05–2020 . Entregar até 10–05–2020 23:59h**

**Aluno:** **Docente** **Número:** **Docente**

• Determinados sistemas de equações lineares $A \cdot x = b$ exigem demasiadas condições nas incógnitas, o qual faz com que não existam pontos onde são satisfeitas todas as igualdades, isto é, não existe um vetor coluna $s$ que satisfaz $A \cdot s = b$.

Nestes casos, resulta habitual admitir que os termos independentes foram introduzidos com um erro, e procuramos uma **solução de mínimos quadrados**, um vetor coluna $s$ que satisfaz $A \cdot s = b^*$, pudendo ser a coluna $b^*$ diferente dos termos independentes $b$ fornecidos no enunciado, mas de maneira que o erro $\Delta(b^*, b) = \|b^* - b\|_2$ seja mínimo. Por tanto procuramos encontrar $s$ de maneira que $(A \cdot s - b)^t \cdot (A \cdot s - b)$ seja mínimo.

É conhecido que sempre existe algum ponto $s$ que produz um erro $\|A \cdot s - b\|_2$ mínimo, e que os pontos com esta propriedade são as soluções do **sistema de equações normais** associado ao sistema $A \cdot x = b$

$$\text{Sistema de equações normais } A^t \cdot A \cdot x = A^t \cdot b$$

Estes sistemas de equações normais sabemos que sempre têm solução, e a matriz de coeficientes do sistema $A^t \cdot A$ é sempre simétrica.

. . . . . .

• Muitos sistemas de classificação de autoridades (por exemplo o page-rank de Google) baseiam-se num mecanismo onde diferentes indivíduos $i = 1, 2, \ldots, k$ dispõem dum número $n_i$ de votos que podem distribuir entre os restantes indivíduos, como avaliação da sua capacidade numa área. Assim criamos uma tabela $v_{ij}$ indicando quantos votos recebe o indivíduo $i$ desde o indivíduo $j$, e onde o total de votos emitidos por um individuo $v_{1j} + v_{2j} + \ldots + v_{kj} = \sum_i v_{ij}$ está determinado (é o valor conhecido $n_j$). Assumimos que os votantes são também autoridades na área avaliada, alguns com maior e outros com menor peso.

Queremos que neste sistema o número de votos $n_i$ que reparte cada indivíduo coincida com o número de votos que recebe: $n_i = v_{i1} + \ldots + v_{ik} = \sum_j v_{ij}$. Desta maneira os indivíduos mais votados, sendo eles uma autoridade mais reconhecida, vão ter maior peso e capacidade para influenciar o total de votos que recebem os restantes indivíduos.

Admitamos que no total, os indivíduos repartem 1000 votos: $n_1 + \ldots + n_k = 1000$. Se chamamos $p_{ij} = v_{ij}/n_j$ a proporção de votos do indivíduo $j$ que são associados ao indivíduo $i$, neste sistema estaríamos a exigir:

$$\sum_i n_i = 1000, \quad n_i = \sum_k v_{ik} = \sum_k p_{ik} n_k, \quad (\forall i = 1 \ldots k)$$

Temos um sistema de $k + 1$ equações com $k$ incógnitas $n_i$. Se denotamos por $P \in M_{k\times k}(\mathbb{R})$ a matriz dos valores $p_{ij}$, se denotamos por $N \in M_{k\times 1}(\mathbb{R})$ a matriz coluna dos valores $n_i$, e se denotamos por $u_{1\times k} \in M_{1\times k}(\mathbb{R})$ a matriz linha com $k$ vezes a entrada 1, o sistema é representado em forma matricial:

$$\begin{bmatrix} (\mathrm{Id} - P) \\ u_{1\times k} \end{bmatrix} \cdot N = \begin{bmatrix} 0_{k\times 1} \\ 1000 \end{bmatrix}$$

[ Docente – Docente ]

• Queremos estabelecer os méritos dos 5 integrantes duma equipa de programação numa empresa.

