**Forma Canônica de Jordan e Método de
Ponto Fixo Aplicados à Resolução de
Equações Polinomiais Matriciais**

Adalberto Rodrigo Rosales Carrasco

DISSERTAÇÃO APRESENTADA
AO
INSTITUTO DE MATEMÁTICA E ESTATÍSTICA
DA
UNIVERSIDADE DE SÃO PAULO
PARA A OBTENÇÃO DO GRAU DE MESTRE
EM
MATEMÁTICA APLICADA

Área de Concentração: Análise Numérica
Orientador: **Prof. Dr. Luiz Antonio Peresi**

Durante a elaboração desta dissertação,
o autor recebeu apoio financeiro do CNPq.

**Forma Canônica de Jordan e Método de
Ponto Fixo Aplicados à Resolução de
Equações Polinomiais Matriciais**

Este exemplar corresponde à redação final da dissertação
apresentada por Adalberto Rodrigo Rosales Carrasco,
devidamente corrigida e aprovada pela comissão julgadora.

São Paulo, agosto de 1997.

Banca examinadora:

Prof. Dr. Luiz Antonio Peresi (orientador)

IME-USP

Prof. Dr. Cyro de Carvalho Patarra

IME-USP

Prof. Dr. Lúcio Tunes dos Santos

IMECC-UNICAMP

Agradecimentos

Gostaria de agradecer:

- ao Prof. Luiz A. Peresi pela paciência que teve para corrigir cada uma das minhas falhas na redação deste trabalho e por ter me ajudado, mediante as discussões que tivemos, a compreender algumas coisas,
- ao Prof. Cyro Patarra por ter proposto o problema e por sua ajuda como o meu orientador inicial,
- ao Departamento de Matemática Aplicada pelo seu apoio material,
- aos meus amigos pela sua ajuda para “sobreviver” num país que não é o meu, e muito especialmente
- aos meus pais, porque sem eles não poderia chegar até onde tenho chegado até agora.

Obrigado.

Agradecimentos

Às pessoas que me ajudaram a escrever este livro, especialmente à minha esposa, à minha filha e aos meus pais.

À minha esposa, que me deu a oportunidade de escrever este livro, e à minha filha, que me deu a oportunidade de escrever este livro. Aos meus pais, que me deram a oportunidade de escrever este livro. Aos meus amigos, que me deram a oportunidade de escrever este livro. Aos meus colegas, que me deram a oportunidade de escrever este livro. Aos meus professores, que me deram a oportunidade de escrever este livro. Aos meus alunos, que me deram a oportunidade de escrever este livro. Aos meus leitores, que me deram a oportunidade de escrever este livro.

Aos meus amigos, que me deram a oportunidade de escrever este livro. Aos meus colegas, que me deram a oportunidade de escrever este livro. Aos meus professores, que me deram a oportunidade de escrever este livro. Aos meus alunos, que me deram a oportunidade de escrever este livro. Aos meus leitores, que me deram a oportunidade de escrever este livro.

Sumário

Agradecimentos	vii
Resumo	ix
Abstract	xi
Introdução	1
1 Preliminares	3
1.1 A Forma Canônica de Jordan	3
1.2 Exemplos	9
1.3 Equações Diferenciais Ordinárias	15
2 Equações Polinomiais Matriciais	19
2.1 Definições	19
2.2 Existência de Soluções	22
2.3 Algoritmo	32
3 Um Método de Ponto Fixo	35
3.1 Existência de Soluções	35
3.2 Algoritmo	42
4 Discussão	43
4.1 Existência de Soluções	43
4.2 Estabilidade de Soluções	47

4.3 Conclusões	49
A Programas	51
A.1 Programação do Algoritmo 2.1	51
A.2 Programação do Algoritmo 3.1	73
Referências Bibliográficas	79

Resumo

No presente trabalho apresentamos dois métodos numéricos para resolver Equações Polinomiais Matriciais que tem como coeficientes matrizes complexas. Um deles é baseado na Forma Canônica de Jordan e o outro é baseado no Teorema de Ponto Fixo de Banach.

No primeiro método (Forma Canônica de Jordan) só existe uma restrição com respeito à Equação Polinomial Matricial, a saber, o coeficiente matricial do termo de maior grau deve ser a matriz identidade. No segundo método (Ponto Fixo) a restrição principal é que o coeficiente matricial do termo de potência 1 seja uma matriz não-singular.

Este trabalho generaliza um método numérico existente para resolver Equações Polinomiais Matriciais quando os coeficientes são matrizes triangulares superiores. No final do trabalho são discutidos os três métodos.

CONTÍDUA

1. Introdução	1
2. Objetivos	2
3. Metodologia	3
4. Resultados	4
5. Conclusões	5
6. Referências	6
7. Anexos	7
8. Bibliografia	8
9. Glossário	9
10. Índice	10
11. Apêndice	11
12. Resumo	12
13. Abstract	13
14. Introdução	14
15. Objetivos	15
16. Metodologia	16
17. Resultados	17
18. Conclusões	18
19. Referências	19
20. Anexos	20
21. Bibliografia	21
22. Glossário	22
23. Índice	23
24. Apêndice	24
25. Resumo	25
26. Abstract	26

Abstract

In this work, two numerical methods to solve the Matrix Polynomial Equations (that have complex matrices as coefficients) are presented. One of them is based on the Jordan's Canonical Form and the other one is based on the Banach's Fix Point Theorem.

In the first method (Jordan's Canonical Form) there is only one restriction with respect to the Matrix Polynomial Equation, that is, the matrix coefficient of the term with the higher order must be the identity matrix. In the second method (Fix Point) the main restriction is that the matrix coefficient of the term with power one must be a non-singular matrix.

This work generalizes a numerical method to solve Matrix Polynomial Equations when all the coefficients are upper triangular matrices. The three methods are discussed in the final part.

Abstract

The purpose of this study was to determine the effect of the use of a mobile application on the learning of the concepts of the cell cycle and mitosis in a group of students of the Faculty of Sciences of the University of the Pacific. The study was carried out in a classroom of the Faculty of Sciences of the University of the Pacific, during the first semester of 2023. The sample consisted of 30 students, who were divided into two groups: a control group and an experimental group. The control group received traditional teaching, while the experimental group received teaching using the mobile application. The results showed that the use of the mobile application had a positive effect on the learning of the concepts of the cell cycle and mitosis, as evidenced by the higher scores achieved by the experimental group compared to the control group. The conclusions of the study indicate that the use of mobile applications can be a useful tool for teaching biology concepts, as it allows for a more interactive and personalized learning experience.

Introdução

O presente trabalho tem por finalidade estudar dois métodos para resolver Equações Polinomiais Matriciais que tem como coeficientes matrizes complexas. Um deles é baseado no estudo da Forma Canônica de Jordan de uma matriz associada com a equação, e o outro é baseado no Teorema de Ponto Fixo de Banach. Também comparamos estes métodos com o já existente no caso em que as matrizes coeficientes da equação são todas matrizes triangulares superiores ou simultaneamente triangularizáveis. Com este propósito, no capítulo dedicado às Preliminares são abordados aspectos teóricos relacionados com o cálculo da Forma Canônica de Jordan e com as Equações Diferenciais. Isto é feito porque vamos encontrar uma relação entre a Equação Polinomial Matricial e uma Equação Diferencial da ordem do grau da primeira.

No Capítulo 2, estudamos os diferentes casos que se apresentam com respeito à Forma Canônica de Jordan associada com a equação em estudo. A partir deste estudo detalhado, enunciamos o Teorema 2.1 que fornece condições necessárias para a existência de soluções de Equações Polinomiais Matriciais quando o coeficiente do termo de maior grau é a matriz identidade (este estudo aparece em um artigo escrito em chinês, cuja tradução nos foi comunicada pelo Professor Cyro de Carvalho Patarra). Neste mesmo capítulo, fornecemos um algoritmo que encontra uma solução da equação em estudo sempre que existir solução.

No Capítulo 3, apresentamos um Método de Ponto Fixo, baseado justamente no Teorema de Ponto Fixo de Banach (Teorema 3.1). O resultado original deste capítulo é o Teorema 3.2 que dá condições suficientes para a existência de soluções numa Equação Polinomial Matricial. Também fornecemos um algoritmo que encontra as soluções de uma

equação polinomial matricial neste caso. Como veremos no Capítulo 4, a condição imposta à matriz correspondente à menor potência de X na equação não é muito forte, embora tenhamos essa impressão de início. Esta última afirmação pode ser confirmada pelo Teorema 4.2 que fala sobre aproximações de coeficientes matriciais e a relação com as soluções da equação.

No Capítulo 4 mostramos o método já existente (feito pelo Professor F. E. Menzaque em sua Tese de Doutorado, apresentada no IME-USP em 1993) e o comparamos com os dois métodos que fornecemos neste trabalho.

Como parte final apresentamos os programas usados na elaboração dos exemplos dados no trabalho.

Capítulo 1

Preliminares

Os pré-requisitos que precisamos para poder solucionar o problema, tratado nesta dissertação, serão dados neste capítulo.

1.1 A Forma Canônica de Jordan

Teorema 1.1 (Teorema da Decomposição Primária). Sejam V um espaço vetorial de dimensão finita sobre o corpo $\mathbb{C}$ e $T : V \mapsto V$ um operador linear. Seja p o polinômio minimal de T ,

$$p = p_1^{r_1} \cdots p_k^{r_k},$$

onde os p_i são polinômios distintos, irredutíveis e unitários sobre $\mathbb{C}$ e os r_i são inteiros positivos. Seja W_i o núcleo de $p_i^{r_i}(T)$, $i = 1, \dots, k$. Então temos:

- (a) $V = W_1 \oplus \dots \oplus W_k$;
- (b) cada W_i é invariante sob T ;
- (c) se T_i é o operador induzido sobre W_i por T , então o polinômio minimal de T_i é $p_i^{r_i}$.

Prova. Ver [8], pp. 194-195. ■

Seja V um espaço de dimensão finita. Suponhamos que T seja um operador linear sobre V e que o polinômio característico de T se decompõe sobre o corpo $\mathbb{C}$ como segue:

$$f = (x - \lambda_1)^{m_1} \dots (x - \lambda_k)^{m_k},$$

onde $\lambda_1, \dots, \lambda_k$ são elementos distintos em $\mathbb{C}$ e $m_i \geq 1$. Então, o polinômio minimal de T é

$$p = (x - \lambda_1)^{r_1} \dots (x - \lambda_k)^{r_k},$$

onde $1 \leq r_i \leq m_i$. Se W_i é o núcleo de $(T - \lambda_i I)^{r_i}$, então o Teorema da Decomposição Primária nos diz que

$$V = W_1 \oplus \dots \oplus W_k,$$

e que o operador T_i induzido sobre W_i por T possui polinômio minimal igual a $(x - \lambda_i)^{r_i}$. Seja N_i o operador linear sobre W_i definido por $N_i = T_i - \lambda_i I$. Então, N_i é nilpotente e seu polinômio minimal é x^{r_i} . Sobre W_i , T age como N_i mais o escalar λ_i vezes o operador idêntico. Suponhamos que temos uma base do subespaço W_i correspondente à decomposição cíclica do operador nilpotente N_i . Então a matriz de T_i em relação a esta base ordenada é a soma direta das matrizes

$$\begin{pmatrix} \lambda & 1 & 0 & \dots & 0 \\ 0 & \lambda & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \dots & \lambda & 1 \\ 0 & 0 & \dots & 0 & \lambda \end{pmatrix}, \quad (1.1)$$

cada uma com $\lambda = \lambda_i$. Além disso, as dimensões destas matrizes diminuem quando se lê da esquerda para a direita. Uma matriz da forma (1.1) é dita uma **matriz elementar de Jordan com valor característico λ** . Reunindo todas as bases dos W_i obtemos uma base de V . Seja J a matriz de T em relação a esta base ordenada. A matriz J é a soma direta

$$\begin{pmatrix} J_{\lambda_1} & 0 & \dots & 0 \\ 0 & J_{\lambda_2} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & J_{\lambda_k} \end{pmatrix} \quad (1.2)$$

das matrizes $J_{\lambda_1}, \dots, J_{\lambda_k}$. Cada J_{λ_i} é da forma

$$J_{\lambda_i} = \begin{pmatrix} J_{\lambda_i 1} & 0 & \dots & 0 \\ 0 & J_{\lambda_i 2} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & J_{\lambda_i r_i} \end{pmatrix}$$

onde cada $J_{\lambda_i j}$ é uma matriz elementar de Jordan com valor característico λ_i . Além disso, dentro de cada J_{λ_i} , as dimensões das matrizes $J_{\lambda_i j}$ diminuem à medida que j aumenta. Dizemos que uma matriz J de ordem n que satisfaz todas as condições descritas até agora nesta seção (para certos escalares $\lambda_1, \dots, \lambda_k$) está sob a **forma canônica de Jordan**.

