Métodos Numéricos

Um curso para o Mestrado Integrado em Engenharia Informática e Computadores da FEUP

Carlos Madureira
Cristina Vila
José Soeiro de Carvalho

Faculdade de Engenharia da Universidade do Porto
Departamento de Engenharia de Minas

Este documento é uma versão provisória, de utilização exclusiva no âmbito da disciplina de Métodos Numéricos do Mestrado Integrado em Engenharia Informática e Computadores da Faculdade de Engenharia da Universidade do Porto.

Todas as correcções e contribuições são bem-vindas!

Carlos M. N. Madureira (cmad@fe.up.pt)
Maria Cristina C. Vila (mvila@fe.up.pt)
José M. S. Soeiro de Carvalho (jmsoeiro@fe.up.pt)

Conteúdo

Lista de Figuras

Lista de Tabelas

1 Sistemas de Equações Lineares

Contents

Tables

1.1 Eliminação Gaussiana

O problema de encontrar a solução numérica para um sistema de equações lineares simultâneas

$$\begin{cases} a_{11}.x_1 + a_{12}.x_2 + ... + a_{1n}.x_n = b_1 \\ a_{21}.x_1 + a_{22}.x_2 + ... + a_{2n}.x_n = b_2 \\ \quad ..... \\ a_{n1}.x_1 + a_{n2}.x_2 + ... + a_{nn}.x_n = b_n \end{cases}$$

é um problema muito frequente, em larga medida porque tanto cientistas como engenheiros frequentemente não resistem à perigosa tentação de linearizar os seus problemas intrinsecamente não-lineares, cuja resolução é sempre incómoda e problemática.

O facto de dedicarmos um capítulo exclusivamente a este problema corresponde a reconhecer, mas não a endossar, esta tendência e também a reconhecer o seu carácter muito especial. Com efeito, do ponto de vista estrito da Análise Numérica, o problema dos sistemas lineares coloca-se, em certo sentido, no extremo oposto do da resolução de equações não-lineares que examinámos no Capítulo anterior: a sua solução canónica é de carácter directo mas iterativo; porém, devido ao risco de acumulação dos inevitáveis erros de arredondamento, quer dos dados quer dos resultados intermédios, o resultado final encontra-se frequentemente sujeito a erros

grosseiros, pelo que há necessidade de tomar precauções especiais a fim de evitar essas situações desagradáveis.

Notemos por **A** a matriz quadrada dos coeficientes e por **b** e **x** as matrizes-colunas dos termos independentes e das incógnitas:

$$\mathbf{A} = \begin{Vmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \dots & \dots & \dots & \dots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{Vmatrix} \quad \mathbf{b} = \begin{Vmatrix} b_1 \\ b_2 \\ \dots \\ b_n \end{Vmatrix} \quad \mathbf{x} = \begin{Vmatrix} x_1 \\ x_2 \\ \dots \\ x_n \end{Vmatrix}$$

de modo que o sistema pode ser escrito na forma simbólica $\mathbf{A}.\mathbf{x} = \mathbf{b}$ isto é,

$$\sum_{j=1}^{n} a_{ij}.x_j = b_i \qquad (i = 1,2,\dots,n)$$

Se a matriz quadrada **A** for não-singular, isto é, se $\det\{\mathbf{A}\} \neq 0$, então, como sabemos, o sistema tem solução x única que pode ser calculada pela regra de Cramer, a qual obriga a

$$N_n = N_{n-1} + (2n-1))$$

operações algébricas elementares (somas, subtracções, multiplicações e divisões). Se o leitor se der ao trabalho de calcular esta expressão recorrente até $n = 10$, por exemplo, dar-se-á conta, perante a enormidade dos números, da necessidade de encontrar um método computacionalmente mais económico. Um desses métodos é o *método de eliminação de Gauss*, habitualmente estudado nos cursos elementares de Álgebra e presente em muitas bibliotecas de rotinas matemáticas, que se desenvolve do seguinte modo:

- divida-se a primeira equação por a_{11}, para tornar unitário o primeiro coeficiente;
- multiplique-se esta nova primeira equação por $-a_{21}$ e some-se à segunda, obtendo-se uma nova segunda equação sem a incógnita x_1; em seguida multiplique-se a nova primeira equação por $-a_{31}$ e some-se à terceira, obtendo-se uma nova terceira equação, também sem a incógnita x_1 ; continue-se deste modo até que, após $n-1$ passos, se tenham $n-1$ equações que não contêm a incógnita x_1;
- repita-se o processo, trabalhando agora sobre a segunda coluna de coeficientes, de modo a eliminar a incógnita x_2 nas $n-2$ últimas equações, e assim sucessivamente, até que, se tudo correr bem, a última equação contém apenas a incógnita x_n com coeficiente unitário e , portanto, resolvida; o conjunto destes passos corresponde à fase de *triangularização* da matriz dos coeficientes;
- substitua-se o valor de x_n assim achado na penúltima equação (que tem apenas x_{n-1}, x_n) e resolva-se esta em ordem a x_{n-1} ; substituam-se os valores de x_{n-1}, x_n na penúltima equação e resolva-se esta em ordem a x_{n-2} ; continue-se deste modo até ter resolvido a primeira equação em ordem a x_1 ; o conjunto destes passos corresponde à fase de *substituição para trás* que culmina com a *diagonalização* da matriz do sistema.

Note-se de passagem que, neste processo, dividimos cada equação (depois de devidamente reduzida) pelo seu coeficiente em posição diagonal, num total de n divisões; ora, cada divisão divide

o valor do determinante do sistema pela mesma quantidade e as outras operações (simples combinações lineares de linhas) não alteram o valor do determinante; além disso, o determinante do sistema triangularizado vale uma unidade, de modo que o determinante do sistema pode ser calculado pelo produto dos divisores utilizados, o que constitui uma regra de cálculo bem mais eficaz que a conhecida regra de Sarrus.

Exemplo 1.1 Aplicação do Método de Gauss

Resolver o sistema:

$$\begin{cases} 3x+6y+9z=39 & (1) \\ 2x+5y-2z=3 & (2) \\ x+3y-z=2 & (3) \end{cases}$$

Divida-se (1) por 3:

$$\begin{cases} x+2y+3z=13 & (1') \\ 2x+5y-2z=3 & (2) \\ x+3y-z=2 & (3) \end{cases}$$

Multiplique-se $(1')$ por -2 e some-se a (2):

$$\begin{cases} x+2y+3z=13 & (1') \\ y-8z=-23 & (2') \\ x+3y-z=2 & (3) \end{cases}$$

Multiplique-se $(1')$ por -1 e some-se a (3):

$$\begin{cases} x+2y+3z=13 & (1') \\ y-8z=-23 & (2') \\ y-4z=-11 & (3') \end{cases}$$

Multiplique-se $(2')$ por -1 e some-se a $(3')$:

$$\begin{cases} x+2y+3z=13 & (1') \\ y-8z=-23 & (2') \\ 4z=12 & (3'') \end{cases}$$

Divida-se $(3")$ por 4:

$$\begin{cases} x+2y+3z=13 & (1') \\ y-8z=-23 & (2') \\ z=3 & (3''') \end{cases}$$

Substitua-se $z=3$ em $(2')$:

$$\begin{cases} x+2y+3z=13 & (1') \\ y=1 & (2'') \\ z=3 & (3''') \end{cases}$$

substitua-se $z = 3$ e $y = 1$ em $(1')$:

$$\begin{cases} x & = & 2 & (1'') \\ y & = & 1 & (2'') \\ z & = & 3 & (3''') \end{cases}$$

O valor do determinante do sistema é $3 \times 1 \times 4 = 12$.