- Pediu-se a cada um deles distribuir votos entre os companheiros, em função da qualidade dos códigos que estes geravam. A proporção de votos que o programador $j$ pretende destinar ao programador $i$ ficou registado numa matriz $[p_{ij}]$

- A seguir pediu-se a cada um deles distribuir votos entre os companheiros, em função da mais-valia que este trabalhador supunha para a equipa. A proporção de votos que o programador $j$ pretende destinar ao programador $i$ ficou registado numa matriz $q_{ij}$.
  As matrizes são dadas no formato Matlab (Atenção ao símbolo (’), os termos que aparentam ser linhas são os votos dum indivíduo e devem ser guardados como colunas) :

```
P=[ 0.0,0.6,0.3,0.0,0.1; 0.1,0.0,0.5,0.2,0.2; 0.5,0.3,0.0,0.2,0.0;
0.4,0.4,0.2,0.0,0.0; 0.5,0.2,0.2,0.1,0.0]';
Q=[ 0.0,0.3,0.2,0.4,0.1; 0.5,0.0,0.3,0.1,0.1; 0.4,0.3,0.0,0.2,0.1;
0.4,0.4,0.2,0.0,0.0; 0.1,0.4,0.3,0.2,0.0]';
```

Vamos considerar os dois sistemas de equações que permitem determinar as avaliações $(n_1, n_2, \ldots, n_5)$ dos programadores, relativamente à qualidade dos códigos, e à mais-valia para a equipa, respetivamente.

$$\boxed{A} \qquad\qquad \boxed{B}$$

$$\begin{bmatrix} (\mathrm{Id} - P) \\ u_{1\times 5} \end{bmatrix} \cdot X = \begin{bmatrix} 0_{5\times 1} \\ 1000 \end{bmatrix} \qquad \begin{bmatrix} (\mathrm{Id} - Q) \\ u_{1\times 5} \end{bmatrix} \cdot X = \begin{bmatrix} 0_{5\times 1} \\ 1000 \end{bmatrix}$$

Pretendemos ainda considerar um sistema adicional onde a avaliação trate de combinar os dois aspetos antes indicados, através dum sistema que não tem soluções:

$$\boxed{C} \qquad \begin{bmatrix} (\mathrm{Id} - P) \\ (\mathrm{Id} - Q) \\ u_{1\times 5} \end{bmatrix} \cdot X = \begin{bmatrix} 0_{5\times 1} \\ 0_{5\times 1} \\ 1000 \end{bmatrix}$$

1. Aplique o algoritmo de eliminação gaussiana com escolha parcial de pivô para resolver a equação matricial $\boxed{A}$

2. Aplique a instrução `linsolve` para resolver a equação matricial $\boxed{B}$

3. Identifique o sistema de equações normais $M \cdot X = D$ associado a $\boxed{C}$.

4. Identifique uma decomposição de Cholesky da matriz simétrica $M$. Use o algoritmo de substituição para determinar a inversa de $U$. Use $U$ e a sua inversa para encontrar uma solução $S$ de mínimos quadrados do sistema $\boxed{C}$.

5. Identifique, com a norma-infinito, o erro relativo se usamos $S$ como valor aproximado da solução no sistema $\boxed{A}$. Repetir para o sistema $\boxed{B}$.

6. Identifique qual dos sistemas estudados está melhor condicionado.

[ Docente – Docente ]

Começamos por introduzir as matrizes do enunciado em Matlab:

```
>>P=[ 0.0,0.6,0.3,0.0,0.1; 0.1,0.0,0.5,0.2,0.2; 0.5,0.3,0.0,0.2,0.0;
0.4,0.4,0.2,0.0,0.0; 0.5,0.2,0.2,0.1,0.0]';

>> Q=[ 0.0,0.3,0.2,0.4,0.1; 0.5,0.0,0.3,0.1,0.1; 0.4,0.3,0.0,0.2,0.1;
0.4,0.4,0.2,0.0,0.0; 0.1,0.4,0.3,0.2,0.0]';
```

• Partimos da primeira equação matricial. Sabemos construir a matriz identidade em Matlab como `eye()`, e uma matriz de entradas 1 como `ones()`. Podemos construir a matriz de coeficientes e a matriz de termos independentes em Matlab:

```
A1=[eye(5)-P;ones(1,5)]

B1=[zeros(5,1);1000]
```