Acabamos de salientar que se T é um operador linear para o qual o polinômio característico se decompõe completamente sobre o corpo de escalares, então existe uma base ordenada de V em relação à qual T é representado por uma matriz que está sob a forma canônica de Jordan. Como nosso interesse está no espaço de matrizes, então representaremos T pela matriz B , e J será a sua matriz de Jordan associada. Fazendo isto, temos as seguintes observações:

1. Todo elemento de J que não esteja na diagonal principal ou imediatamente acima dela é nulo. Na diagonal de J aparecem os k valores próprios distintos $\lambda_1, \dots, \lambda_k$ de B . Além disso, λ_i aparece m_i vezes, sendo m_i a multiplicidade de λ_i como uma raiz do polinômio característico (**multiplicidade algébrica**).

2. Para cada i , a matriz J_{λ_i} é a soma direta de r_i matrizes elementares de Jordan $J_{\lambda_i j}$ com valor próprio λ_i . O número r_i é exatamente a dimensão do espaço dos vetores próprios associados ao valor próprio λ_i (**multiplicidade geométrica**). De fato, r_i é o número de blocos elementares de Jordan que conformam a matriz J_{λ_i} .

3. Para cada i , o primeiro bloco $J_{\lambda_i 1}$ na matriz J_{λ_i} é uma matriz $r_i \times r_i$, sendo r_i a multiplicidade de λ_i como uma raiz do polinômio minimal de B .

Lema 1.1. Sejam A, B, M matrizes complexas, com M não-singular, tal que $M^{-1}AM = B$. Sejam e_1, e_2, v_1, v_2 vetores em $\mathbb{C}^n$ tais que $Me_1 = v_1$ e $Me_2 = v_2$. Suponhamos que existe um número complexo α tal que $(B - \lambda I)e_1 = \alpha e_2$. Então, temos também $(A - \lambda I)v_1 = \alpha v_2$.

Prova. Temos que

$$\begin{aligned}
 (A - \lambda I)v_1 &= (MBM^{-1} - \lambda I)v_1 \\
 &= M(B - \lambda I)M^{-1}v_1 \\
 &= M(B - \lambda I)e_1 \\
 &= M\alpha e_2 \\
 &= \alpha M e_2 \\
 &= \alpha v_2. \blacksquare
 \end{aligned}$$

Lema 1.2. Seja J_λ uma matriz de ordem n dada por

$$J_\lambda = \begin{pmatrix} J_{\lambda 1} & 0 & \dots & 0 \\ 0 & J_{\lambda 2} & \dots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \dots & J_{\lambda r} \end{pmatrix},$$

onde cada $J_{\lambda i}$ ($i = 1, \dots, r$) é uma matriz elementar de Jordan com valor característico λ . Seja e_i ($i = 1, \dots, n$) o i -ésimo vetor canônico em $\mathbb{C}^n$, i.e., e_i é o vetor de coordenadas e_{ij} ($j = 1, \dots, n$) com $e_{ij} = 1$ para $i = j$ e $e_{ij} = 0$ para $i \neq j$. Então

$$(J_\lambda - \lambda I)e_i = e_{i-1} \quad \text{para } i \notin \left\{ \sum_{q=1}^k n_q \mid k = 1, 2, \dots, r \right\}$$

e

$$(J_\lambda - \lambda I)e_i = 0 \quad \text{para } i \in \left\{ \sum_{q=1}^k n_q \mid k = 1, 2, \dots, r \right\},$$

onde n_q é a ordem da matriz $J_{\lambda q}$.

Prova. Como cada $J_{\lambda k}$ é da forma

$$J_{\lambda k} = \begin{pmatrix} \lambda_k & 1 & 0 & \dots & 0 \\ 0 & \lambda_k & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \dots & \lambda_k & 1 \\ 0 & 0 & \dots & 0 & \lambda_k \end{pmatrix},$$

temos que $(J_\lambda - \lambda I)e_i$ reproduz a i -ésima coluna de $(J_\lambda - \lambda I)$, e portanto se

$i \in \left\{ \sum_{q=1}^k n_q \mid k = 1, 2, \dots, r \right\}$ temos que a coluna é nula e senão a coluna é exatamente e_{i-1} . $\blacksquare$

Lema 1.3. Seja A uma matriz complexa de ordem n com valores próprios $\lambda_1, \lambda_2, \dots, \lambda_s$, $\lambda_\alpha \neq \lambda_\beta$ para $\alpha \neq \beta$. Denotemos por m_k a multiplicidade algébrica de λ_k e por r_k a dimensão de $\{v \in \mathbb{C}^n \mid Av = \lambda_k v\}$ para cada $k = 1, 2, \dots, s$. Então, para cada k , $k = 1, 2, \dots, s$, existe uma matriz não-singular M_k de ordem n tal que $M_k^{-1}AM_k = H_k$ com

$$H_k = \begin{pmatrix} H_{nm} & O_{pq} \\ & J_{\lambda_k} \end{pmatrix},$$

onde $m = n - m_k$, $p = n - m_k$, $q = m_k$, H_{nm} é uma matriz complexa $n \times m$, O_{pq} a matriz nula $p \times q$, e

$$J_{\lambda_k} = \begin{pmatrix} J_{\lambda_k 1} & 0 & \dots & 0 \\ 0 & J_{\lambda_k 2} & \dots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \dots & J_{\lambda_k r_k} \end{pmatrix}.$$

Prova. A prova decorre de aplicar o Lema 1.2 de [5] (pp. 13) com $B = A^t$. ■

Agora, consideremos a matriz H_k do Lema 1.3. Denotemos por f_{ki} o vetor canônico e_{n-m_k+i} , $k = 1, 2, \dots, s$ e $i = 1, 2, \dots, m_k$. Dos Lemas 1.2 e 1.3, é imediato que

$$(H_k - \lambda_k I)f_{ki} = f_{ki-1} \quad \text{para } i \notin \left\{ \sum_{q=1}^h n_{kq} \mid h = 1, 2, \dots, r_k \right\}$$

e

$$(H_k - \lambda_k I)f_{ki} = 0 \quad \text{para } i \in \left\{ \sum_{q=1}^h n_{kq} \mid h = 1, 2, \dots, r_k \right\},$$

onde n_{kq} é a ordem da matriz $J_{\lambda_k q}$. Agora, fazemos

$$M_k f_{ki} = v_{ki},$$

$k = 1, 2, \dots, s$ e $i = 1, 2, \dots, m_k$. Do Lema 1.1 obtemos que

$$Av_{ki} = \lambda_k v_{ki} + v_{ki+1} \quad \text{para } i \notin \left\{ \sum_{q=1}^h n_{kq} \mid h = 1, 2, \dots, r_k \right\} \quad (1.3)$$

e

$$Av_{ki} = \lambda_k v_{ki} \quad \text{para } i \in \left\{ \sum_{q=1}^h n_{kq} \mid h = 1, 2, \dots, r_k \right\}. \quad (1.4)$$

Por indução, podemos provar facilmente que os n vetores v_{ki} ($k = 1, 2, \dots, s$ e $i = 1, 2, \dots, m_k$) são linearmente independentes.

Agora, consideremos o sistema $\dot{x} = Ax$ e façamos a seguinte mudança de variável:

$$x = \sum_{k=1}^s \sum_{i=1}^{m_k} y_{ki} v_{ki}.$$

Obtemos:

$$\dot{x} = \sum_{k=1}^s \sum_{i=1}^{m_k} \dot{y}_{ki} v_{ki} \quad \text{e} \quad Ax = \sum_{k=1}^s \sum_{i=1}^{m_k} y_{ki} A v_{ki}.$$

Logo, das equações (1.3) e (1.4) acima, é imediato que

$$\dot{y}_k = J_{\lambda_k} y_k, \quad k = 1, 2, \dots, s,$$

onde

$$y_k = \begin{pmatrix} y_{k1} \\ y_{k2} \\ \vdots \\ y_{km_k} \end{pmatrix}.$$

Daí temos $\dot{y} = Jy$ onde

$$J = \begin{pmatrix} J_{\lambda_1} & 0 & \dots & 0 \\ 0 & J_{\lambda_2} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & J_{\lambda_s} \end{pmatrix},$$

e y é

$$y = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_s \end{pmatrix}.$$

Observamos que a mudança de variável

$$x = \sum_{k=1}^s \sum_{i=1}^{m_k} y_{ki} v_{ki}$$

pode ser escrita da seguinte maneira

$$x = My$$

onde M é a seguinte matriz não-singular $n \times n$

$$M = [v_{11} \dots v_{1m_1} v_{21} \dots v_{2m_2} \dots v_{s1} \dots v_{sm_s}].$$

Assim $J = M^{-1}AM$. Com isto provamos o seguinte teorema:

Teorema 1.2 (Forma Canônica de Jordan). Seja A uma matriz complexa $n \times n$ com valores próprios $\lambda_1, \lambda_2, \dots, \lambda_s$, $\lambda_\alpha \neq \lambda_\beta$ para $\alpha \neq \beta$. Denotemos por m_k a multiplicidade algébrica de λ_k e por r_k a dimensão de $\{v \in \mathbb{C}^n \mid Av = \lambda_k v\}$ para cada $k = 1, 2, \dots, s$. Então existe uma matriz M não-singular tal que $M^{-1}AM = J$ onde

$$J = \begin{pmatrix} J_{\lambda_1} & 0 & \dots & 0 \\ 0 & J_{\lambda_2} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & J_{\lambda_s} \end{pmatrix}$$

e, para $k = 1, 2, \dots, s$, J_{λ_k} é a matriz quadrada de ordem m_k dada por

$$J_{\lambda_k} = \begin{pmatrix} J_{\lambda_k 1} & 0 & \dots & 0 \\ 0 & J_{\lambda_k 2} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & J_{\lambda_k r_k} \end{pmatrix}.$$

Temos também, para $k = 1, 2, \dots, s$, $\sum_{q=1}^{r_k} n_{kq} = m_k$ e $n_{k1} \geq n_{k2} \geq \dots \geq n_{kr_k}$ onde n_{kq} é a ordem do bloco de Jordan $J_{\lambda_k q}$.

1.2 Exemplos

Nesta seção vamos calcular a forma canônica de Jordan das matrizes B que usaremos nos exemplos do Capítulo 2.

Exemplo 1.1. Seja a matriz

$$B = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 2 & 1 & 0 & 0 \\ 1 & 2 & 0 & 0 \end{pmatrix}.$$

Então $\det(B - \lambda I) = \lambda^4 - 4\lambda^2 + 3$ e os valores próprios são: $-\sqrt{3}, -1, 1, \sqrt{3}$.

A matriz P obtida a partir dos vetores próprios é

$$P = \begin{pmatrix} \sqrt{3} & 1 & 1 & \sqrt{3} \\ \sqrt{3} & -1 & -1 & \sqrt{3} \\ -3 & -1 & 1 & 3 \\ -3 & 1 & -1 & 3 \end{pmatrix}.$$

Para computar a forma canônica de Jordan só é preciso fazer $J = P^{-1}BP$. Logo,

$$J = \begin{pmatrix} -\sqrt{3} & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & \sqrt{3} \end{pmatrix}.$$

Exemplo 1.2. Seja a matriz

$$B = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}.$$

Então $\det(B - \lambda I) = \lambda^4$.

Para $\lambda = 0$ temos

$$BX = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \Rightarrow X = \begin{pmatrix} x_1 \\ 0 \\ 0 \\ 0 \end{pmatrix}.$$

Portanto, a multiplicidade geométrica de λ é 1. Para encontrar outros três vetores linearmente independentes com este, usamos a equação (1.3). Assim, temos

$$BX = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} \Rightarrow X = \begin{pmatrix} x_1 \\ 0 \\ 1 \\ 0 \end{pmatrix}.$$

Agora temos

$$BX = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} \Rightarrow X = \begin{pmatrix} x_1 \\ -1 \\ 0 \\ 0 \end{pmatrix}.$$

Seguimos, e temos

$$BX = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} 0 \\ -1 \\ 0 \\ 0 \end{pmatrix} \Rightarrow X = \begin{pmatrix} x_1 \\ 0 \\ 0 \\ -1 \end{pmatrix}.$$

Portanto, uma matriz P gerada a partir destes vetores é

$$P = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix},$$

e a matriz de Jordan é

$$J = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}.$$

Exemplo 1.3. Seja a matriz

$$B = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 2 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}.$$

Então $\det(B - \lambda I) = (\lambda - 1)^2(\lambda + 1)^2$.