Uma condição suficiente para que o algoritmo de Gauss seja viável é a de que nenhum dos divisores seja nulo; se tal condição se não verificar em dado passo, pode sempre pensar-se em trocar a ordem das colunas ainda não triangularizadas; se mesmo esse expediente falhar, o sistema não é resolúvel.

Algoritmo 1.1

```
  DADOS W(n,n+1)= A(n,n);b(n)
  INICIALIZE-SE p DE MODO QUE p(i) = i   (i = 1, 2, ..., n)
        PARA k =  1, 2, ..., n-1 {
                ACHAR O MENOR i >=  k TAL QUE W(i,k) != 0
                SE NÃO EXISTIR, A NÃO É INVERTÍVEL; PARAR
                CASO CONTRÁRIO, TROCAR p(k)  COM p(i)  E AS LINHAS k E I DE W
                PARA i = k+1, ..., n {
                        SEJAM m = W(k,k) , W(i,k) = W(i,k)/m
                                PARA j = k+1, ..., n+1 {
                                        SEJA W(i,j) = W(i,j) - m. W(k,j)
                                }
                }
        }
SE W(n,m)= 0 , A NÃO É INVERTÍVEL; PARAR
SEJA x(n) = b(n)
PARA k = n-1, ..., 1 {
    SEJA s= 0
    PARA j = k+1, ..., n {
        SEJA s = s + W(k,j).x(j)
    }
        SEJA x(k) = (b(k) - s)/W(k,k)
}
```

É fácil calcular o número de operações algébricas elementares necessárias para resolver um sistema de n equações pelo método de Gauss:

$$N_{Gn} = \frac{2}{3}.n.(n+1).(n+2) + n.(n-1)$$

e podemos compará-lo com o número de operações exigidas pela regra de Cramer. A economia obtida a partir de $n = 4$ torna-se verdadeiramente espectacular

Exemplo 1.2 Organização de cálculos

Uma das formas possíveis de organizar os cálculos é a que se apresenta a seguir. Repare na indexação das linhas, indicando **iteração.equação.transformação** e no registo da operação de que resulta a linha. Operações sobre colunas podem ser registadas na célula de título da coluna. São óbvias algumas regras de previsão e arrumação do espaço para os cálculos.

índice	operação linha	x_1	x_2	x_3	b
1.1		0.100	0.700	-0.300	-19.300
1.1.1	**1.1** / 0.100	1.000	7.000	-3.000	-193.000
1.1.2	**1.1.1** $\times$ -(0.300)	- 0.300	- 2.100	0.900	57.900
1.1.3	**1.1.1** $\times$ -(3.000)	-3.000	21.000	9.000	579.000
1.2		0.300	-0.200	10.000	71.400
1.2.1	**1.2** + **1.1.2**	0	-2.3000	10.900	129.300
1.3		3.0	-0.100	-0.200	7.850
1.3.1	**1.3** + **1.1.3**	0	20.900	8.800	656.850

índice	operação linha	x_1	x_2	x_3	b
2.1	**1.1.1**	1.000	7.000	-3.000	-193.000
2.2	**1.2.1**	0	-2.300	10.900	129.300
2.2.1	**2.2** / (-2.300)	0	1.000	-4.739	-56.217
2.2.2	**2.2.1** $\times$ -(20.900)	0	-20.900	99.045	1174.935
2.3	**1.3.1**	0	20.900	8.800	656.850
2.3.1	**2.3** + **2.2.2**	0	0	107.845	1831.785

índice	operação linha	x_1	x_2	x_3	b
3.1	**1.1.1**	1.000	7.000	-3.000	-193.000
3.2	**2.2.1**	0	1.000	-4.739	-56.217
3.3	**2.3.1**	0	0	107.845	1831.785
3.3.1	**3.3** / (107.845)	0	0	1.000	16.985

índice	operação linha	x_1	x_2	x_3	b
4.1	**1.1.1**	1.000	7.000	-3.000	-193.000
4.2	**2.2.1**	0	1.000	-4.739	-56.217
4.3	**3.3.1**	0	0	1.000	16.985

1.2 O Erro no Método de Gauss

O método de Gauss é, entre os métodos da Análise Numérica, um tanto invulgar pelo facto de, teoricamente, produzir uma solução exacta em um número finito de passos. No entanto, não seremos tão ingénuos que vamos supor que é isso exactamente que se passa na prática. Por um lado, e logo à partida, há os erros contidos na conversão dos dados (coeficientes, termos

independentes) e que, embora raramente pensemos neles, podem ter consequências sérias.

Exemplo 1.3 Representação binária exacta

Tome-se o caso de um número tão simples como $0.1_{(10)}$ que não tem representação binária exacta.

Exercício 1.1 Para confirmar a afirmação anterior, calcule $1_{(2)} \,/\, 110_{(2)}$.

Por outro lado, há os inevitáveis erros de arredondamento no decorrer das operações próprias do algoritmo, que podem produzir erros graves mesmo em um sistema de ordem baixa e, portanto, com poucas operações, como mostram os exemplos seguintes:

Exemplo 1.4 Erros graves

Seja o sistema:

$$\begin{cases} x+\frac{1}{3}y=1 \\ 2x+\frac{2}{3}y=2 \end{cases}$$

que é indeterminado (porque o seu determinante é nulo) e que na nossa máquina imaginária seria representado na forma

$$\begin{cases} 0.100\times10^{1}x+0.333\times10^{0}y=0.100\times10^{1} \\ 0.200\times10^{1}x+0.667\times10^{0}y=0.200\times10^{1} \end{cases}$$

que é perfeitamente determinado (porque o determinante vale 0.100×10^{-2}) e se resolve na forma

$$\begin{cases} 0.100\times10^{1}x+0.333\times10^{0}y=0.100\times10^{1} \\ 0.100\times10^{-2}y=0 \end{cases}$$

e dá

$$\begin{cases} x=1 \\ y=0 \end{cases}$$

o que é obviamente falso.
Note-se como é irónico que a técnica de arredondamento ($\frac{2}{3} \;\rightarrow\; 0.667\times10^{0}$) , destinada a diminuir o erro nos resultados finais, seja, neste caso, a responsável por um erro catastrófico: se, em vez de arredondar, tivéssemos truncado, o sistema continuaria indeterminado.

Exemplo 1.5 Mais erros graves

Resolver, na nossa máquina imaginária, o sistema

$$\begin{cases} x+400y=801 \\ 200x+200y=600 \end{cases}$$

cuja solução óbvia é

$$\begin{cases} x = 1 \\ y = 2 \end{cases}$$

Multiplicando a primeira equação por -200 somando o resultado à segunda, obtemos

$$\begin{cases} 0.100 \times 10^1 x + 0.400 \times 10^3 y = \quad 0.801 \times 10^3 \\ \qquad\qquad\qquad -0.798 \times 10^5 y = -0.159 \times 10^6 \end{cases}$$

o que dá

$$\begin{cases} x = 0.500 \times 10^1 \\ y = 0.199 \times 10^1 \end{cases}$$

O erro relativo em x é, portanto, de 400% e resulta da equação de substituição

$$x = 801 - 400y$$

que multiplica por 400 o erro cometido em y.