A matriz ampliada deste sistema é `M=[A1,B1]`. Aplicamos o algoritmo de eliminação gaussiana com escolha parcial de pivô. Isto implica escolher sempre o pivô na primeira coluna disponível, e na linha onde se encontre a maior entrada (em valor absoluto) desta coluna. Podemos então aplicar eliminação gaussiana, e calcular manualmente as transformações elementares com instruções do tipo `M(i,:)+=c*M(j,:)` (somar na linha i, $c$ vezes a linha j) Depois, o sistema triangular é resolvido por substituição inversa, através de novas transformações elementares. Podemos aproveitar o código `gaussred()` e o código `subinv()` para simplificar este processo. Devemos ter cuidado e escolher sempre a primeira coluna disponível, e nesta coluna a linha onde esteja a entrada maior (em valor absoluto):

```
[Mred,l,c]=gaussred([A1,B1],1:5)

...

Linha do novo pivo:  1

Coluna do novo pivo:  1

...

Linha do novo pivo:  6

Coluna do novo pivo:  2

...

Linha do novo pivo:  2

Coluna do novo pivo:  3

...

Linha do novo pivo:  4

Coluna do novo pivo:  4

...
```

```
Linha do novo pivo:  3

Coluna do novo pivo:  5
```

Os pivôs foram escolhidos através do critério de escolha parcial, situados por esta ordem, nas linhas 1,6,2,4,3, e nas colunas 1,2,3,4,5,6. No fim estamos a guardar na variável `Mred` uma matriz ampliada que define um sistema equivalente ao original, mas com forma triangular, se as linhas são reordenadas. Podemos aplicar substituição inversa para resolver o sistema. Para isto podemos aplicar transformações elementares ou usar `subinv`, onde indicamos qual é `Mred`, uma lista `l` que indica, qual é o pivô presente em cada linha, e uma lista `c` que indica qual é o pivô presente em cada coluna.

```
>> subinv(Mred,l,c)
```

A resposta mostra que o número de votos para cada programador é $n_1 = 247.9$, $n_2 = 290.71$, $n_3 = 260.0$, $n_4 = 118.4$, $n_5 = 82.9$.

• Acabamos de executar os códigos para explorar como é obtida a solução. No entanto este mesmo processo seria aplicado de maneira silenciosa se tivéssemos pedido a Matlab `linsolve(A1,B1)`. Vamos agora repetir para a segunda equação, com esta instrução:

```
>> A2=[eye(5)-Q;ones(1,5)]

>> B2=B1

>> linsolve(A2,B2)
```

Encontramos que o número de votos para cada programador é agora $n_1 = 288$, $n_2 = 251.3$, $n_3 = 193.7$, $n_4 = 193.7$, $n_5 = 73.3$

• Agora consideramos o sistema de 11 equações com 5 incógnitas indicado em $\boxed{C}$. Matricialmente temos $A_3 \cdot X = B_3$. Tem demasiadas equações para poucas incógnitas e provavelmente não tem solução. O sistema de equações normais associado é $A_3^t \cdot A_3 \cdot X = A_3^t \cdot B_3$, um sistema cuja solução é a que menor erro quadrático produz no sistema $\boxed{C}$. Temos que usar então o sistema $M \cdot X = D$ com $M = A_3^t \cdot A_3$, $D = A_3^t \cdot B_3$.

```
>> A3=[eye(5)-P;eye(5)-Q;ones(1,5)]

>> B3=[zeros(5,1);zeros(5,1);1000]

>> M=A3'*A3

>> D=A3'*B3
```

Temos assim uma matriz de coeficientes simétrica $M$. Podemos ver com ajuda de `chol(M)` que existe decomposição de Cholesky $M = U^t \cdot U$, e portanto é definido positiva. Guardamos esta decomposição como uma matriz $U$:

```
>> U=chol(M)
```

Obtemos a matriz seguinte:

$$U = \begin{bmatrix} 1.93907 & -0.11346 & -0.02063 & 0.13408 & 0.33005 \\ 0.00000 & 1.92019 & -0.04288 & 0.16416 & 0.24344 \\ 0.00000 & 0.00000 & 1.91774 & 0.42227 & 0.49394 \\ 0.00000 & 0.00000 & 0.00000 & 1.86996 & 0.52793 \\ 0.00000 & 0.00000 & 0.00000 & 0.00000 & 1.71730 \end{bmatrix}$$