Para $\lambda_1 = 1$ temos

$$(B - I)X = \begin{pmatrix} -1 & 0 & 1 & 0 \\ 0 & -1 & 0 & 1 \\ 1 & 2 & -1 & 0 \\ 0 & 1 & 0 & -1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \implies X = \begin{pmatrix} x_1 \\ 0 \\ x_1 \\ 0 \end{pmatrix}.$$

Portanto, a multiplicidade geométrica de λ_1 é 1. Para encontrar um outro vetor linearmente independente com este, usamos a equação (1.3). Assim, temos

$$(B - I)X = \begin{pmatrix} -1 & 0 & 1 & 0 \\ 0 & -1 & 0 & 1 \\ 1 & 2 & -1 & 0 \\ 0 & 1 & 0 & -1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 1 \\ 0 \end{pmatrix} \implies X = \begin{pmatrix} x_1 \\ 1 \\ x_1 + 1 \\ 1 \end{pmatrix}.$$

Para $\lambda_2 = -1$ temos

$$(B + I)X = \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 2 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \implies X = \begin{pmatrix} x_1 \\ 0 \\ -x_1 \\ 0 \end{pmatrix}.$$

Portanto, a multiplicidade geométrica de λ_2 é 1. Para encontrar um outro vetor linearmente independente com este, usamos a equação (1.3). Assim, temos

$$(B + I)X = \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 2 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ -1 \\ 0 \end{pmatrix} \implies X = \begin{pmatrix} x_1 \\ -1 \\ 1 - x_1 \\ 1 \end{pmatrix}.$$

Portanto, a matriz P, gerada a partir dos vetores achados acima, é

$$P = \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & -1 \\ 1 & 1 & -1 & 1 \\ 0 & 1 & 0 & 1 \end{pmatrix},$$

e a matriz de Jordan é

$$J = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 1 \\ 0 & 0 & 0 & -1 \end{pmatrix}.$$

Exemplo 1.4. Seja a matriz

$$B = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -\frac{1}{4} & 0 & 1 & 0 \\ 0 & -\frac{1}{4} & 0 & 1 \end{pmatrix}.$$

Então $\det(B - \lambda I) = (\lambda - \frac{1}{2})^4$.

Para $\lambda = \frac{1}{2}$ temos

$$(B - \tfrac{1}{2}I)X = \begin{pmatrix} -\frac{1}{2} & 0 & 1 & 0 \\ 0 & -\frac{1}{2} & 0 & 1 \\ -\frac{1}{4} & 0 & \frac{1}{2} & 0 \\ 0 & -\frac{1}{4} & 0 & \frac{1}{2} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \implies X = \begin{pmatrix} 2x_3 \\ 2x_4 \\ x_3 \\ x_4 \end{pmatrix}.$$

Portanto, a multiplicidade geométrica de λ é 2. Para encontrar os vetores linearmente independentes com aqueles da base

$$X_1 = \begin{pmatrix} 2 \\ 0 \\ 1 \\ 0 \end{pmatrix}, \quad X_2 = \begin{pmatrix} 0 \\ 2 \\ 0 \\ 1 \end{pmatrix},$$

aplicamos a equação (1.3). Desta forma, temos para X_1

$$(B - \tfrac{1}{2}I)X = \begin{pmatrix} -\frac{1}{2} & 0 & 1 & 0 \\ 0 & -\frac{1}{2} & 0 & 1 \\ -\frac{1}{4} & 0 & \frac{1}{2} & 0 \\ 0 & -\frac{1}{4} & 0 & \frac{1}{2} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} 2 \\ 0 \\ 1 \\ 0 \end{pmatrix} \implies X = \begin{pmatrix} 2x_3 - 4 \\ 2x_4 \\ x_3 \\ x_4 \end{pmatrix}.$$

Para X_2 temos

$$(B - \frac{1}{2}I)X = \begin{pmatrix} -\frac{1}{2} & 0 & 1 & 0 \\ 0 & -\frac{1}{2} & 0 & 1 \\ -\frac{1}{4} & 0 & \frac{1}{2} & 0 \\ 0 & -\frac{1}{4} & 0 & \frac{1}{2} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} 0 \\ 2 \\ 0 \\ 1 \end{pmatrix} \Rightarrow X = \begin{pmatrix} 2x_3 \\ 2x_4 - 4 \\ x_3 \\ x_4 \end{pmatrix}.$$

Portanto, uma matriz P gerada a partir destes vetores é

$$P = \begin{pmatrix} 2 & 2 & 0 & 0 \\ 0 & 0 & 2 & 2 \\ 1 & 3 & 0 & 0 \\ 0 & 0 & 1 & 3 \end{pmatrix},$$

e a matriz de Jordan tem a seguinte forma

$$J = \begin{pmatrix} \frac{1}{2} & 1 & 0 & 0 \\ 0 & \frac{1}{2} & 0 & 0 \\ 0 & 0 & \frac{1}{2} & 1 \\ 0 & 0 & 0 & \frac{1}{2} \end{pmatrix}.$$

1.3 Equações Diferenciais Ordinárias

Nesta seção revisaremos alguns conceitos básicos de Equações Diferenciais Ordinárias que são usados no presente trabalho.

Seja $A(t)$ uma função matricial de ordem n e X um vetor em $\mathbb{C}^n$ que depende de t . Chamamos a equação

$$\dot{X} = A(t)X \tag{1.5}$$

de **sistema homogêneo** de ordem n .

Definição 1.1. Sejam $u_1(t), u_2(t), \dots, u_k(t)$ funções vetoriais definidas para todo t . Dizermos que $u_1(t), u_2(t), \dots, u_k(t)$ são linearmente dependentes se existem números complexos $c_1, c_2, \dots, c_k$, não todos nulos, tais que $\sum_{i=1}^k c_i u_i(t) \equiv 0$. Se esta equação vale só quando

$c_1 = c_2 = \dots = c_k = 0$, dizemos que as funções são linearmente independentes.

Proposição 1.1. Qualquer combinação linear de soluções de (1.5) é também uma solução.

Prova. Seja

$$u(t) = \sum_{i=1}^k c_i u_i(t),$$

onde $u_1(t), u_2(t), \dots, u_k(t)$ são soluções de (1.5), e $c_1, c_2, \dots, c_k$ são números. Derivando, obtemos

$$\begin{aligned} \dot{u}(t) &= \sum_{i=1}^k c_i \dot{u}_i(t) \\ &= \sum_{i=1}^k c_i A(t) u_i(t) \\ &= A(t) \sum_{i=1}^k c_i u_i(t) \\ &= A(t) u(t). \blacksquare \end{aligned}$$

Teorema 1.3. Sejam $u_1(t), u_2(t), \dots, u_n(t)$ n soluções linearmente independentes de (1.5). Então, para cada solução $u(t)$, existe um único conjunto de números $c_1, c_2, \dots, c_n$, tal que $u(t) = \sum_{i=1}^n c_i u_i(t)$ para todo t .

Prova. Ver [11], pp. 150-151. ■

Definição 1.2. Dizemos que uma matriz C de ordem n é **matriz solução** de (1.5) se cada uma das suas colunas são soluções de (1.5). Uma matriz solução cujas colunas são linearmente independentes é denominada **matriz fundamental** de (1.5).

Proposição 1.2. Seja $W(t)$ uma função matricial de ordem n . Então $W(t)$ é uma matriz solução, se e somente se,

$$\dot{W}(t) = A(t)W(t)$$

para todo t .

Prova. Representemos as colunas de $W(t)$ por $u_1(t), u_2(t), \dots, u_n(t)$. Suponhamos que $W(t)$ é uma matriz solução, isto é, que $\dot{u}_i(t) = A(t)u_i(t)$, $i = 1, \dots, n$. A coluna i de $A(t)W(t)$ é $A(t)u_i(t)$. Logo, as matrizes $\dot{W}$ e $A(t)W(t)$ tem as mesmas colunas e são portanto iguais. Reciprocamente, se $\dot{W}(t) = A(t)W(t)$, igualando as colunas, obtemos $\dot{u}_i(t) = A(t)u_i(t)$, e portanto $W(t)$ é uma matriz solução. ■

Observação 1.1.

1. Podemos escrever a Proposição 1.1 da seguinte forma: Se $W(t)$ é uma matriz solução de (1.5), e c é um vetor, então $W(t)c$ é uma solução.
2. O Teorema 1.3 fica como segue: Se $W(t)$ é uma matriz fundamental, e u é uma solução, então existe um único vetor c tal que $u(t) = W(t)c$.

Capítulo 2

Equações Polinomiais Matriciais

2.1 Definições

Definição 2.1. Chama-se **equação polinomial matricial** a uma equação definida no espaço das matrizes quadradas de ordem n , $n > 1$, cuja forma é:

$$A_0 X^m + A_1 X^{m-1} + \dots + A_{m-1} X + A_m = 0, \quad (2.1)$$

onde $m > 1$ e os coeficientes $\{A_i : i = 0, \dots, m\}$ são matrizes quadradas de ordem n . Se A_0 for uma matriz não-nula, dizemos que a equação é de grau m .

Neste capítulo consideramos equações onde o coeficiente A_0 é uma matriz não-singular. Podemos então considerar A_0 como a matriz identidade. Associada à equação (2.1) existe uma equação diferencial que tem a seguinte forma:

$$\frac{d^m Y}{dt^m} + A_1 \frac{d^{m-1} Y}{dt^{m-1}} + \dots + A_{m-1} \frac{dY}{dt} + A_m Y = 0, \quad (2.2)$$

onde Y é uma função matricial, desconhecida, de ordem n . Esta associação é dada pela seguinte proposição:

Proposição 2.1. A matriz X_0 é uma solução de (2.1) se e somente se Y_0 , solução de

$$\frac{dY}{dt} = X_0 Y, \quad Y(0) = I, \quad (2.3)$$

é uma solução de (2.2).

Prova.

1. Suponhamos que X_0 é solução de (2.1).

Então X_0 satisfaz

$$A_0 X_0^m + A_1 X_0^{m-1} + \dots + A_{m-1} X_0 + A_m = 0.$$

Seja Y_0 uma solução de (2.3). Temos que Y_0 satisfaz

$$\frac{d^k Y_0}{dt^k} = X_0^k Y_0, \quad (k \geq 1).$$

Logo,

$$\frac{d^m Y_0}{dt^m} + A_1 \frac{d^{m-1} Y_0}{dt^{m-1}} + \dots + A_{m-1} \frac{d Y_0}{dt} + A_m Y_0 = (A_0 X_0^m + A_1 X_0^{m-1} + \dots + A_{m-1} X_0 + A_m) Y_0 = 0$$

e segue que Y_0 é solução de (2.2).

2. Suponhamos que Y_0 , solução de (2.3), é solução de (2.2).

Temos que $Y_0 = \exp(tX_0)^1$. Substituindo em (2.2) e fatorando temos a seguinte equação:

$$(A_0 X_0^m + A_1 X_0^{m-1} + \dots + A_{m-1} X_0 + A_m) \exp(tX_0) = 0.$$

Como $\exp(tX_0)$ é uma matriz invertível, temos que X_0 é solução de (2.1). ■

Seja Y_j a coluna j da matriz Y . Então, fazendo

$$\begin{cases} Z_{1j} = Y_j \\ Z_{2j} = \frac{dZ_{1j}}{dt} \\ \vdots \\ Z_{mj} = \frac{dZ_{m-1j}}{dt} \end{cases} \quad (2.4)$$

temos que, associado à equação (2.2), está o sistema

$$\frac{dZ}{dt} = BZ, \quad (2.5)$$

onde B é a matriz

$$B = \begin{pmatrix} 0 & I_n & 0 & \dots & 0 \\ 0 & 0 & I_n & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & I_n \\ -A_m & -A_{m-1} & -A_{m-2} & \dots & -A_1 \end{pmatrix} \quad (2.6)$$

¹Veja [3](pp. 57-58).

e Z é uma função matricial de ordem $mn \times n$, cujas colunas estão dadas por (2.4). Vamos considerar, na equação (2.5), a matriz Z como sendo quadrada de ordem mn .

Definição 2.2. A matriz B é conhecida como **matriz companheira por blocos** associada à equação (2.1).

Definição 2.3. Dizemos que um bloco de Jordan R_i está contido num bloco de Jordan J_h se e somente se $r_i = j_h$ e $m_i \leq l_h$, onde r_i, j_h são os valores próprios de R_i e J_h respectivamente, e m_i, l_h são as respectivas multiplicidades algébricas. Denotamos este fato por $R_i \subset J_h$.

Definição 2.4. Dizemos que uma matriz de Jordan R está contida numa matriz de Jordan J se e somente se para todo bloco de Jordan de R existe um bloco de Jordan de J que o contém.

Sejam as equações diferenciais:

$$\frac{dN}{dt} = RN,$$

$$\frac{dV}{dt} = JV,$$

onde R e J são matrizes de Jordan.

Definição 2.5. Dizemos que $N \subset V$ se e somente se $R \subset J$.

Definição 2.6. Dizemos que um bloco N_i da matriz N está contido num bloco V_h da matriz V se e somente se $R_i \subset J_h$.

2.2 Existência de Soluções

Nesta seção vamos determinar as condições necessárias e suficientes para a existência de soluções da equação (2.1), quando o coeficiente A_0 é a matriz identidade. Estas condições de existência estão relacionadas à verificação de certas propriedades da matriz de Jordan associada à matriz companheira por blocos da equação (2.1).