Torna-se, portanto, necessário, uma vez obtida a solução, tentar uma avaliação do erro cometido. Para isso, partindo do problema da forma

$$A.x = b$$

consideremos o problema perturbado

$$(A + \delta A).(x + \delta x) = b + \delta b$$

em que δA e δb são os erros (pequenos!) da matriz do sistema e dos termos constantes e δx é o erro resultante para a solução. Teremos, portanto, sucessivamente:

$$A.x + A.\delta x + \delta A.x + A.\delta x = b + \delta b$$
$$A.\delta x + \delta A.x + \delta A.\delta x = \delta b$$

e, se os erros nos dados forem suficientemente pequenos, poderemos desprezar $\delta A.\delta x$ em face de $A.\delta x$ e $\delta A.x$, pelo que:

$$A.\delta x = \delta b - \delta A.x$$

o que mostra que é possível calcular os δx (aproximadamente, se os erros forem pequenos!) mediante o próprio processo de eliminação gaussiana sobre um sistema de matriz idêntica à do problema inicial e só com segundos membros diferentes, mas só depois de conhecer a solução x.

Por outro lado, não basta estimar esta *estabilidade externa* (isto é, em relação aos potenciais erros dos coeficientes e dos termos constantes): é necessário também estimar a *estabilidade interna* (isto é, em relação aos erros de arredondamento no decorrer do cálculo); notar-se-á que esta segunda forma de estabilidade não depende apenas das características do sistema de equações em questão, mas também do método de resolução adoptado e da precisão da representação dos números na máquina.

Vejamos então como obter uma estimativa da estabilidade interna: seja x_0 uma solução aproximada (a nossa solução!) tal que

$$x = x_0 + \delta$$

e teremos

$$A.(x_0 + \delta) = b$$

isto é

$$A.\delta = b - A.x_0 = \varepsilon$$

em que ε é o que chamamos a *coluna dos resíduos* (cujas componentes são os resíduos das equações, isto é, as diferenças entre os segundos membros e os valores calculados dos primeiros). Esta relação mostra que também os δ podem ser obtidos por eliminação gaussiana sobre um sistema com matriz idêntica à do problema inicial, uma vez conhecida a solução deste. Note-se, porém, que esta segunda resolução será, por sua vez, afectada por erros do mesmo tipo dos da resolução do sistema inicial. Por isso, é absolutamente desaconselhável a tentativa ingénua de corrigir os valores x_0 calculados com as estimativas δ.

Destas considerações resulta claramente que, por trás da sua aparência directa e simples, o método de eliminação de Gauss contém muitas subtilezas ocultas.

Exemplo 1.6 Estabilidade externa

Seja o sistema

$$\begin{cases} 7x + 8y + 9z = 24 \\ 8x + 9y + 10z = 27 \\ 9x + 10y + 8z = 27 \end{cases}$$

e resolvamo-lo em **Maxima** pelo método de eliminação gaussiana, usando a colecção de funções para álgebra linear[1]:

```
(%i1) load(linearalgebra);
```

mediante os comandos `echelon(M)`[2] e `rowop(M,i,j,t)`[3]:

```
(%i2) A:matrix(
 [7,8,9],
 [8,9,10],
 [9,10,8]
);
```

[1] ver ficheiro SISTEMAS-estabilidade

[2] `echelon(M)` devolve uma matriz triangular superior de diagonal unitária obtida por eliminação gaussiana a partir da matriz `M`

[3] `rowop(M,i,j,t)` opera sobre a linha `i` da matriz `M` por $R_i \leftarrow R_i - t * R_j$

(%o2)
$$\begin{pmatrix} 7 & 8 & 9 \\ 8 & 9 & 10 \\ 9 & 10 & 8 \end{pmatrix}$$

```
(%i3) B:matrix(
 [24],
 [27],
 [27]
);
```

(%o3)
$$\begin{pmatrix} 24 \\ 27 \\ 27 \end{pmatrix}$$

```
(%i4) AB:addcol(A,B);
```

(%o4)
$$\begin{pmatrix} 7 & 8 & 9 & 24 \\ 8 & 9 & 10 & 27 \\ 9 & 10 & 8 & 27 \end{pmatrix}$$

```
(%i5) AB:echelon(AB);
```

(%o5)
$$\begin{pmatrix} 1 & \frac{8}{7} & \frac{9}{7} & \frac{24}{7} \\ 0 & 1 & 2 & 3 \\ 0 & 0 & 1 & 1 \end{pmatrix}$$

```
(%i6) AB:rowop(AB,2,3,2);
```

(%o6)
$$\begin{pmatrix} 1 & \frac{8}{7} & \frac{9}{7} & \frac{24}{7} \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{pmatrix}$$

```
(%i7) AB:rowop(AB,1,3,9/7);
```

(%o7)
$$\begin{pmatrix} 1 & \frac{8}{7} & 0 & \frac{15}{7} \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{pmatrix}$$

```
(%i8) AB:rowop(AB,1,2,8/7);
```

(%o8)
$$\begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{pmatrix}$$

A solução é, portanto,

$$\begin{cases} x = 1 \\ y = 1 \\ z = 1 \end{cases}$$

e a estabilidade externa pode ser estimada mediante:

$$A.\delta x = \delta b - \delta A.x$$

em que δa é o erro absoluto dos coeficientes da equação e δb o dos segundos membros. Calculando mais uma vez em **Maxima**

```
(%i9) DA:zeromatrix(3,3)+da;
```

$$(\%o9) \quad \begin{pmatrix} da & da & da \\ da & da & da \\ da & da & da \end{pmatrix}$$

```
(%i10) DB:zeromatrix(3,1)+db;
```

$$(\%o10) \quad \begin{pmatrix} db \\ db \\ db \end{pmatrix}$$

```
(%i11) X:col(AB,4);
```

$$(\%o11) \quad \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$$

```
(%i12) BP:DB-DA.X;
```

$$(\%o12) \quad \begin{pmatrix} db-3\,da \\ db-3\,da \\ db-3\,da \end{pmatrix}$$

```
(%i13) AP:addcol(A,BP);
```

$$(\%o13) \quad \begin{pmatrix} 7 & 8 & 9 & db-3\,da \\ 8 & 9 & 10 & db-3\,da \\ 9 & 10 & 8 & db-3\,da \end{pmatrix}$$

```
(%i14) AP:echelon(AP);
```

$$(\%o14) \quad \begin{pmatrix} 1 & \frac{8}{7} & \frac{9}{7} & \frac{db-3\,da}{7} \\ 0 & 1 & 2 & db-3\,da \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

```
(%i15) AP:rowop(AP,2,3,2);
```

$$(\%o15) \quad \begin{pmatrix} 1 & \frac{8}{7} & \frac{9}{7} & \frac{db-3\,da}{7} \\ 0 & 1 & 0 & db-3\,da \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

```
(%i16) AP:rowop(AP,1,3,9/7);
```

(%o16)
$$\begin{pmatrix} 1 & \frac{8}{7} & 0 & \frac{db-3da}{7} \\ 0 & 1 & 0 & db-3da \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

```
(%i17) AP:rowop(AP,1,2,8/7);
```

(%o17)
$$\begin{pmatrix} 1 & 0 & 0 & 3da-db \\ 0 & 1 & 0 & db-3da \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

teremos, portanto,

$$\begin{cases} \delta x = 3\delta A - \delta b \\ \delta y = -3\delta A + \delta b \\ \delta z = 0 \end{cases}$$

o que mostra o facto surpreendente de (para este sistema particular!) a incógnita z ser insensível a (pequenos!) erros dos dados. Se supusermos que $\delta A = \delta b = 0.5$ teremos

$$\begin{cases} \delta x = 1 \\ \delta y = -1 \\ \delta z = 0 \end{cases}$$

Exercício 1.2 Se o valor 0.5 que acabámos de usar fosse tomado como um majorante do erro dos dados, o resultado seria um majorante do erro do resultado?