Agora, para calcular a inversa poderiamos usar `inv(U)`, mas vamos ser obedientes e vamos aplicar o algoritmo de substituição inversa para resolver $U \cdot X = \mathrm{Id}$. Podemos fazer transofrmações elementares de forma manual na matriz $[M, \mathrm{Id}]$ ou poupar esforço e aplicar o código `subinv([U,eye(5)],l,c)` indicando em `l` que as linhas 1 até 5 têm os pivôs 1 até 5, nesta ordem, e indicamos em `c` que as colunas 1 até 5 têm os pivòs 1 até 5, nesta ordem, e que as colunas 6 até 10 não correspondem com pivôs (escrevemos `NaN`):

```
>> Uinv=subinv([U,eye(5)],(1:5)',[1:5,NaN(1,5)])
```

Obtemos a seguinte inversa:

$$U^{-1} = \begin{bmatrix} 0.51571 & 0.03047 & 0.00623 & -0.04106 & -0.09260 \\ 0.00000 & 0.52078 & 0.01164 & -0.04835 & -0.06231 \\ 0.00000 & 0.00000 & 0.52145 & -0.11775 & -0.11378 \\ 0.00000 & 0.00000 & 0.00000 & 0.53477 & -0.16440 \\ 0.00000 & 0.00000 & 0.00000 & 0.00000 & 0.58231 \end{bmatrix}$$

Resolver o sistema com ajuda da inversa:

$$M \cdot X = D \Leftrightarrow U^t \cdot U \cdot X = D \Leftrightarrow U \cdot X = (U^t)^{-1} \cdot D = (U^{-1})^t \cdot D \Leftrightarrow X = U^{-1} \cdot (U^{-1})^t \cdot D$$

```
>> S=Uinv*Uinv'*D
```

Encontramos avaliações $n_1 = 258.8$, $n_2 = 268.2$, $n_3 = 225.7$, $n_4 = 150.7$, $n_5 = 86.9$

Pretendemos ainda determinar o erro relativo de $S$ como aproximação das soluções do sistema $\boxed{A}$ e do sistema $\boxed{B}$. Vamos guardar estas soluções como vetores $S_1$, $S_2$. Já foram calculados, mas podemos repetir a resolução e criar variáveis para cada um deste vetores:

```
>> S1=linsolve(A1,B1)

>> S2=linsolve(A2,B2)
```

Os erros relativos pedidos são $\frac{\|S_1 - S\|_\infty}{\|S_1\|_\infty}$, $\frac{\|S_2 - S\|_\infty}{\|S_2\|_\infty}$, que podem ser obtidos como:

```
>> norm(S-S1,inf)/norm(S1,inf)

>> norm(S-S2,inf)/norm(S2,inf)
```

No primeiro caso obtemos uma percentagem do 12% de erro, e no segundo caso uma percentagem do 15% de erro.

Finalmente, para saber qual dos sistemas está melhor condicionado, consideramos a matriz de coeficientes e calculamos o número de condição:

```
>> cond(A1,inf)

>> cond(A2,inf)

>> cond(A3,inf)

>> cond(M,inf)
```

Devemos observar que as matrizes de coeficientes $A_1$, $A_2$, $A_3$ nem sequer são quadradas, não se pode falar do número de condição no sentido clássico. De facto o sistema $\boxed{C}$ nem sequer tem soluções, e nos sistemas $\boxed{A}$ ou $\boxed{B}$ pequenas alterações nos coeficientes de $A_1$ ou $A_2$ poderiam fazer com que estes sistemas deixassem de ter soluções. Só faz sentido calcular o número de condição para a matriz quadrada $M$, que define as equações normais associadas a $\boxed{C}$.

**Respostas no relatório Final (Inquérito Moodle):**

- `Chave: 196`
- `Questão 2.1: 1,6,2,4,3`
- `Questão 2.2: 2,3,1,4,5`
- `Questão 2.3: O sistema [A] tem uma única solução`
- `Questão 2.4: 1,2,3,4,5`
- `Questão 2.5: 1.94`
- `Questão 2.6: 0.52`
- `Questão 2.7: 2,1,3,4,5`
- `Questão 2.8: 12`
- `Questão 2.9: 15`
- `Questão 2.10: O sistema de equações normais`

**Aviso: Não pode copiar diretamente estas respostas na sua resolução. As respostas corretas são diferentes, nas diferentes versões de exercício**