Antes de estudarmos os diferentes casos que apresenta a matriz de Jordan J associada à matriz B , encontraremos algumas novas equações associadas às equações anteriores, para chegarmos mais facilmente aos resultados desejados.

Na equação (2.5) fazemos $Z=PV$, e obtemos:

$$\frac{dV}{dt} = JV, \quad (2.7)$$

onde

$$J = P^{-1}BP$$

é a matriz canônica de Jordan de B . Aqui estamos considerando as matrizes de blocos

$$J = \begin{pmatrix} J_1 & 0 & \dots & 0 \\ 0 & J_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & J_k \end{pmatrix}, \quad V = \begin{pmatrix} V_1 & 0 & \dots & 0 \\ 0 & V_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & V_k \end{pmatrix},$$

onde

$$J_i = \begin{pmatrix} j_i & 1 & \dots & 0 \\ 0 & j_i & \ddots & 0 \\ \vdots & \vdots & \ddots & 1 \\ 0 & 0 & \dots & j_i \end{pmatrix}, \quad V_i = \begin{pmatrix} 1 & t & \dots & t^{l_i-1}/(l_i-1)! \\ 0 & 1 & \dots & t^{l_i-2}/(l_i-2)! \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 1 \end{pmatrix} e^{j_i t},$$

e V_i e J_i são matrizes de ordem l_i , $i = 1, \dots, k$.

Além disso, suponhamos que $Y = QN$ em (2.3). Temos então que

$$\frac{dN}{dt} = RN, \quad (2.8)$$

onde R é a matriz canônica de Jordan de X_0 , $R = Q^{-1}X_0Q$,

$$R = \begin{pmatrix} R_1 & 0 & \dots & 0 \\ 0 & R_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & R_s \end{pmatrix}, \quad R_i = \begin{pmatrix} r_i & 1 & \dots & 0 \\ 0 & r_i & \ddots & 0 \\ \vdots & \vdots & \ddots & 1 \\ 0 & 0 & \dots & r_i \end{pmatrix}$$

e R_i são matrizes de ordem m_i para todo $i = 1, \dots, s$.

Como $\det Q \neq 0$, $Y = QN$ é uma solução fundamental de (2.3). De forma similar, $Z = PV$ é uma solução fundamental de (2.7). Se X_0 é uma solução de (2.1), então

$$G = \begin{pmatrix} Y \\ X_0 Y \\ \vdots \\ X_0^{m-1} Y \end{pmatrix} \quad (2.9)$$

é uma matriz $mn \times n$ que satisfaz

$$\frac{dG}{dt} = BG, \quad (2.10)$$

e existe uma matriz constante C^2 de ordem $mn \times n$ tal que

$$G = PVC. \quad (2.11)$$

²Pela Observação 1.1.

Vamos introduzir agora uma notação que usaremos no restante deste capítulo. Dada uma matriz P e M e N dois conjuntos de índices, denotamos:

por $P[M;]$ a matriz obtida de P tomando as linhas i tais que $i \in M$, e todas as colunas;

por $P[; N]$ a matriz obtida de P tomando todas as linhas e as colunas j tais que $j \in N$;

por $P[M; N]$ a matriz obtida de P tomando as linhas i tais que $i \in M$ e as colunas j tais que $j \in N$.

Proposição 2.2. Se existe uma solução X_0 de (2.1), então $R \subset J$.

Prova. Se existe uma solução X_0 de (2.1) então existe uma função matricial não-singular, solução de (2.2), que satisfaz $\frac{dY}{dt} = X_0 Y$. Seja G definida como acima. Então, por (2.11),

$$Y = P[\{1, \dots, n\};]VC.$$

Isto é, cada coluna de Y é uma combinação linear das colunas de $P[\{1, \dots, n\};]V$. De $Y = QN$ temos que $P^{-1}[\{1, \dots, n\}]QN = VC$. Pela forma especial que têm N e V , temos que cada valor próprio de R é igual a algum valor próprio de J . Portanto $R \subset J$. ■

Agora, discutiremos os casos que se apresentam com respeito à matriz J .

Caso 1. Suponhamos que J é uma matriz diagonal e os seus valores próprios $j_1, \dots, j_{mn}$ são diferentes.

I. Se existe uma solução X_0 de (2.1), então existe uma função matricial não-singular Y tal que $Z = PV$. Sejam $Q = (q_{ij})$, $i, j = 1, \dots, n$ e $P = (p_{ik})$, $i, k = 1, \dots, mn$. Notemos que G , como em (2.9), satisfaz (2.10). Pela proposição anterior $R \subset J$. Desta forma R é uma matriz diagonal e os seus valores próprios $r_1, \dots, r_n$ são diferentes. Daí existem inteiros positivos $k_1, \dots, k_n$, $1 \leq k_1, \dots, k_n \leq mn$ tais que $r_1 = j_{k_1}, \dots, r_n = j_{k_n}$. Como existe uma matriz C tal que $P[\{1, \dots, n\};]VC = QN$, temos que existem constantes

não-nulas $c_1, c_2, \dots, c_n$ tais que

$$\begin{pmatrix} q_{11} & q_{12} & \dots & q_{1n} \\ q_{21} & q_{22} & \dots & q_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ q_{n1} & q_{n2} & \dots & q_{nn} \end{pmatrix} = \begin{pmatrix} c_1 & 0 & \dots & 0 \\ 0 & c_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \dots & \dots & c_n \end{pmatrix} \begin{pmatrix} p_{1k_1} & p_{1k_2} & \dots & p_{1k_n} \\ p_{2k_1} & p_{2k_2} & \dots & p_{2k_n} \\ \vdots & \vdots & \ddots & \vdots \\ p_{nk_1} & p_{nk_2} & \dots & p_{nk_n} \end{pmatrix}.$$

Em particular, podemos tomar $c_1 = 1, \dots, c_n = 1$, porque a matriz Q não é única. Seja T a matriz $mn \times mn$ tal que $P[\{1, \dots, n\};]T$ tem as primeiras n colunas linearmente independentes. Então temos que $Q = (P[\{1, \dots, n\};]T)[; \{1, \dots, n\}]$, $R = (T^{-1}JT)[\{1, \dots, n\}, \{1, \dots, n\}]$, $N = (T^{-1}VT)[\{1, \dots, n\}, \{1, \dots, n\}]$ satisfaz (2.8) e $X_0 = QRQ^{-1}$ é uma solução da equação (2.1).

II. Como $\det P \neq 0$, o posto da matriz $P[\{1, \dots, n\};]$ é n , portanto é possível encontrar uma matriz T tal que as primeiras n colunas de $P[\{1, \dots, n\};]T$ sejam linearmente independentes, e portanto existe pelo menos uma solução de (2.1). Também, como queremos n elementos de J , e J tem mn valores próprios, então o número máximo de soluções de (2.1) é

$$C(mn, n) = \frac{(mn)!}{n!(mn-n)!}.$$

Exemplo 2.1. Seja a equação

$$X^2 - A = 0 \tag{2.12}$$

onde

$$A = \begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix}.$$

Temos que

$$B = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 2 & 1 & 0 & 0 \\ 1 & 2 & 0 & 0 \end{pmatrix}.$$

Temos, pelo Exemplo 1.1, que

$$P = \begin{pmatrix} \sqrt{3} & 1 & 1 & \sqrt{3} \\ \sqrt{3} & -1 & -1 & \sqrt{3} \\ -3 & -1 & 1 & 3 \\ -3 & 1 & -1 & 3 \end{pmatrix},$$

e a matriz de Jordan é

$$J = \begin{pmatrix} -\sqrt{3} & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & \sqrt{3} \end{pmatrix}.$$

Portanto

$$Q = \begin{pmatrix} \sqrt{3} & 1 \\ \sqrt{3} & -1 \end{pmatrix} \quad \text{e} \quad R = \begin{pmatrix} -\sqrt{3} & 0 \\ 0 & -1 \end{pmatrix}.$$

Então uma solução para a equação (2.12) é

$$X_0 = \begin{pmatrix} \sqrt{3} & 1 \\ \sqrt{3} & -1 \end{pmatrix} \begin{pmatrix} -\sqrt{3} & 0 \\ 0 & -1 \end{pmatrix} \begin{pmatrix} \frac{\sqrt{3}}{6} & \frac{\sqrt{3}}{6} \\ \frac{1}{2} & -\frac{1}{2} \end{pmatrix} = \begin{pmatrix} -\frac{\sqrt{3}+1}{2} & -\frac{\sqrt{3}-1}{2} \\ -\frac{\sqrt{3}-1}{2} & -\frac{\sqrt{3}+1}{2} \end{pmatrix}.$$

Caso 2. Suponhamos que os valores próprios de J , que pertencem a diferentes blocos de Jordan, são diferentes (i.e, existem k valores próprios de J).

Se existe uma solução X_0 de (2.1), então temos que $N \subset V$ (pela proposição anterior e Definição 2.5). Além disso, $r_1, \dots, r_s$ são diferentes porque $j_1, \dots, j_k$ são diferentes. Não é difícil mostrar que existe uma matriz de permutação T das colunas $k_1, \dots, k_n$ de P tal que $J' = T^{-1}JT$ tem a seguinte forma:

$$J' = \begin{pmatrix} J'_{11} & J'_{12} \\ J'_{21} & J'_{22} \end{pmatrix} \tag{2.13}$$

onde

$$J'_{11} = \begin{pmatrix} J'_1 & 0 & \dots & 0 \\ 0 & J'_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & J'_s \end{pmatrix},$$

$$J'_{22} = \begin{pmatrix} J'_{s+1} & 0 & \dots & 0 \\ 0 & J'_{s+2} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & J'_{k'} \end{pmatrix};$$

e $V' = T^{-1}VT$ tem a seguinte forma:

$$V' = \begin{pmatrix} V'_{11} & V'_{12} \\ V'_{21} & V'_{22} \end{pmatrix} \quad (2.14)$$

onde

$$V'_{11} = \begin{pmatrix} V'_1 & 0 & \dots & 0 \\ 0 & V'_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & V'_s \end{pmatrix},$$

$$V'_{22} = \begin{pmatrix} V'_{s+1} & 0 & \dots & 0 \\ 0 & V'_{s+2} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & V'_{k'} \end{pmatrix},$$

com $J'_i = R_i$, $V'_i = N_i$, $i = 1, \dots, s$. Supondo $Y = QN$ e G a matriz dada por (2.9), obtemos que G satisfaz (2.10). Por outro lado, temos:

$$P'V' = \begin{pmatrix} P'[\{1, \dots, n\};]N & M_1 \\ M_2 & M_3 \end{pmatrix}$$

onde $P' = PT$. Como $P'[\{1, \dots, n\};]V'C = QN$, temos que existem constantes $c_1, \dots, c_n$ tais que

$$\begin{pmatrix} q_{11} & q_{12} & \dots & q_{1n} \\ q_{21} & q_{22} & \dots & q_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ q_{n1} & q_{n2} & \dots & q_{nn} \end{pmatrix} = \begin{pmatrix} c_1 & 0 & \dots & 0 \\ 0 & c_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \dots & \dots & c_n \end{pmatrix} \begin{pmatrix} p_{1k_1} & p_{1k_2} & \dots & p_{1k_n} \\ p_{2k_1} & p_{2k_2} & \dots & p_{2k_n} \\ \vdots & \vdots & \ddots & \vdots \\ p_{nk_1} & p_{nk_2} & \dots & p_{nk_n} \end{pmatrix}.$$

Se $P'[\{1, \dots, n\}; \{1, \dots, n\}]$ é invertível, então Q dada anteriormente é tal que $X_0 = QRQ^{-1}$ é solução de (2.1). Por outro lado, se $P'[\{1, \dots, n\}; \{1, \dots, n\}]$ não é invertível para toda permutação ordenada³ T tal que $P' = PT$, então a equação (2.1) não tem solução.

Exemplo 2.2. Seja a equação matricial

$$X^2 + A = 0, \quad (2.15)$$

onde

$$A = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}.$$

Temos que

$$B = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix},$$

(veja o Exemplo 1.2). Uma solução fundamental de (2.5) é

$$Z = \begin{pmatrix} 1 & t & t^2/2! & t^3/3! \\ 0 & 0 & -1 & -t \\ 0 & 1 & t & t^2/2! \\ 0 & 0 & 0 & -1 \end{pmatrix}.$$

³i.e., os índices j permutados que pertencem às colunas de um bloco de Jordan são consecutivos.

Temos também que $Z = PV$, com

$$P = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix},$$

$$V = \begin{pmatrix} 1 & t & t^2/2! & t^3/3! \\ 0 & 1 & t & t^2/2! \\ 0 & 0 & 1 & t \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

Então temos que $Q = P[\{1, 2\}; \{1, 2\}]$ e $\det Q = 0$, o que significa que a equação (2.15) não tem solução.