Exemplo 1.7 Estabilidade interna

A estabilidade interna do mesmo sistema pode, por sua vez, ser estimada mediante[4]:

```
(%i18) AX0:addcol(A,B-A.X0);
```

(%o18)
$$\begin{pmatrix} 7 & 8 & 9 & 0 \\ 8 & 9 & 10 & 0 \\ 9 & 10 & 8 & 0 \end{pmatrix}$$

```
(%i19) AX0:echelon(AX0);
```

(%o19)
$$\begin{pmatrix} 1 & \frac{8}{7} & \frac{9}{7} & 0 \\ 0 & 1 & 2 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

```
(%i20) AX0:rowop(AX0,2,3,2);
```

[4] ver ficheiro SISTEMAS-estabilidade

(%o20)
$$\begin{pmatrix} 1 & \frac{8}{7} & \frac{9}{7} & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

```
(%i21) AX0:rowop(AX0,1,3,9/7);
```

(%o21)
$$\begin{pmatrix} 1 & \frac{8}{7} & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

```
(%i22) AX0:rowop(AX0,1,2,8/7);
```

(%o22)
$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

o que significa que, os erros internos são nulos, como seria de esperar visto termos usado o modo de cálculo exacto, com representação racional.

Se, porém, tivéssemos calculado a seis casas significativas em vírgula flutuante, a solução do problema inicial seria, como o leitor pode verificar

$$\begin{cases} x &= 0.999257 \times 10^{0} \\ y &= 0.100069 \times 10^{1} \\ z &= 0.999973 \times 10^{0} \end{cases}$$

o que representa

$$\begin{cases} \delta x &= -0.743000 \times 10^{-3} \\ \delta y &= +0.690000 \times 10^{-4} \\ \delta z &= -0.270000 \times 10^{-4} \end{cases}$$

A precisão externa, calculada nos mesmo termos e com estes dados seria:

$$\begin{cases} \delta x &= +0.100046 \times 10^{0} \\ \delta y &= -0.100043 \times 10^{0} \\ \delta z &= +0.166583 \times 10^{-4} \end{cases}$$

(note-se que estes números não podem comparar-se com os anteriores) A precisão interna, por seu lado, seria,

$$\begin{cases} \delta x &= +0.743674 \times 10^{-3} \\ \delta y &= -0.690639 \times 10^{-4} \\ \delta z &= +0.270489 \times 10^{-4} \end{cases}$$

Não há, portanto (como seria de esperar dado o carácter meramente aproximado do cálculo) total concordância entre estes valores e os efectivamente observados; este resultado mostra, como já tínhamos deixado apontado, que há que pôr de parte toda a esperança de, por este processo, suprir os erros de cálculo; com efeito, neste caso, se corrigíssemos os valores calculados das raízes por estes desvios calculados, a solução y pioraria substancialmente.

Porém, o método de Gauss não é de todo desprovido de esperança: o conceito de *vector dos resíduos*, ε , sugere, dada a linearidade do problema, uma técnica iterativa para melhoramento da solução:

- seja x_0 uma primeira solução pelo método de Gauss e calcule-se $\varepsilon_0 = b - A.x_0$;
- resolva-se em seguida o sistema $A.y_0 = \varepsilon_0$ e faça-se $x_1 = x_0 + y_0$, a nova aproximação;
- calcule-se em seguida $\varepsilon_1 = b - A.x_1$ e resolva-se o sistema $A.y_1 = \varepsilon_1$, fazendo $x_2 = x_1 + y_1$,

e assim sucessivamente, até que se obtenha um vector de resíduos ε_n satisfatório. Dada, porém, a natureza sempre aproximada dos cálculos, não se podem esperar demasiadas esperanças neste esquema, pelo que o número de iterações deve ser fixado sensatamente.

1.3 Técnicas Clássicas para Minimização dos Erros

Para evitar estes erros típicos do método de Gauss, têm, ao longo dos tempos, sido imaginados numerosos esquemas que, no fundo, se reduzem a combinações ou variantes das técnicas que descreveremos em seguida. No entanto, nenhum desses esquemas é perfeito. Na maior parte dos casos trata-se, fundamentalmente de evitar o aparecimento de valores muito altos como multiplicadores dos erros e de valores muito baixos como divisores das equações.

pivotagem parcial Já encontramos o conceito de *pivotagem parcial* quando, no algoritmo de eliminação, tratámos de evitar o embaraço de uma eventual divisão por zero e, para isso, fomos levados a permutar equações; no sentido mais geral, esta técnica consiste em, na eliminação de cada coluna, escolher, não o primeiro coeficiente não-nulo, mas o maior (em valor absoluto) coeficiente dessa coluna e em, naturalmente, usar a equação correspondente para proceder à eliminação. O efeito da pivotagem parcial é apenas o de uma reordenação das equações, embora essa reordenação não precise ser fisicamente implementada.

pivotagem total A *pivotagem total* consiste em escolher, não o maior coeficiente da coluna que se pretende eliminar, mas o maior de todos os coeficientes das equações ainda não tratadas. Existe razoável consenso entre os analistas numéricos no sentido de que a vantagem marginal produzida pela pivotagem total não compensa o esforço computacional envolvido no reordenamento total da matriz.

escalagem de linhas A *escalagem de linhas* consiste em, antes de iniciar o processo de eliminação, dividir cada equação por uma potência de 10 adequada, de modo a que o maior coeficiente se encontre entre 0.1 e 10; em termos mais gerais: se, em dada máquina, se utilizar a base β (o caso mais frequente é $\beta = 2$) para a aritmética de vírgula flutuante, divida-se cada equação pela potência de β que torna o seu maior coeficiente compreendido entre β^{-1} e β . Um outro esquema, mais simples de programar mas menos eficiente, consiste em dividir cada equação pelo seu mais alto coeficiente (ao contrário da técnica

anterior, esta produz desde logo erros de arredondamento). Na ausência de erros, qualquer destas técnicas não afecta a solução teórica do sistema.

escalagem de colunas A *escalagem de colunas* é semelhante à de linhas, mas afecta as soluções no sentido de que, se uma coluna i for dividida por β^{α} , a solução x_i encontrada no final deve ser multiplicada por β^{α} .

Em relação às técnicas de escalagem, pode mostrar-se (cf. [JR75]) que, quando se usa aritmética de vírgula flutuante, a escalagem por uma potência da base de numeração é completamente inócua, salvo quando usada em conjunto com a pivotagem.

Exemplo 1.8 Escalagem

Seja o sistema anterior

$$\begin{cases} x + 400y = 801 \\ 200x + 200y = 600 \end{cases}$$

que, escalado por linhas, dá

$$\begin{cases} 0.100 \times 10^{-2}x + 0.400 \times 10^{0}y = 0.801 \times 10^{0} \\ 0.200 \times 10^{0}x + 0.200 \times 10^{0}y = 0.600 \times 10^{0} \end{cases}$$

A técnica de pivotagem parcial exige que se comece a eliminação na 2^a equação e não na 1^a:

$$\begin{cases} 0.399 \times 10^{0}y = 0.798 \times 10^{0} \\ 0.100 \times 10^{1}x + 0.100 \times 10^{1}y = 0.300 \times 10^{2} \end{cases}$$

de onde resulta

$$\begin{cases} y = 0.200 \times 10^{1} \\ x = 0.100 \times 10^{1} \end{cases}$$

Note-se que, deste modo, um eventual pequeno erro de arredondamento em y (que neste caso não ocorreu) não seria amplificado por um coeficiente desastradamente alto, como sucedia no exemplo anterior.