Exemplo 2.3. Seja a equação matricial

$$X^2 - A = 0, \tag{2.16}$$

onde

$$A = \begin{pmatrix} 1 & 2 \\ 0 & 1 \end{pmatrix}.$$

Temos que

$$B = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 2 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}.$$

Temos, pelo Exemplo 1.3, que

$$P = \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & -1 \\ 1 & 1 & -1 & 1 \\ 0 & 1 & 0 & 1 \end{pmatrix},$$

e a matriz de Jordan J é

$$\begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 1 \\ 0 & 0 & 0 & -1 \end{pmatrix}.$$

Portanto

$$Q = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \quad \text{e} \quad R = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}.$$

Daí, uma solução da equação (2.16) é

$$X_0 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}.$$

Caso 3. Suponhamos que alguns valores próprios de J que pertencem a blocos de Jordan diferentes podem ser repetidos.

I. Se existe uma solução X_0 de (2.1), então temos que $N \subset V$. Se $Y = QN$ satisfaz (2.3) e G está definida por (2.9), então G satisfaz (2.10). E existe uma matriz constante C , $mn \times n$, tal que $P[\{1, \dots, n\};]VC = QN$ e $Z = PV$ é uma matriz fundamental de $\frac{dZ}{dt} = BZ$.

II. Se temos uma matriz constante C , $mn \times n$, e uma matriz Q , $n \times n$, com $\det Q \neq 0$, tal que

$$PVC = \begin{pmatrix} QN \\ Z_2 \\ \vdots \\ Z_m \end{pmatrix},$$

onde N satisfaz $\frac{dN}{dt} = RN$, R sendo uma matriz canônica de Jordan, então $X_0 = QRQ^{-1}$ satisfaz (2.1).

III. Como a matriz C é de ordem $mn \times n$, o número de parâmetros nas soluções não é maior do que mn^2 .

Exemplo 2.4. Seja a equação matricial

$$X^2 - I_2 X + \begin{pmatrix} \frac{1}{4} & 0 \\ 0 & \frac{1}{4} \end{pmatrix} = 0. \quad (2.17)$$

Então

$$B = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -\frac{1}{4} & 0 & 1 & 0 \\ 0 & -\frac{1}{4} & 0 & 1 \end{pmatrix}.$$

Temos que (veja o Exemplo 1.4)

$$\begin{aligned} Z = PV &= \begin{pmatrix} 2 & 2 & 0 & 0 \\ 0 & 0 & 2 & 2 \\ 1 & 3 & 0 & 0 \\ 0 & 0 & 1 & 3 \end{pmatrix} \begin{pmatrix} 1 & t & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & t \\ 0 & 0 & 0 & 1 \end{pmatrix} e^{\frac{1}{2}t} \\ &= \begin{pmatrix} 2 & 2+2t & 0 & 0 \\ 0 & 0 & 2 & 2+2t \\ 1 & 3+t & 0 & 0 \\ 0 & 0 & 1 & 3+t \end{pmatrix} e^{\frac{1}{2}t}. \end{aligned}$$

Se definimos C como

$$C = \begin{pmatrix} \frac{1}{2}q_{11} & (q_{12} - q_{11})/2 \\ 0 & \frac{1}{2}q_{11} \\ \frac{1}{2}q_{21} & (q_{22} - q_{21})/2 \\ 0 & \frac{1}{2}q_{21} \end{pmatrix},$$

temos que

$$P[\{1, 2\};]VC = \begin{pmatrix} q_{11} & q_{12} \\ q_{21} & q_{22} \end{pmatrix} \begin{pmatrix} 1 & t \\ 0 & 1 \end{pmatrix} e^{\frac{1}{2}t} = QN = Y.$$

Então, Y satisfaz $\frac{dY}{dt} = X_0 Y$, onde

$$X_0 = Q \begin{pmatrix} \frac{1}{2} & 1 \\ 0 & \frac{1}{2} \end{pmatrix} Q^{-1}.$$

Daí X_0 é uma solução de (2.17).

Teorema 2.1. Sejam a equação polinomial matricial

$$X^m + A_1 X^{m-1} + \dots + A_{m-1} X + A_m = 0, \quad (2.18)$$

e B a sua matriz companheira por blocos. Seja J a forma canônica de Jordan de B . Então temos:

1. Se J é diagonalizável então a equação (2.18) tem solução.
2. Se os valores próprios de J , que pertencem a diferentes blocos de Jordan, são diferentes, então a equação (2.18) pode ou não ter solução.
3. Se alguns valores próprios de J que pertencem a blocos de Jordan diferentes, são iguais, então a equação (2.18) tem solução.

Prova. Decorre dos casos acima. ■

2.3 Algoritmo

Nesta seção, vamos fornecer o algoritmo que encontra uma solução da Equação (2.18) sempre que existir.

Em primeiro lugar, precisamos saber como diferenciar os casos que se apresentam com respeito à matriz de Jordan J . Para isto só precisamos dar uma olhada no número de valores próprios diferentes e a multiplicidade geométrica que eles tem na forma canônica de Jordan. Fazemos isto da seguinte maneira:

- (i) Se o número de valores próprios diferentes é exatamente a ordem da matriz J , então estamos no caso 1, caso contrário estamos no caso 2 ou 3.
- (ii) Se achamos a soma das multiplicidades geométricas g_i de cada valor próprio λ_i e obtemos como resultado o número de valores próprios, isto quer dizer que os valores próprios tem multiplicidade geométrica igual a 1 e, em consequencia, para cada bloco de Jordan diferente temos um valor próprio diferente (caso 2).
- (iii) Se as duas situações acima não acontecem então estamos no caso 3.

Algoritmo 2.1

1. Ler as matrizes A_i de ordem n ($i = 1, \dots, m$).
2. Construir a matriz B a partir das matrizes A_i , como em (2.6).
3. Calcular os valores próprios λ_k de B .
4. Armazenar em nv o número de valores próprios.
5. Calcular os vetores próprios associados a cada valor próprio de B e armazenar as multiplicidades geométricas no vetor mg .
6. Construir as matrizes P e J a partir da informação dada pelos vetores e valores próprios.
7. Calcular $mg_{total} = \sum_{i=1}^{nv} mg_i$.
8. Se ($nv = m.n$)

V: {Caso 1.}

8.1. Escolha as n primeiras colunas linearmente independentes de $P[\{1, \dots, n\};]$ e seja I o conjunto que contém os índices destas n colunas.

8.2. $Q[\{1, \dots, n\}; \{1, \dots, n\}] = P[\{1, \dots, n\}; I]$.

8.3. $R[\{1, \dots, n\}; \{1, \dots, n\}] = J[I, I]$.

8.4. $X_0 = QRQ^{-1}$.

F: {Casos 2 e 3.}

8.1. Se ($mg_{total} = nv$)

V: {Caso 2.}

8.1.1. fim= falso.

8.1.2. $i = 1$.

8.1.3. Enquanto (fim=falso) e ($i \leq nv$) faça:

8.1.3.1. Procure os vetores linearmente independentes nas colunas de $P[\{1, \dots, n\};]$ (esta procura deve ser feita em ordem, isto é, os vetores

linearmente independentes devem ser adjacentes com respeito a cada sub-bloco de Jordan).

8.1.3.2. Seja vli uma variável que contém o número de vetores linearmente independentes achados até o momento.

8.1.3.3. Se ($vli = n$)

V: 8.1.3.3.1. fim = verdadeiro.

F: 8.1.3.3.1. $i = i + 1$.

8.1.4. Se (fim = verdadeiro)

V: {O problema tem solução.}

8.1.4.1. $Q[\{1, \dots, n\}; \{1, \dots, n\}] = P[\{1, \dots, n\}; I]$.

8.1.4.2. $R[\{1, \dots, n\}; \{1, \dots, n\}] = J[I, I]$.

8.1.4.3. $X_0 = QRQ^{-1}$.

F: {Caso 3.}

8.1.1. fim = falso.

8.1.2. $i = 1$.

8.1.3. Enquanto (fim=falso) e ($i \leq nv$) faça:

8.1.3.1. Procure os vetores linearmente independentes nas colunas de $P[\{1, \dots, n\};]$ (esta procura deve ser feita em ordem, isto é, os vetores linearmente independentes devem ser adjacentes com respeito a cada sub-bloco de Jordan).

8.1.3.2. Seja vli uma variável que contém o número de vetores linearmente independentes achados até o momento.

8.1.3.3. Se ($vli = n$)

V: 8.1.3.3.1. fim = verdadeiro.

F: 8.1.3.3.1. $i = i + 1$.

8.1.4. $Q[\{1, \dots, n\}; \{1, \dots, n\}] = P[\{1, \dots, n\}; I]$.

8.1.5. $R[\{1, \dots, n\}; \{1, \dots, n\}] = J[I, I]$.

8.1.6. $X_0 = QRQ^{-1}$.

Capítulo 3

Um Método de Ponto Fixo

Neste capítulo denotaremos por $\mathcal{M}(n, \mathbb{C})$ o espaço das matrizes de ordem n com elementos em $\mathbb{C}$. Também denotaremos por $\|\cdot\|$ uma norma do espaço das matrizes tal que $\|I\| = 1$, onde I é a matriz identidade.

3.1 Existência de Soluções

Nesta seção obtemos um teorema de ponto fixo que diz quando a equação (2.1) tem solução e por meio da prova deste teorema obtemos um método para encontrar tais soluções.

Definição 3.1. Dizemos que uma norma $\|\cdot\|$ no espaço das matrizes $\mathcal{M}(n, \mathbb{C})$ é **multiplicativa** se $\|AB\| \leq \|A\| \|B\|$ para toda A e $B \in \mathcal{M}(n, \mathbb{C})$.

Proposição 3.1. Sejam X, Y duas matrizes complexas de ordem n . Então, para qualquer norma multiplicativa no espaço das matrizes e para todo número inteiro $s \geq 1$, temos

$$\|X^s - Y^s\| \leq \sum_{j=0}^{s-1} \|X\|^{s-1-j} \|Y\|^j \|X - Y\|. \quad (3.1)$$

Prova. A prova é feita por indução sobre s .

$s = 1$: É óbvio.

Suponhamos que para $s = p$ vale a desigualdade.

Provemos para $s = p + 1$.

$$\begin{aligned}
 \|X^{p+1} - Y^{p+1}\| &= \|X^p X - Y^p X + Y^p X - Y^p Y\| \\
 &\leq \|(X^p - Y^p)X\| + \|Y^p(X - Y)\| \\
 &\leq \|X^p - Y^p\| \|X\| + \|Y\|^p \|X - Y\| \\
 &\leq \|X\| \sum_{j=0}^{p-1} \|X\|^{p-1-j} \|Y\|^j \|X - Y\| + \|Y\|^p \|X - Y\| \\
 &= \sum_{j=0}^p \|X\|^{p-j} \|Y\|^j \|X - Y\|. \blacksquare
 \end{aligned}$$

Proposição 3.2. Sejam um número real a ($0 \leq a < 1$) e n um inteiro ≥ 1 . Então

$$\sum_{i=1}^n a^i i = \frac{a^{n+1}(na - n - 1) + a}{(1 - a)^2}. \quad (3.2)$$

Prova. A prova é feita por indução sobre n .

$n = 1$: É óbvio.

Suponhamos que para $n = p$ a igualdade é satisfeita.

Vejamos para $n = p + 1$.

$$\begin{aligned}
 \sum_{i=1}^{p+1} a^i i &= \sum_{i=1}^p a^i i + a^{p+1}(p + 1) \\
 &= \frac{a^{p+1}(pa - p - 1) + a}{(1 - a)^2} + a^{p+1}(p + 1) \\
 &= \frac{a^{p+1}(p + 1)(a - 1)^2 + a^{p+1}(pa - p - 1) + a}{(1 - a)^2} \\
 &= \frac{a^{p+1}[(p + 1)a^2 - 2(p + 1)a + (p + 1) + pa - p - 1] + a}{(1 - a)^2} \\
 &= \frac{a^{p+2}[(p + 1)a - p - 2] + a}{(1 - a)^2}. \blacksquare
 \end{aligned}$$

Na fórmula (3.2) é fácil ver que se $n \rightarrow \infty$ temos

$$\sum_{i=1}^{\infty} a^i i = \frac{a}{(1-a)^2}. \quad (3.3)$$

Proposição 3.3. Seja $\|\cdot\|$ uma norma multiplicativa no espaço das matrizes complexas de ordem n , e seja A uma matriz tal que $\|A\| < 1$. Então

$$(A + I)^{-1} = \sum_{i=0}^{\infty} (-1)^i A^i.$$

Prova. Como $\|A\| < 1$ então a soma dada acima é convergente. Também, temos

$$(A + I) \sum_{i=0}^{\infty} (-1)^i A^i = \sum_{i=0}^{\infty} (-1)^i A^{i+1} + \sum_{i=0}^{\infty} (-1)^i A^i = I. \blacksquare$$

Teorema 3.1. Sejam V um espaço de Banach, $\mathcal{G} : V \mapsto V$ uma contração¹ e $x_0 \in V$. Então $\mathcal{G}$ tem um único ponto fixo x^* e $x^* = \lim_{m \rightarrow \infty} x_m$, onde x_m é obtido por $x_{m+1} = \mathcal{G}(x_m)$.