1.4 Ordem e Condição de um Sistema

Como sabemos, a existência de uma *dependência linear entre linhas* (equações) de um sistema faz com que a sua solução seja *indeterminada*; na prática corrente do método de Gauss, esta situação é detectada pelo aparecimento de uma linha de zeros na matriz; na versão de pivotagem sistemática, porém, tal detecção não é automática; é, por isso, de boa prática, tentar sempre detectar a existência de um zero na coluna e, nesse caso, investigar se a respectiva linha é toda nula (incluindo o termo independente); se for este o caso, torna-se evidente que a equação correspondente é *linearmente dependente* das anteriores e que a variável que pretendíamos eliminar passa a constituir um *parâmetro* da solução do sistema (esta situação pode ocorrer mais que uma vez no decorrer da resolução de um mesmo sistema); se todos os coeficientes forem nulos mas o

termo independente o não for, a equação é *incompatível* com as anteriores e, portanto, o sistema é impossível.

Pode também suceder que surja uma coluna de zeros, precisamente aquela sobre a qual pretendíamos proceder à eliminação; isto significa, obviamente, que existe uma *dependência linear entre colunas*; neste caso o sistema é também indeterminado na incógnita correspondente (o que não impede que possa também ser impossível por outras razões).

Em qualquer dos casos (dependência de linhas ou de colunas), a *ordem* do sistema (ordem do maior determinante não nulo da matriz dos coeficientes) cai de uma unidade.

A ordem do sistema é, portanto, um indicador deste tipo de anomalias, mas um indicador pouco interessante porque não é capaz de distinguir uma situação de indeterminação de outra de impossibilidade. De qualquer modo, a consideração é irrelevante dado que, como vimos, a melhor maneira de calcular um determinante consiste em aplicar o método de Gauss à respectiva matriz.

Exemplo 1.9 Dependência linear de colunas

Dependência linear de colunas (ver ficheiro SISTEMAS-deplincol):

```
(%i1) m:matrix(
 [2,4,1,13],
 [1,2,-1,2],
 [1,2,2,11]
);
```

$$(\%o1)\qquad \begin{pmatrix} 2 & 4 & 1 & 13 \\ 1 & 2 & -1 & 2 \\ 1 & 2 & 2 & 11 \end{pmatrix}$$

```
(%i2) echelon(m);
```

$$(\%o2)\qquad \begin{pmatrix} 1 & 2 & \frac{1}{2} & \frac{13}{2} \\ 0 & 0 & 1 & 3 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

Exemplo 1.10 Dependência linear entre linhas

Dependência linear entre linhas (ver ficheiro SISTEMAS-deplinlin):

```
(%i1) m:matrix(
 [2,4,1,13],
 [1,2,.5,6.5],
 [1,1,1,6]
);
```

$$(\%o1)\qquad \begin{pmatrix} 2 & 4 & 1 & 13 \\ 1 & 2 & 0.5 & 6.5 \\ 1 & 1 & 1 & 6 \end{pmatrix}$$

```
(%i2) echelon(m);
```

(%o2)
$$\begin{pmatrix} 1 & 2 & \frac{1}{2} & \frac{13}{2} \\ 0 & 1 & -\frac{1}{2} & \frac{1}{2} \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

Detectada que seja uma dependência linear, um bom programa deve identificá-la tão de perto quanto possível, porque é altamente provável que o utilizador pretendesse, ou supusesse, que o sistema tivesse a maior ordem possível. Em termos numéricos, o problema da ordem de um sistema é um problema complexo, até porque, no decorrer do cálculo, os efeitos do arredondamento tanto podem fazê-lo descer (por exemplo, quando ocorre perda total de significado em uma subtracção de linhas) como subir (como no exemplo atrás, relativo ao arredondamento dos dados).

Exercício 1.3 Resolver os sistemas:

1.
$$\begin{cases} x+y+z & =6 \\ 3x-y+z & =-2 \\ 2x+0y+z & =2 \end{cases}$$

2.
$$\begin{cases} x+3y+2z & =6 \\ x-y+0z & =-2 \\ x+0y+z & =2 \end{cases}$$

3.
$$\begin{cases} x+z & =a \\ x+z & =b \\ x+z & =c \end{cases}$$

Dado que, em termos numéricos, a identificação do que é um valor nulo é sempre, em princípio, um problema muito complicado, sucede que a determinação da ordem de um sistema é, também, um problema complicado. No entanto, é óbvio que, por razões práticas, temos que construir um conceito que possa, pelo menos de forma parcial ou de modo aproximado, ajudar-nos a resolver o problema. Este conceito é o de *condição de um sistema*, o qual, por sua vez, dada a própria natureza do problema, é um conceito mal definido. A ideia geral é a de que um sistema é mal condicionado se pequenas variações dos coeficientes podem dar origem a variações desproporcionadamente grandes das soluções; ideia corresponde, portanto, aproximadamente à de *instabilidade externa* que atrás analisámos e o seu carácter vago resulta, naturalmente, do carácter vago das noções de "grande"e "pequeno"; em um outro sentido, o conceito de condição pode ser considerado como uma tentativa de formalizar um pouco a ideia de dependência linear face

à inevitabilidade dos arredondamentos. Se o mau condicionamento de um sistema resulta da própria natureza do processo físico que ele descreve, preferiremos falar em *instabilidade*. No entanto, nem todos os maus condicionamentos provêm do sistema físico, mas sim da formulação matemática que para ele escolhemos: em certo sentido, a discussão do Capítulo Um sobre o cálculo numérico de expressões mostrou já como diferentes expressões analíticas do mesmo problema matemático podem conduzir a cálculos com precisões completamente diferentes; com efeito, em casos extremos, pode mesmo suceder que a própria estratégia de cálculo seja responsável pelo mau condicionamento; por exemplo, sabe-se que a pivotagem pode transformar um sistema bem condicionado em um mal condicionado. Existem numerosos mal-entendidos acerca dos sistemas mal-condicionados:

- um deles é o de que um sistema adequadamente escalado que tem um determinante pequeno é necessariamente mal-condicionado;
- um outro é o de que o mau condicionamento se liga com a existência de um pequeno ângulo entre os gráficos de duas ou mais das equações.

Definiremos a *norma espectral* de uma matriz quadrada $\mathbf{A}$ como:

$$\|\mathbf{A}\| = \max_{\|\mathbf{x}\|=1} \|\mathbf{A}.\mathbf{x}\|$$

em que

$$\|\mathbf{x}\| = \sum_{i=1}^{n} x_i^2$$

é a *norma euclidiana* do vector $\mathbf{x}$.

exemplo:
Interpretação: Se $\mathcal{T}(\mathbf{x}) = \mathbf{A}.\mathbf{x}$ for a transformação linear produzida sobre o vector $\mathbf{x}$ pela matriz quadrada $\mathbf{A}$, a norma espectral da matriz pode ser interpretada como o máximo alongamento que sofre o raio de uma hiperesfera unitária que sofre essa transformação.

Definiremos a condição de uma matriz quadrada não-singular (isto é, invertível) como:

$$\text{cond}(\mathbf{A}) = \|\mathbf{A}\| \, . \left\|\mathbf{A}^{-1}\right\|$$

Esta definição não é, porém, adequadamente operacional no contexto do nosso problema, na medida em que implica o cálculo de $\mathbf{A}^{-1}$, que é, precisamente, o problema que pretendemos resolver. Em um tratamento mais sofisticado, o cálculo de $\text{cond}(\mathbf{A})$ baseia-se na teoria das forma quadráticas hermitianas, que não podemos aqui abordar; diremos apenas que pode ser estimado através das relações

$$\|\mathbf{A}\| \leq \sqrt{\sum_{j=1}^{n} \sum_{i=1}^{n} \left|a_{ij}\right|^2}$$

e

$$\text{cond}(\mathbf{A}) \geq \frac{\max_j \left\|a_{.j}\right\|}{\min_j \left\|a_{.j}\right\|}$$

O significado concreto da condição é dado pelas suas seguintes propriedades:

- se $\mathbf{A}$ for não-singular e

  $$\mathbf{A}.\mathbf{x} = \mathbf{b}$$
  $$\mathbf{A}.(\mathbf{x}+\delta\mathbf{x}) = \mathbf{b}+\delta\mathbf{b}$$

  será

  $$\frac{\|\delta\mathbf{x}\|}{\mathbf{x}} \leq \text{cond}(\mathbf{A}).\frac{\|\delta\mathbf{b}\|}{\|\mathbf{b}\|}$$

  o que significa que $\text{cond}(\mathbf{A})$ é um majorante da amplificação do erro relativo introduzido na solução pelo erro relativo dos segundos membros;

- Sendo

  $$\mathbf{A}.\mathbf{x} = \mathbf{b}$$
  $$\mathbf{A}.(\mathbf{x}+\delta\mathbf{x}) = \mathbf{b}+\delta\mathbf{b}$$

  será

  $$\frac{\|\delta\mathbf{x}\|}{\|\mathbf{x}+\delta\mathbf{x}\|} \leq \text{cond}(\mathbf{A}).\frac{\|\delta\mathbf{A}\|}{\|\mathbf{A}\|} = r$$

  e, se for $r < 1$, será ainda

  $$\frac{\|\delta\mathbf{x}\|}{\|\mathbf{x}\|} \leq \frac{r}{1-r}$$

  o que significa essencialmente a mesma coisa para os erros induzidos por $\mathbf{A}$.

O ponto a reter é, portanto, o seguinte: se a condição da matriz dos coeficientes do sistema for elevada, deveremos ter pouca confiança na solução, mesmo que os cálculos sejam executados de modo muito exacto; naturalmente, a inevitável presença de arredondamentos só pode piorar esta situação. Em particular, usando aritmética de base 2 em vírgula flutuante com n_m bits na mantissa, então um valor da condição da matriz do sistema maior que 2^{n_m-1} indica que a solução encontrada terá, provavelmente, pouco significado. Pode, além disso, mostrar-se (cf. [Wil65]) que a solução $\mathbf{x}$ calculada pelo método de Gauss é solução exacta de uma equação da forma

$$(\mathbf{A}+\delta\mathbf{A}).\mathbf{x} = \mathbf{b}$$

com

$$\|\delta\mathbf{A}\| = \max_i \sum_{j=1}^{n} |\delta a_{ij}|$$

aproximadamente proporcional a n^3 . Este é um primeiro exemplo de uma análise inversa de erros, em que se mostra que a solução calculada aproximadamente para o nosso problema é solução exacta de um problema ligeiramente perturbado.

Todas estas considerações apontam para o facto de as fraquezas do método de Gauss e das suas variantes e "aperfeiçoamentos"serem bem mais graves que o que correntemente se supõe e que o que a sua presença sistemática nas bibliotecas de subrotinas matemáticas pode levar a crer. Na realidade, trata-se de um método que hoje já se não deve recomendar especialmente. O leitor aceitará que todo o tempo e esforço que lhe pedimos até ao momento se destinou a desenvolver a sensibilidade e o espírito crítico para as peculiaridades dos métodos de cálculo numérico e, por outro lado, para pôr em guarda contra o risco da excessiva confiança nos resultados que saem do computador com a benção de uma grande softwarehouse. Na realidade, o método que sugerimos como preferível é o que se apresenta de seguida.

O método de eliminação gaussiana constituiu objecto de extensos e profundos estudos, cuja história dá uma boa medida do que é, em termos reais, a construção das matemáticas. Como recorda Herman H. Goldstine (cf. [Gol90]: "Dada uma matriz quadrada $n \times n$, real e invertível, $\mathbf{A}$, e uma matriz coluna $\mathbf{b} \in \mathbb{R}$, pretendemos calcular $\mathbf{x} = \mathbf{A}^{-1}.\mathbf{b}$. O processo conhecido como eliminação gaussiana, que constitui uma resposta ao problema, presta-se a implementação em computadores digitais automáticos e é também uma maneira de factorizar $\mathbf{A}$ em um produto $\mathbf{L}.\mathbf{U}$ em que $\mathbf{L}$ é triangular inferior e $\mathbf{U}$ é triangular superior. Uma vez conhecidas $\mathbf{L}$ e $\mathbf{U}$, a solução $\mathbf{x}$ é obtida resolvendo os dois sistemas triangulares

$$\mathbf{L}.\mathbf{c} = \mathbf{b}$$
$$\mathbf{U}.\mathbf{x} = \mathbf{c}$$

sendo a truncatura a única fonte de erro.

Em 1943 o famoso estatístico Hotelling (a resolução de sistemas de equações lineares é um problema constante e importante em Estatística) publicou uma análise que mostrava que o erro no cálculo do inverso aproximado, $\mathbf{X}$, podia crescer como 4^{n-1}; em especial, mostrou, de modo heurístico e não muito rigoroso, que o método de Gauss exige cerca de $k+0.6\times n$ dígitos durante o cálculo para se obter uma precisão de k dígitos; assim, para inverter uma matriz de ordem 100 seriam necessários 70 dígitos se se quisesse uma precisão final de 10 dígitos. Começou então a temer-se que a eliminação gaussiana fosse instável com o erro de truncatura e iniciou-se uma procura frenética de algoritmos alternativos.

Em 1947, Goldstine e von Neumann, em um formidável artigo de 80 páginas, corrigiram em certa medida esta ideia. Certos estudiosos escolheram esta data como a do nascimento da moderna Análise Numérica. Entre outras coisas, este artigo mostrava como o uso sistemático de normas matriciais podia permitir a análise de erros; porém, um seu lamentável efeito involuntário foi o de deixar a impressão de que só matemáticos do calibre de Goldstine e von Neumann seriam capazes de empreender tais análises e, pior ainda, que tal trabalho era espantosamente maçador. Em termos técnicos, o principal resultado estabelecido era o de que, se $\mathbf{A}$ for simétrica e definida

positiva, então o inverso calculado, $\mathbf{X}$, satisfaz a

$$\|\mathbf{A}.\mathbf{X}-\mathbf{I}\| \leq 14.2 \times n^2 \times \varepsilon \times \text{cond}(\mathbf{A})$$

com

$$\text{cond}(\mathbf{A}) = \|\mathbf{A}\| \, . \left\|\mathbf{A}^{-1}\right\|$$

em que $\|.\|$ é a norma espectral e ε é a unidade de truncatura do computador. Só quando $\mathbf{A}$ é demasiado próxima de singular é que o algoritmo falha e não produz nenhum $\mathbf{X}$, mas isso é exactamente o que seria de esperar. O interessante neste resultado era o carácter polinomial em n do erro e o mais difícil era estabelecer o valor do coeficiente numérico. Por outro lado, ninguém até hoje foi capaz de mostrar que um inverso computacionalmente correcto, $\mathbf{X}$, tenha necessariamente que ser o inverso de uma matriz próxima de $\mathbf{A}$, isto é, que

$$\mathbf{X} = (\mathbf{A}+\mathbf{E})^{-1}$$

com

$$\|\mathbf{E}\|/\|\mathbf{A}\| \ll 1$$

Na realidade, é altamente provável que tal condição não se cumpra na generalidade; o que pode mostrar-se é que cada coluna de $\mathbf{X}$ é a coluna correspondente da inversa de uma matriz próxima de $\mathbf{A}$; infelizmente, será, em geral, uma matriz diferente para cada coluna.