Prova. Temos que $x_{m+1} = \mathcal{G}(x_m)$ e $x_m = \mathcal{G}(x_{m-1})$. Logo

$$\|x_{m+1} - x_m\| = \|\mathcal{G}(x_m) - \mathcal{G}(x_{m-1})\| \leq L \|x_m - x_{m-1}\|.$$

Por indução obtemos $\|x_{m+1} - x_m\| \leq L^m \|x_1 - x_0\|$. Daí, pela desigualdade triangular,

$$\begin{aligned} \|x_{m+p} - x_m\| &\leq \sum_{i=1}^p \|x_{m+i} - x_{m+i-1}\| \\ &\leq \sum_{j=m}^{m+p-1} L^j \|x_1 - x_0\| \\ &\leq \frac{L^m}{1-L} \|x_1 - x_0\|. \end{aligned}$$

Como $L^m \rightarrow 0$ quando $m \rightarrow \infty$, (x_m) é uma sequência de Cauchy. Como V é completo, $\lim_{m \rightarrow \infty} x_m = x^*$ existe em V . Agora, $\mathcal{G}(x^*) = \mathcal{G}(\lim_{m \rightarrow \infty} x_m) = \lim_{m \rightarrow \infty} \mathcal{G}(x_m) =$

¹i.e., $\|\mathcal{G}(x) - \mathcal{G}(y)\| \leq L \|x - y\|$, $\forall x, y \in V$ ($0 < L < 1$).

$\lim_{m \rightarrow \infty} x_{m+1} = x^*$. Portanto, x^* é um ponto fixo de $\mathcal{G}$ em V . Suponhamos que existe outro $y \in V$ tal que $\mathcal{G}(y) = y$. Então

$$\|x^* - y\| = \|\mathcal{G}(x) - \mathcal{G}(y)\| \leq L \|x^* - y\|.$$

Daqui $(1 - L) \|x^* - y\| \leq 0$. Como $L < 1$, $\|x^* - y\| = 0$ e $x^* = y$. ■

Teorema 3.2. Seja a equação polinomial matricial

$$A_0 X^m + A_1 X^{m-1} + \dots + A_{m-1} X + A_m = 0. \quad (3.4)$$

Seja $\|\cdot\|$ uma norma multiplicativa em $\mathcal{M}(n, \mathbb{C})$. Sejam $M = \max \{\|A_i\| : i = 0, 1, \dots, m\}$ e N um número real tal que $0 < N \leq \frac{1}{4}$.

Se

$$\exists A_{m-1}^{-1} \text{ e } \|A_{m-1}^{-1}\| < \min \left\{ \frac{N(1-N)}{(1-N)\|A_m\| + M(N^2 - N^{m+1})}, \frac{(1-N)^2}{M(mN^{m+1} - mN^m - N^m + 2N - N^2)} \right\},$$

então a equação (3.4) tem uma solução em $\mathbb{E} = \{X \in \mathcal{M}(n, \mathbb{C}) : \|X\| \leq N\}$.

Observação 3.1. Sejam $f, g : (0, \frac{1}{4}] \mapsto \mathbb{R}$ as funções polinomiais definidas por

$$f(x) = \sum_{i=0}^m \alpha_i x^i \quad \text{e} \quad g(x) = \sum_{i=0}^m \beta_i x^i,$$

onde

$$\alpha_i = \begin{cases} M \|A_{m-1}^{-1}\|, & i = 2, 3, \dots, m \\ 0, & i = 1 \\ \|A_m\| \|A_{m-1}^{-1}\|, & i = 0 \end{cases}$$

$$\beta_i = \begin{cases} -mM \|A_{m-1}^{-1}\|, & i = m \\ M \|A_{m-1}^{-1}\|, & i = 2, 3, \dots, m-1 \\ 2(M \|A_{m-1}^{-1}\| + 1), & i = 1 \\ -1, & i = 0. \end{cases}$$

Se existe um N tal que $f(N) < N$ e $g(N) < N$, então este N satisfaz as condições do Teorema 3.2.

Antes de provarmos o teorema vamos provar o seguinte lema:

Lema 3.1. Seja $\mathcal{G} : \mathbb{E} \mapsto \mathcal{M}(n, \mathbb{C})$ a aplicação definida por

$$\mathcal{G}(X) = (A_{m-1}^{-1} \sum_{i=2}^m A_{m-i} X^{i-1} + I)^{-1} A_{m-1}^{-1} (-A_m).$$

Se as condições do Teorema 3.2 são satisfeitas, então temos:

1. $\mathcal{G}$ está bem definida;
2. $\mathcal{G}(\mathbb{E}) \subset \mathbb{E}$;
3. $\mathcal{G} : \mathbb{E} \mapsto \mathbb{E}$ é uma contração.

Prova. Sejam X qualquer elemento de $\mathbb{E}$ e $\mu = \frac{M \|A_{m-1}^{-1}\| (N - N^m)}{1 - N}$.

Prova do item 1. Temos que

$$\begin{aligned} \|A_{m-1}^{-1} \sum_{i=2}^m A_{m-i} X^{i-1}\| &\leq \|A_{m-1}^{-1}\| \sum_{i=2}^m \|A_{m-i}\| \|X\|^{i-1} \\ &\leq \|A_{m-1}^{-1}\| M \sum_{i=1}^{m-1} \|X\|^i \\ &\leq \|A_{m-1}^{-1}\| M \sum_{i=1}^{m-1} N^i \\ &= \frac{M \|A_{m-1}^{-1}\| (N - N^m)}{1 - N} \\ &= \mu < 1, \end{aligned}$$

pois $\|A_{m-1}^{-1}\| \leq \frac{N(1 - N)}{(1 - N) \|A_m\| + M(N^2 - N^{m+1})} < \frac{1 - N}{M(N - N^m)}$. Logo, pela Proposição

3.3, $(A_{m-1}^{-1} \sum_{i=2}^m A_{m-i} X^{i-1} + I)^{-1}$ existe. Portanto, $\mathcal{G}$ está bem definida. ■

Prova do item 2. Temos pela Proposição 3.3 que

$$\begin{aligned}
 \|\mathcal{G}(X)\| &= \left\| \sum_{s=0}^{\infty} (-1)^s (A_{m-1}^{-1} \sum_{i=2}^m A_{m-i} X^{i-1})^s A_{m-1}^{-1} (-A_m) \right\| \\
 &\leq \sum_{s=0}^{\infty} \left\| (A_{m-1}^{-1} \sum_{i=2}^m A_{m-i} X^{i-1})^s \right\| \|A_{m-1}^{-1}\| \|A_m\| \\
 &\leq \frac{1}{1-\mu} \|A_m\| \|A_{m-1}^{-1}\| \\
 &\leq N,
 \end{aligned}$$

pois $\mu = \frac{M \|A_{m-1}^{-1}\| (N - N^m)}{1 - N} \leq \frac{MN(N - N^m)}{(1 - N) \|A_m\| + M(N^2 - N^{m+1})}$. Portanto $\mathcal{G} : \mathbb{E} \mapsto \mathbb{E}$. ■

Prova do item 3. Seja Y uma matriz em $\mathbb{E}$. Então

$$\begin{aligned}
 \|\mathcal{G}(X) - \mathcal{G}(Y)\| &= \left\| (A_{m-1}^{-1} \sum_{i=2}^m A_{m-i} X^{i-1} + I)^{-1} A_{m-1}^{-1} (-A_m) - (A_{m-1}^{-1} \sum_{i=2}^m A_{m-i} Y^{i-1} + I)^{-1} A_{m-1}^{-1} (-A_m) \right\| \\
 &\stackrel{(\text{Prop. 3.3.})}{=} \left\| \sum_{s=0}^{\infty} (-1)^s \left[(A_{m-1}^{-1} \sum_{i=2}^m A_{m-i} X^{i-1})^s - (A_{m-1}^{-1} \sum_{i=2}^m A_{m-i} Y^{i-1})^s \right] A_{m-1}^{-1} (-A_m) \right\| \\
 &\leq \|A_{m-1}^{-1}\| \|A_m\| \sum_{s=0}^{\infty} \sum_{i=2}^m \left\| (A_{m-1}^{-1} A_{m-i} X^{i-1})^s - (A_{m-1}^{-1} A_{m-i} Y^{i-1})^s \right\| \\
 &\stackrel{(\text{Prop. 3.1.})}{\leq} \|A_{m-1}^{-1}\| \|A_m\| \sum_{s=1}^{\infty} \sum_{k=0}^{s-1} \left\| \sum_{i=2}^m A_{m-1}^{-1} A_{m-i} X^{i-1} \right\|^{s-1-k} \left\| \sum_{i=2}^m A_{m-1}^{-1} A_{m-i} Y^{i-1} \right\|^k \left\| \sum_{i=2}^m A_{m-1}^{-1} A_{m-i} (X^{i-1} - Y^{i-1}) \right\| \\
 &\leq \|A_{m-1}^{-1}\| M \sum_{s=1}^{\infty} \left(\|A_{m-1}^{-1}\| \|A_m\| \sum_{i=2}^m \|X^{i-1} - Y^{i-1}\| \right) \sum_{k=0}^{s-1} \mu^{s-1} \\
 &\stackrel{(\text{Prop. 3.1.})}{\leq} \|A_{m-1}^{-1}\|^2 \|A_m\| M \sum_{i=2}^m \left(\sum_{k=0}^{i-2} \|X\|^{i-2-k} \|Y\|^k \|X - Y\| \right) \left(\sum_{s=1}^{\infty} s \mu^{s-1} \right)
 \end{aligned}$$

$$\begin{aligned}
& \stackrel{(\text{Por (3.3)})}{\leq} \frac{\|A_{m-1}^{-1}\|^2 \|A_m\| M}{(1-\mu)^2} \left(\sum_{i=2}^m \sum_{k=0}^{i-2} N^{i-2} \right) \|X - Y\| \\
& \stackrel{(\text{Prop. 3.2.})}{=} \frac{\|A_{m-1}^{-1}\|^2 \|A_m\| M(mN^m - N^m - mN^{m-1} + 1)}{(1-\mu)^2(1-N)^2} \|X - Y\|
\end{aligned}$$

$$< \|X - Y\|,$$

pois $\frac{\|A_{m-1}^{-1}\| \|A_m\|}{1-\mu} \leq N$ e $\|A_{m-1}^{-1}\| < \frac{(1-N)^2}{M(mN^{m+1} - mN^m - N^m + 2N - N^2)}$.
 Portanto $\mathcal{G}$ é uma contração sobre $\mathbb{E}$. ■

Prova do Teorema 3.2. Do Lema 3.1 e do Teorema 3.1 temos que $\mathcal{G} : \mathbb{E} \mapsto \mathbb{E}$ tem um único ponto fixo em $\mathbb{E}$. Portanto, existe um único X_* tal que $\mathcal{G}(X_*) = X_*$, isto é,

$$X_* = (A_{m-1}^{-1} \sum_{i=2}^m A_{m-i} X_*^{i-1} + I)^{-1} A_{m-1}^{-1} (-A_m).$$

Logo

$$-A_m = [A_{m-1} (A_{m-1}^{-1} \sum_{i=2}^m A_{m-i} X_*^{i-1} + I)] X_*,$$

isto é

$$0 = \left(\sum_{i=2}^m A_{m-i} X_*^{i-1} + A_{m-1} \right) X_* + A_m,$$

isto é

$$0 = \sum_{i=0}^m A_{m-i} X_*^i.$$

Portanto X_* é solução da Equação (3.4). ■

Exemplo 3.1. Seja a equação

$$\begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} X^2 + \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} X + \begin{pmatrix} 0.1 & 0 \\ 0 & 0 \end{pmatrix} = 0. \quad (3.5)$$

Seja a norma matricial:

$$\|A\| = \max \left\{ \sum_{i=1}^n |a_{ik}| : k = 1, \dots, n \right\}, \quad \text{onde } A \text{ é uma matriz de ordem } n.$$

Utilizando o algoritmo que forneceremos na próxima seção, temos que uma solução para a Equação (3.5) é:

$$\begin{pmatrix} -0.11270 & 0 \\ 0 & 0 \end{pmatrix}.$$

3.2 Algoritmo

Nesta seção fornecemos um algoritmo para o método dado neste capítulo. Usaremos a Equação (3.4) como nossa equação a resolver e a norma multiplicativa dada acima.

Algoritmo 3.1

1. Ler as matrizes A_i ($i = 0, \dots, m$).
2. Fornecer a precisão ϵ com que se quer trabalhar.
3. Encontrar N de acordo com a Observação 3.1.
4. Calcular $M = \max\{\|A_i\|: i = 0, 1, \dots, m\}$.
5. Se A_{m-1} satisfaz as condições do Teorema 3.2

V: {A equação tem solução.}

5.1. $X_1 = 0$.

5.2. Repita

5.2.1. $X_0 = X_1$.