Infelizmente, estes resultados referem-se apenas a um caso particular, o das matrizes simétricas e definidas positivas.

A experiência prática, anterior a 1963, da resolução de sistemas de equações com calculadoras mecânicas com n até 18 convenceu o matemático James Hardy Wilkinson, do National Physical Laboratory do Reino Unido, e os seus colegas L. Fox e E. T. Goodwin, de que a eliminação gaussiana dá excelentes resultados mesmo quando $\mathbf{A}$ está longe de ser simétrica e definida positiva. Escrevendo $\mathbf{L}.\mathbf{U} = \mathbf{A}+\mathbf{K}$, em que $\mathbf{K}$ é uma pequena "matriz de erro", os resultados que então publicou foram que a solução calculada, $\mathbf{z}$, satisfaz a

$$(\mathbf{A}+\mathbf{K}).\mathbf{z} = \mathbf{b}$$

com (se os produtos internos forem acumulados em dupla precisão antes da truncatura final)

$$\|\mathbf{K}\| \leq g.\varepsilon(2.005 \times n^2 + n^3 + \frac{1}{4} \times \varepsilon \times n^4).|\mathbf{A}\|$$

em que g é o "factor de crescimento", definido como o cociente entre o maior valor intermédio gerado no processo e o elemento máximo de $\mathbf{A}$. O correspondente majorante dos resíduos é

$$\|\mathbf{b}-\mathbf{A}.\mathbf{z}\| \leq g.\varepsilon(2.005 \times n^2 + n^3).|\mathbf{z}\mathbf{A}\|$$

sob a condição de ser

$$\varepsilon \times n \ll$$

A importante quantidade g é extremamente fácil de calcular no próprio decorrer da execução do algoritmo.

1.5 Método de Khaletsky

Dado um sistema

$$\mathbf{A}.\mathbf{x} = \mathbf{b}$$

procuremos representar **A** na forma do produto de uma matriz triangular inferior, **B**, por uma matriz triangular superior de diagonal unitária, **C**:

$$\mathbf{A} = \mathbf{B}.\mathbf{C}$$

com

$$\begin{Vmatrix} b_{11} & 0 & \dots & 0 \\ b_{21} & b_{22} & \dots & 0 \\ \dots & \dots & \dots & \dots \\ b_{n1} & b_{n2} & \dots & b_{nN} \end{Vmatrix} \qquad \begin{Vmatrix} 1 & c_{12} & \dots & c_{1n} \\ 0 & 1 & \dots & c_{2n} \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & 1 \end{Vmatrix}$$

e é imediato verificar, procedendo à multiplicação, que

$$\begin{aligned} b_{i1} &= a_{i1} \\ b_{ij} &= a_{ij} - \sum_{k=1}^{j-1} b_{ik}.c_{kj} \quad (i \geq j) \\ c_{1i} &= \frac{a_{1i}}{b_{11}} \\ c_{ij} &= \frac{a_{ij} - \sum_{k=1}^{i-1} b_{ik}.c_{kj}}{b_{ii}} \quad (i < j) \end{aligned}$$

Então o vector procurado, **x**, pode ser calculado a partir dos sistemas encadeados

$$\begin{aligned} \mathbf{B}.\mathbf{y} &= \mathbf{b} \\ \mathbf{C}.\mathbf{x} &= \mathbf{y} \end{aligned}$$

sistema que é facilmente resolúvel por ter matrizes triangulares:

$$\begin{aligned} y_1 &= a_{1.n+1} \\ y_i &= \frac{a_{1.n+1} - \sum_{k=1}^{i-1} b_{ik}.y_k}{b_{ii}} \\ x_n &= y_n \\ x_i &= y_i - \sum_{k=i+1}^{n} b_{ik}.y_k \end{aligned}$$

Resulta evidente que os números y_i se calculam mediante o mesmo esquema que os c_{ij} e os x_i mediante o mesmo esquema que os b_{ij}, o que facilita notavelmente a programação. Além disso, o *método de Khaletsky* exige menos memória que o método de Gauss e torna-se mesmo particularmente simples:

- no caso das *matrizes simétricas* $a_{ki} = a_{ik}$, em que

$$c_{ij} = \frac{b_{ji}}{b_{ii}} \qquad (i < j)$$

- no caso de *matrizes definidas positivas* (isto é, tais que $\mathbf{x}^T.\mathbf{A}.\mathbf{x} > 0$ qualquer que seja $\mathbf{x} \neq 0$), em que $\mathbf{C} = \mathbf{B}^T$, o que reduz o cálculo à primeira fase, isto é, a cerca de metade.

Infelizmente, devido à sua menor divulgação e ao facto de, irracionalmente, muitos autores continuarem convencidos de que apenas se aplica ao caso das matrizes definidas positivas, não existem ainda estudos exaustivos sobre o erro deste método.

Exercício 1.4

1. Investigue as condições em que o método de Khaletsky pode falhar por envolver divisores nulos, e que tipos de soluções remediais se podem imaginar para esses casos.

2. Escreva um programa para resolver sistemas pelo método de Khaletsky e use-o sobre os exemplos trabalhados pelo método de Gauss, comparando tempos de execução, consumo de memória e resultados.

1.6 Métodos iterativos

Um método iterativo de resolução de sistemas de equações lineares pode ser vantajoso em relação a um método directo, quando, por exemplo, o número de equações é muito grande, e então os erros de truncatura e arredondamento propagados ao longo da aplicação de um método directo podem não compensar os erros afectos a uma solução aproximada resultante da aplicação de um método iterativo.
Pensemos no método de Picard-Peano ou da iteração simples que estudámos no capítulo anterior. É fácil estender a sua aplicação a um sistema de equações lineares.
Seja dado o sistema

$$\begin{cases} a_{11}x_1 & +a_{12}x_2 & + & \dots & +a_{1n}x_n = b_1 \\ a_{21}x_1 & +a_{22}x_2 & + & \dots & +a_{2n}x_n = b_2 \\ \vdots & \vdots & \vdots & \dots & \vdots \\ a_{n1}x_1 & +a_{n2}x_2 & + & \cdots & a_{nn}x_n = b_n \end{cases}$$

que em notação condensada toma a forma:

$$\sum_{j=1}^{n} a_{ij}x_j = b_i \ , \quad (1 \leq i \leq n)$$

Se em cada equação isolarmos uma incógnita no primeiro membro, teremos:

$$a_{ii}x_i = \sum_{\substack{j=1 \\ j\neq i}}^{n} -a_{ij}x_j + b_i \ , \quad (1 \leq i \leq n)$$

Como estamos lembrados, do capítulo **??**, uma das maiores limitações dos métodos iterativos é o facto de por vezes (não tão poucas vezes assim!), estes se afastarem progressivamente das soluções procuradas, divergindo. Portanto, também aqui, teremos que verificar as condições de convergência. Estas derivam das condições de convergência do método de Picard-Peano, sendo que para equações lineares as funções $g_i(x)$, do sistema de equações **??** são funções lineares e portanto as suas derivadas são constantes. Matematicamente, as condições de convergência dos métodos iterativos traduzem-se por:

$$\frac{1}{|a_{ii}|} \sum_{\substack{j=1 \\ j\neq i}}^{n} \left|a_{ij}\right| < 1 \ , \quad (1 \leq i \leq n)$$

Estas condições de convergência, correspondem, em álgebra matricial, à imposição de a matriz dos coeficientes das incógnitas ser diagonalmente dominante (o módulo de cada elemento da diagonal principal é superior à soma dos restantes elementos da linha correspondente).