5.2.2. $X_1 = \left(\sum_{i=1}^m A_{m-i} X_0^{i-1}\right)^{-1} (-A_m)$.

5.2.3. Até que $(\|X_1 - X_0\| < \epsilon)$.

5.3. A solução é X_1 .

F: {Não podemos afirmar nada.}

5.1 Podemos tentar com uma equação próxima, segundo a norma de Frobenius, da equação original. (Ver os Teoremas 4.2 e 4.3).

6. Fim do algoritmo.

Capítulo 4

Discussão

Neste capítulo apresentamos a comparação dos métodos dados nos Capítulos 2 e 3 com o método obtido por Menzaque [10].

4.1 Existência de Soluções

Nesta seção, vamos explicar o método de Menzaque para resolver a equação

$$A_0X^m + A_1X^{m-1} + \dots + A_{m-1}X + A_m = 0, \quad (4.1)$$

quando a triangularização simultânea dos coeficientes A_i ($i = 0, \dots, m$) é possível.

Por meio de um exemplo mostraremos o método. Note que o método depende inicialmente da triangularização simultânea das matrizes $A_0, \dots, A_m$ e em seguida de um algoritmo para resolver a equação associada que tem como coeficientes matrizes triangulares superiores.

Exemplo 4.1. Seja a equação polinomial matricial

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{pmatrix} X^2 + \begin{pmatrix} 2 & -2 & 0 \\ -2 & 4 & -2 \\ 0 & -1 & 1 \end{pmatrix} X + \begin{pmatrix} 1 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 1 \end{pmatrix} = 0.$$

Se

$$U = \begin{pmatrix} \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{6}} \\ \frac{1}{\sqrt{3}} & 0 & -\frac{2}{\sqrt{6}} \\ \frac{1}{\sqrt{3}} & -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{6}} \end{pmatrix} \quad \text{e} \quad V = \begin{pmatrix} \frac{1}{\sqrt{6}} & \frac{2}{\sqrt{6}} & \frac{1}{\sqrt{6}} \\ \frac{1}{\sqrt{2}} & 0 & -\frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{3}} & -\frac{1}{\sqrt{3}} & \frac{1}{\sqrt{3}} \end{pmatrix},$$

multiplicando os coeficientes da equação à esquerda por V e à direita por U , e definindo $Y = U^* X U^1$, obtemos que a equação é equivalente a:

$$\begin{pmatrix} \sqrt{2} & 0 & -1 \\ 0 & 1 & 0 \\ 0 & 0 & \sqrt{2} \end{pmatrix} Y^2 + \begin{pmatrix} 0 & 0 & -2 \\ 0 & 2 & 0 \\ 0 & 0 & 4\sqrt{2} \end{pmatrix} Y + \begin{pmatrix} 0 & 0 & -1 \\ 0 & 1 & 0 \\ 0 & 0 & 2\sqrt{2} \end{pmatrix} = 0. \quad (4.2)$$

Se Y for uma matriz triangular

$$\begin{pmatrix} y_{11} & y_{12} & y_{13} \\ 0 & y_{22} & y_{23} \\ 0 & 0 & y_{33} \end{pmatrix},$$

substituindo em (4.2), temos

$$\begin{pmatrix} \sqrt{2}y_{11}^2 & \sqrt{2}(y_{11}+y_{22})y_{12} & -1+\sqrt{2}y_{11}y_{13}+\sqrt{2}y_{12}y_{23}+y_{33}(-2+\sqrt{2}y_{13})+y_{33}^2 \\ 0 & (1+y_{22})^2 & y_{23}(2+y_{22}+y_{33}) \\ 0 & 0 & \sqrt{2}(2+4y_{33}+y_{33}^2) \end{pmatrix} = 0.$$

Resolvemos esta equação partindo da diagonal da matriz até a supradiagonal que fica mais longe (nessa ordem) e obtemos que

$$Y = \begin{pmatrix} 0 & 0 & \frac{1-\sqrt{2}}{2} \\ 0 & -1 & 0 \\ 0 & 0 & -2+\sqrt{2} \end{pmatrix},$$

é uma solução para (4.2).

Agora, daremos o algoritmo para calcular soluções de uma equação polinomial matricial cujos coeficientes são matrizes triangulares superiores.

¹ A^* representa a matriz transposta conjugada de A .

Algoritmo 4.1

Se $z_1, \dots, z_n$ são números complexos, então definimos uma matriz triangular superior S com elementos s_{ij} por:

```

Para  $i = 1, \dots, n$ 
     $s_{ii} = z_i$ 
     $y_{ii}^{(0)} = a_{ii}^{(0)}$ 
    Para  $k = 1, \dots, m$ 
         $y_{ii}^{(k)} = y_{ii}^{(k-1)} s_{ii} + a_{ii}^{(k)}$ 
Para  $h = 1, \dots, n - 1$ 
    Para  $i = 1, \dots, n - h$ 
        Definir  $j = i + h$ 
        Calcular  $c_{ij} = c_{ij}^{(m)}$  por
             $c_{ij}^{(0)} = 0$ 
            Para  $k = 1, \dots, m$ 
                 $c_{ij}^{(k)} = y_{ii}^{(k-1)} + c_{ij}^{(k-1)} s_{jj}$ 
        Calcular  $d_{ij} = d_{ij}^{(m)}$  por
             $d_{ij}^{(0)} = a_{ij}^{(0)}$ 
            Para  $k = 1, \dots, m$ 
                 $d_{ij}^{(k)} = d_{ij}^{(k-1)} s_{jj} + \sum_{l=i+1}^{j-1} y_{il}^{(k-1)} s_{lj} + a_{ij}^{(k)}$ 
         $s_{ij} = -d_{ij}/c_{ij}$ 
        Para  $k = 1, \dots, m - 1$ 
             $y_{ij}^{(k)} = d_{ij}^{(k)} + s_{ij} c_{ij}^{(k)}$ 

```

Se todos os coeficientes da Equação (4.1) são matrizes diagonais, todos os números c_{ij} são não-nulos. Assim sendo, neste caso, sempre existe uma matriz diagonal, solução de (4.1). No caso geral, a existência de soluções depende de que os coeficientes c_{ij} não se anulem.

As condições nas quais a matriz S , construída pelo algoritmo acima, é uma solução de (4.1) (com matrizes triangulares $A_0, \dots, A_m$) são fornecidas pelo Teorema 4.1.

Para cada $i = 1, \dots, n$, seja $p_i(z)$ o polinômio definido, a partir da Equação (4.1), por:

$$p_i(z) = \sum_{l=0}^m a_{ii}^{(m-l)} z^l. \quad (4.3)$$

Para cada $i = 1, \dots, n$ seja z_i uma raiz do polinômio $p_i(z)$ definido em (4.3). Se $\tilde{p}_i(z)$ é definido por

$$\tilde{p}_i(z) = \begin{cases} \frac{p_i(z)}{z - z_i} & \text{se } z \neq z_i, \\ p'_i(z) & \text{se } z = z_i, \end{cases} \quad (4.4)$$

temos a seguinte caracterização dos coeficientes c_{ij} :

Lema 4.2. Suponhamos que na Equação (4.1) os A_i ($i = 0, \dots, m$) são matrizes triangulares superiores. Se S é uma matriz triangular superior, solução de (4.1), tal que para todo i , $s_{ii} = z_i$ é um zero de $p_i(z)$, então para $i < j$

$$c_{ij} = \tilde{p}_i(z_j).$$

Prova. Ver [10], pag. 89. ■

Teorema 4.1. Seja

$$A_0 X^m + A_1 X^{m-1} + \dots + A_{m-1} X + A_m = 0, \quad (4.5)$$

onde os coeficientes A_i ($i = 0, \dots, m$) são matrizes triangulares superiores. Sejam $p_1(z), \dots, p_n(z)$ os polinômios definidos por (4.3). Se existem números complexos $z_1, \dots, z_n$ satisfazendo

$$p_i(z_i) = 0 \quad (i = 1, \dots, n), \quad \tilde{p}_i(z_j) \neq 0 \quad (1 \leq i < j \leq n),$$

então existe uma matriz triangular superior S , tal que $s_{ii} = z_i$, solução de (4.5).

Prova. Ver [10], pag. 90. ■

4.2 Estabilidade de Soluções

Os teoremas nesta seção estão relacionados a erros na representação dos coeficientes da Equação (4.1), e nada tem a ver com os erros cometidos por algum método particular empregado no cálculo de soluções de dita equação.

Seja o polinômio matricial

$$P(X) = \sum_{i=0}^m A_{m-i} X^i. \quad (4.6)$$

Definição 4.1. A Derivada de Fréchet do polinômio $P(X)$ em X_0 é o operador linear $P'(X_0)$ definido para toda matriz $H \in \mathcal{M}(n, \mathbb{C})$ por:

$$P'(X_0)[H] = \sum_{i=1}^m A_{m-i} \sum_{k=1}^i X_0^{k-1} H X_0^{i-k} = \sum_{i=1}^m \left(\sum_{k=i}^m A_{m-k} X_0^{k-i} \right) H X_0^{i-1}. \quad (4.7)$$

A Derivada de Fréchet de $P(X)$ em X_0 é chamada **regular** se o operador $P'(X_0)$ for bijetor.

Definição 4.2. Um zero S de $P(X)$ é chamado **zero simples** se $P'(S)$ for invertível.

No que segue usaremos a **norma de Frobenius** $\|\cdot\|_F$ que é definida por

$$\|A\|_F^2 = \sum_{i=1}^n \sum_{j=1}^n |a_{ij}|^2,$$

onde $A \in \mathcal{M}(n, \mathbb{C})$.

Sejam $P(X)$ o polinômio em (4.6), S e $\tilde{S} = S + \Delta S$ os zeros de $P(X)$ e do polinômio perturbado $\tilde{P}(X)$ respectivamente, onde

$$\tilde{P}(X) = \sum_{i=0}^m \tilde{A}_{m-i} X^i,$$

onde $\tilde{A}_i = A_i + \Delta A_i$.

Sejam $P'(X)$ e $\tilde{P}'(X)$ as derivadas de Fréchet de $P(X)$ e $\tilde{P}(X)$ respectivamente, e sejam, por definição:

$$\Delta P(X) = \tilde{P}(X) - P(X) = \sum_{i=0}^m \Delta A_{m-i} X^i$$

e

$$\Delta P'(X)[H] = \tilde{P}'(X)[H] - P'(X)[H].$$

Teorema 4.2 Sejam $P(X)$ e $\tilde{P}(X)$ dois polinômios matriciais tais que, para algum ϵ positivo suficientemente pequeno, as perturbações ΔA_i satisfazem

$$\|\Delta A_i\|_F \leq \epsilon \|A_i\|_F \quad (i = 0, \dots, m).$$

Se S e $\tilde{S}$ forem zeros de $P(X)$ e $\tilde{P}(X)$, respectivamente, satisfazendo $\|\Delta S\|_F \leq \|S\|_F$ e, além disso, se S for um zero simples de $P(X)$ para o qual $1 \leq \|S\|_F$ e

$$\|P'(S)^{-1}\|_F \| \Delta P'(S) \|_F \leq \kappa < 1,$$

então $\|\Delta S\|_F \leq \gamma\epsilon$, onde:

$$\begin{aligned} \gamma &= \frac{2\alpha}{1 + \sqrt{1 - 4\alpha\beta\epsilon}}, \\ \alpha &= \frac{2}{1 - \kappa} \|P'(S)^{-1}\|_F \left(\sum_{i=1}^m \|A_{m-i}\|_F \|S\|_F^i \right), \\ \beta &= \frac{\delta(1 + \epsilon)}{1 - \kappa} \|P'(S)^{-1}\|_F, \\ \delta &= 2^{m+1} \|S\|_F^{m-2} \max\{\|A_{m-i}\|_F : i = 1, \dots, m\}. \end{aligned}$$

Prova. Ver [10], pp. 20-21. ■

O teorema acima fornece uma estimativa caso o erro relativo dos coeficientes seja pequeno, isto é, se $P(X)$ tiver um coeficiente nulo, o mesmo coeficiente terá que ser nulo em $\tilde{P}(X)$. Agora enunciaremos um teorema que não tem esta restrição.