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix}$$

A matriz A será de diagonal dominante se se verificar:

$$\begin{cases} |a_{11}| > \sum_{j=2}^{n} \left|a_{1j}\right| \\ |a_{22}| > \sum_{\substack{j=1 \\ j\neq 2}}^{n} \left|a_{2j}\right| \\ \vdots \\ |a_{nn}| > \sum_{j=1}^{n-1} \left|a_{nj}\right| \end{cases}$$

1.6.1 Método de Gauss-Jacobi

A transposição do método de Picard-Peano para os sistemas de equações lineares denomina-se método de Gauss-Jacobi, que tem por fórmula de recorrência:

$$x_i^{(k)} = \frac{1}{a_{ii}} \left[- \sum_{\substack{j=1 \\ j\neq i}}^{n} a_{ij}x_j^{(k-1)} + b_i \right] \ , \quad (1 \leq i \leq n)$$

1.6.2 Método de Gauss-Seidel

O método anterior pode ser melhorado, se em cada iteração, à medida que vamos obtendo uma nova aproximação para a incógnita x_{i-1} esta entrar no cálculo de x_i ainda na mesma iteração. Este melhoramento, designado por Gauss-Seidel, permite uma convergência mais rápida e traduz-se matematicamente por:

$$x_i^{(k)} = \frac{1}{a_{ii}} \left[-\sum_{\substack{j=1 \\ j<i}}^{n} a_{ij} x_j^{(k)} - \sum_{\substack{j=1 \\ j>i}}^{n} a_{ij} x_j^{(k-1)} + b_i \right] \quad , \quad (1 \leq i \leq n)$$

É possível, em alguns casos, acelerar o processo de convergência do método de Gauss-Seidel através da introdução de um factor de relaxação w. A esta variante conhecida por sobrerelaxação sucessiva (SOR - Successive Overrelaxation) corresponde a fórmula de recorrência:

$$x_i^{(k)} = w \left\{ \frac{1}{a_{ii}} \left[-\sum_{\substack{j=1 \\ j<i}}^{n} a_{ij} x_j^{(k)} - \sum_{\substack{j=1 \\ j>i}}^{n} a_{ij} x_j^{(k-1)} + b_i \right] \right\} + (1-w) x_i^{(k-1)} \quad , \quad (1 \leq i \leq n)$$

Note-se que para $w = 1$, estamos na presença do método de Gauss-Seidel.

Exemplo 1.11 Aplicação de Gauss-Seidel

Seja o sistema de equações lineares:

$$\begin{cases} 7x+2y & = 24 \\ 4x+10y+z & = 27 \\ 5x-2y+8z & = 27 \end{cases}$$

Podemos verificar que a matriz dos coeficientes das incógnitas é diagonalmente dominante:

$$A = \begin{bmatrix} 7 & 2 & 0 \\ 4 & 10 & 1 \\ 5 & -2 & 8 \end{bmatrix} \qquad \text{Com efeito:} \quad \begin{bmatrix} 1^{\text{a}}\text{linha} & |7| > |2|+0 \\ 2^{\text{a}}\text{linha} & |10| > |4|+|1| \\ 3^{\text{a}}\text{linha} & |8| > |5|+|2| \end{bmatrix}$$

Isolando cada incógnita na equação correspondente, virá:

$$\begin{cases} x = \frac{24-2y}{7} \\ y = \frac{27-4x-z}{10} \\ z = \frac{27-5x+2y}{8} \end{cases}$$

Que corresponderá às seguintes fórmulas de recorrência:

Para o método de Gauss-Jacobi

$$\begin{cases} x_{n+1} = \frac{24-2y_n}{7} \\ y_{n+1} = \frac{27-4x_n-z_n}{10} \\ z_{n+1} = \frac{27-5x_n+2y_n}{8} \end{cases}$$

Para o método de Gauss-Seidel

$$\begin{cases} x_{n+1} = \frac{24-2y_n}{7} \\ y_{n+1} = \frac{27-4x_{n+1}-z_n}{10} \\ z_{n+1} = \frac{27-5x_{n+1}+2y_{n+1}}{8} \end{cases}$$

No quadro 1.1 apresentam-se os resultados dos processo iterativos para os dois métodos, com os respectivos resíduos, para quatro iterações.

Tabela 1.1: Aplicação dos Métodos de Gauss-Jacobi e Gauss-Seidel

Gauss - Jacobi				Gauss - Seidel		
	Guess inicial				Guess inicial	
x	0			x	0	
y	0			y	0	
z	0			z	0	
	1ªiteração	Resíduos			1ªiteração	Resíduos
x	3.428571	5.400000		x	3.428571	2.657143
y	2.700000	17.089286		y	1.328571	1.564286
z	3.375000	11.742857		z	1.564286	0.000000
	2ªiteração				2ªiteração	
x	2.657143	3.417857		x	3.048980	0.009184
y	0.991071	4.553571		y	1.323980	0.236097
z	1.907143	0.439286		z	1.800383	0.000000
	3ªiteração				3ªiteração	
x	3.145408	0.910714		x	3.050292	0.048269
y	1.446429	2.007972		y	1.299845	0.006854
z	1.962054	1.530612		z	1.793529	0.000000
	4ªiteração				4ªiteração	
x	3.015306	0.401594		x	3.057187	0.004146
y	1.245631	0.711735		y	1.297772	0.004828
z	1.770727	0.248916		z	1.788701	0.000000

Exercício 1.5 Verifique se o sistema do exemplo da secção 1.1 pode ser resolvido por métodos iterativos.

Bibliografia

[CB81] Conte and De Boor. *Elementary Numerical Analysis, an algorithmic approach*. McGraw-Hill International Editions, Singapore, 1981.

[DB74] Dahlquist and Björck. *Numerical Methods*. Prentice-Hall, Inc., Englewood Cliffs, 1974.

[Gol90] Herman H. Goldstine. *Remembrance of things past*. ACM Press - Addison-Wesley, Reading, Massachusetts, 1990.

[Gol91] Goldberg. What every computer scientist should know about floating-point arithmetic. *ACM Computing Surveys*, 23(1), March 1991.

[Ham71] Hamming. *Introduction to applied numerical analysis*. McGraw-Hill-Kogakusha, Ltd., Tokyo, 1971.

[JR75] Jensen and Rowland. *Methods of Computation*. Scott, Foresman & Co, Glenview, 1975.

[Lie68] Lieberstein. *A course in numerical analysis*. Harper and Row, New York, 1968.

[Moo63] Moore. *Interval Analysis*. Prentice-Hall, Englewood Cliffs, 1963.

[Mul89] Muller. *Arithmétique des ordinateurs*. Masson, Paris, 1989.

[Wal90] Wallis. *Improving floating-point programming*. Wiley, New York, 1990.

[Wil65] Wilkinson. *Rounding errors in algebraic processes*. Clarendon Press, Oxford, 1965.

[Zac96] Joseph L. Zachary. *Introduction to scientific programming: computational problem solving using Maple and C*. Springer Verlag New York Inc., New York, 1 edition, 1996.

Índice

Impresso em 16 de Abril de 2009 a partir dos seguintes ficheiros:

Analise_Numerica.tex	1.10	2009/03/10
Analise_Numerica.sty	1.11	2009/04/16
Titulo.tex	1.5	2009/02/27
3_Sistemas.tex	1.14	2009/04/16