Teorema 4.3. Sejam $P(X)$ e $\tilde{P}(X)$ dois polinômios matriciais tais que, para algum ϵ positivo suficientemente pequeno, as perturbações ΔA_i satisfazem

$$\|\Delta A_i\|_F \leq \epsilon \max\{\|A_{m-i}\|_F : i = 1, \dots, m\}.$$

Se S e $\tilde{S}$ forem zeros de $P(X)$ e $\tilde{P}(X)$, respectivamente, satisfazendo $\|\Delta S\|_F \leq \|S\|_F$ e, além disso, se S for um zero simples de $P(X)$ para o qual $1 \leq \|S\|_F$ e

$$\|P'(S)^{-1}\|_F \| \Delta P'(S) \|_F \leq \kappa < 1,$$

então $\|\Delta S\|_F \leq \gamma\epsilon$, onde:

$$\begin{aligned}\gamma &= \frac{2\alpha}{1 + \sqrt{1 - 4\alpha\beta\epsilon}}, \\ \alpha &= \frac{2}{1 - \kappa} \|P'(S)^{-1}\|_F \max\{\|A_{m-i}\|_F : i = 1, \dots, m\} \left(\sum_{i=1}^m \|S\|_F^i \right), \\ \beta &= \frac{\delta(1 + \epsilon)}{1 - \kappa} \|P'(S)^{-1}\|_F, \\ \delta &= 2^{m+1} \|S\|_F^{m-2} \max\{\|A_{m-i}\| : i = 1, \dots, m\}.\end{aligned}$$

Prova. Ver [10], pag. 22. ■

4.3 Conclusões

De acordo com o que vimos na Seção 4.1, o último método para resolver a equação

$$A_0 X^m + A_1 X^{m-1} + \dots + A_{m-1} X + A_m = 0 \quad (4.8)$$

é muito simples, mas a condição de **triangularização simultânea** dos coeficientes é muito forte em comparação com as condições dadas nos dois métodos anteriores. De fato, a condição dada no Teorema 3.2 sobre a **não-singularidade da matriz A_{m-1}** não é muito forte, pois usando o Teorema 4.3, parece ser possível a aproximação de A_{m-1} por uma matriz não-singular sem afetar as soluções. Notamos também que a eficiência do primeiro método (usando a Forma Canônica de Jordan), depende diretamente da eficiência dos algoritmos existentes para encontrar valores, vetores próprios e a inversa de uma matriz, e o segundo método depende exclusivamente da eficiência no cálculo da inversa de uma matriz. Podemos então concluir que o método mais eficiente para saber sobre a existência de soluções da Equação (4.8), no caso de ter como coeficientes matrizes triangulares superiores, é o último pela sua simplicidade, mas se as matrizes não são desta forma, então os dois métodos anteriores são os melhores e o segundo é mais eficiente do que o primeiro desde que possamos encontrar um N tal que as condições sobre A_{m-1} sejam satisfeitas.

Seria muito interessante estudar a solução da Equação (4.8) no caso em que a matriz A_{m-1} está muito próxima da matriz nula. Isto porque a norma de A_{m-1} estaria muito próxima do zero e a norma de A_{m-1}^{-1} , quando for aproximada por alguma outra matriz, seria um número muito grande e, portanto, as condições do Teorema 3.2 não seriam satisfeitas. O estudo da Equação (4.8), nesse caso, poderia ser feito usando o conceito de **Matriz Inversa Generalizada** como foi proposto pelo Professor Cyro de Carvalho Patarra.

Apêndice A

Programas

A.1 Programação do Algoritmo 2.1

```
Program Jordan_Method;
Uses Crt;
Const max= 10;
      eps= 1E-7;
Type
  matrix = array[1..max, 1..max] of real;
  cvector= array[1..max] of real;
  ivector= array[1..max] of integer;
Var
  bloco, m, n, nv, mgtotal, vli, i, j: integer;
      B, JJ, P, Q, RR: matrix;
      tambl, columns, index, mg, r: ivector;
      lambda: cvector;
      exist: boolean;

Procedure make(var B:matrix; m, n:integer);
Var
  nome, path: string[40];
```

```
    arquivo: text;
    i, j, k: integer;
    ch: string[2];
begin
    {As matrizes tem o nome "matriz"+ um número+".dat" }
    write('Ingresa o caminho: '); readln(path);
    for i:=1 to m-1 do
        for k:=n*(i-1)+1 to n*i do
            for j:=1 to m*n do
                if (j = k+n) then B[k,j] := 1
                else B[k,j] := 0;
            for j:=m downto 1 do
                begin
                    str(j, ch);
                    nome:= path + 'matriz' + ch + '.dat';
                    assign(arquivo, nome);
                    reset(arquivo);
                    for i:=n*(m-1)+1 to m*n do
                        begin
                            for k:=n*(m-j)+1 to n*(m-j+1) do
                                begin
                                    read(arquivo, B[i,k]);
                                    B[i,k] := -B[i,k]
                                end;
                                readln(arquivo)
                            end;
                        close(arquivo)
                    end
                end
            end;
end;
```

```
Procedure Inverse(A:matrix; n:integer; var D:matrix; var exist:boolean);
Var
  i, j, k, l: integer;
  aux: real;
  E: matrix;
begin
  exist:= true;
  for i:=1 to n do
    for j:=1 to n do
      if (i = j) then E[i,i]:= 1
        else E[i,j]:= 0;
    i:=1;
  while (i <= n) and (exist) do
    begin
      j:= i;
      l:= i;
      for k:=i+1 to n do
        if (abs(A[k,j]) > abs(A[l,j])) then l:=k;
      if (abs(A[l,j]) <= eps)
      then exist:= false
      else begin
        for k:=1 to n do
          begin
            aux:= A[l,k];
            A[l,k]:= A[i,k];
            A[i,k]:= aux;
            aux:= E[l,k];
            E[l,k]:= E[i,k];
            E[i,k]:= aux
          end;
        end;
      end;
```

```

    for k:=i+1 to n do
    begin
        aux:= A[k,i]/A[i,i];
        for l:=1 to n do
        begin
            A[k,l]:= A[k,l] - aux*A[i,l];
            E[k,l]:= E[k,l] - aux*E[i,l]
        end
    end;
    i:= i+1
end
end;
if (exist)
then begin
    for j:=1 to n do
    begin
        E[n,j]:= E[n,j]/A[n,n];
        for k:=n-1 downto 1 do
        begin
            for i:=k+1 to n do E[k,j]:= E[k,j]-A[k,i]*E[i,j];
            E[k,j]:= E[k,j]/A[k,k]
        end
    end;
    for j:=1 to n do
        for k:=1 to n do D[j,k]:= E[j,k]
    end
end;
end;

```

```

Procedure eigenvalues(var lambda:cvector; B:matrix);
begin

```



```
writeln('Ingresa todos os valores próprios:');  
for i:=1 to m*n do read(lambda[i]);  
  {Encontra os valores próprios da matriz B, armazenando-os em lambda.}  
end;
```

```
Procedure product(A, B:matrix; p, q, s:integer; var C:matrix);
```

```
Var
```

```
  i, j, k: integer;
```

```
begin
```

```
  for i:=1 to p do
```

```
    for j:=1 to s do
```

```
      begin
```

```
        C[i,j]:=0;
```

```
        for k:=1 to q do C[i,j]:= C[i,j] + A[i,k]*B[k,j]
```

```
      end
```

```
end;
```

```
Procedure product1(A, B:matrix; f1i, f1f, c1i, c1f, c2i, c2f:integer;
```

```
  var C:matrix);
```

```
Var
```

```
  i, j, k:integer;
```

```
begin
```

```
  for i:=f1i to f1f do
```

```
    for j:=c2i to c2f do
```

```
      begin
```

```
        C[i,j]:= 0;
```

```
        for k:=c1i to c1f do C[i,j]:= C[i,j] + A[i,k]*B[k,j]
```

```
      end
```

```
end;
```

```
Procedure prodmatvec(A: matrix; X:cvector; m, n: integer; var Y:cvector);
```

```
Var
```

```
    i, j: integer;
```

```
begin
```

```
    for i:=1 to m do
```

```
        begin
```

```
            Y[i]:= 0;
```

```
            for j:=1 to n do
```

```
                Y[i]:= Y[i] + A[i,j]*X[j]
```

```
            end
```

```
        end;
```

```
Procedure algebraicmult(lambda:cvector; var beta:cvector;
```

```
                        var r:ivector; var nv: integer);
```

```
Var
```

```
    i, j, k: integer;
```

```
    aux: real;
```

```
begin
```

```
    nv:=1;
```

```
    for i:=1 to (m*n-1) do
```

```
        for j:=i+1 to m*n do
```

```
            if (lambda[j] < lambda[i])
```

```
            then begin
```

```
                aux:= lambda[i];
```

```
                lambda[i]:= lambda[j];
```

```
                lambda[j]:= aux
```

```
            end
```

```
            else for k:=j downto i+2 do
```

```
                begin
```

```
                    aux:= lambda[k];
```

```

        lambda[k] := lambda[k-1];
        lambda[k-1] := aux
    end;

    r[1] := 1;
    beta[nv] := lambda[1];
    for i:=2 to m*n do
        if (beta[nv] = lambda[i])
        then r[nv] := r[nv]+1
        else begin
            nv := nv+1;
            beta[nv] := lambda[i];
            r[nv] := 1
        end
    end;

end;

Procedure linindepvect(X:cvector; var P:matrix; var indep:boolean;
                      cl, ini, fi, mafi:integer);

Var
    over, the_end: boolean;
    i, k, l, s: integer;
    aux: real;
    B: matrix;

begin
    indep := false;
    for i:=1 to m*n do
        for k:=ini to fi do B[i,k-ini+1] := P[i,k];
    for i:=1 to m*n do B[i,fi-ini+2] := X[i];
    i := 1;
    the_end := false;
    while (i <= fi-ini+2) and (not the_end) do

```

```
begin
  over:= false;
  k:= i;
  while (k <= fi-ini+2) and (not over) do
    begin
      l:= i;
      while (abs(B[l,k]) <= eps) and (l <= m*n) do l:= l+1;
      if (l > m*n)
        then k:= k+1
        else over:= true
    end;
  if over
  then begin
    if (k <> i)
    then for s:=1 to m*n do
      begin
        aux:= B[s,k];
        B[s,k]:= B[s,i];
        B[s,i]:= aux
      end;
    if (l <> i)
    then for s:=i to fi-ini+2 do
      begin
        aux:= B[i,s];
        B[i,s]:= B[l,s];
        B[l,s]:= aux
      end;
    for k:=i+1 to m*n do
      begin
        aux:= B[k,i]/B[i,i];
```

```

        for l:=i to fi-ini+2 do B[k,l]:= B[k,l] - aux*B[i,l]
    end;
    i:= i+1
end
else the_end:= true
end; {end of while}
if (i > fi-ini+2)
then indep:= true
else indep:= false;
if (indep)
then for i:=1 to m*n do
    for k:=cl to mafi do
    begin
        aux:= P[i,k];
        P[i,k]:= X[i];
        X[i]:= aux
    end
end;

Procedure eigenvectors(lambda:cvector; B:matrix; var P, JJ:matrix;
    var bloco:integer; var tambl, mg:ivector; ma:ivector);
Var
    auxi, i, j, k, l, s, cont, contma: integer;
    the_end, over: boolean;
    permuta: ivector;
    X, Y: cvector;
    aux: real;
    BB, E: matrix;
begin
    contma:= 1;

```

```
bloco:= 1;
for i:=1 to m*n do
    for k:=1 to m*n do BB[i,k] := B[i,k];
for j:=1 to m*n do tambl[j] := 1;
for j:=1 to nv do
begin
    for i:= 1 to ma[j] do J[contma+i-1,contma+i-1] := lambda[j];
    for i:=1 to m*n do
        for k:=1 to m*n do
            begin
                E[i,k] := 0;
                B[i,k] := BB[i,k]
            end;
        for i:=1 to m*n do
            begin
                B[i,i] := B[i,i] - lambda[j];
                permuta[i] := i;
                E[i,i] := 1
            end;
        i:= 1;
        the_end:= false;
        while (i <= m*n) and (not the_end) do
            begin
                over:= false;
                k:= i;
                while (k <= m*n) and (not over) do
                    begin
                        l:= i;
                        while (abs(B[l,k]) <= eps) and (l <= m*n) do l:= l+1;
                        if (l > m*n) then k:= k+1
```

```
else over:= true

end;

if over
then begin
  if (k <> i)
  then begin
    for s:=1 to m*n do
    begin
      aux:= B[s,k];
      B[s,k]:= B[s,i];
      B[s,i]:= aux
    end;
    auxi:= permuta[k];
    permuta[k]:= permuta[i];
    permuta[i]:= auxi
  end;
  if (l <> i)
  then begin
    for s:=i to m*n do
    begin
      aux:= B[i,s];
      B[i,s]:= B[l,s];
      B[l,s]:= aux
    end;
    for s:=1 to m*n do
    begin
      aux:= E[i,s];
      E[i,s]:= E[l,s];
      E[l,s]:= aux
    end
  end
end
```

```

        end;
        for k:=i+1 to m*n do
        begin
            aux:= B[k,i]/B[i,i];
            for l:=i to m*n do
            begin
                B[k,l]:= B[k,l] - aux*B[i,l];
                E[k,l]:= E[k,l] - aux*E[i,l]
            end
        end;
        i:= i+1
    end
    else the_end:= true
end; {end of while}
i:= i-1;
mg[j]:= m*n-i;
inverse(B, i, B, over);
product1(B, B, 1, i, 1, i, i+1, m*n, B);
for s:=1 to m*n-i do
begin
    for k:=1 to m*n do
        if (k <= i)
        then P[permuta[k],s]:= -B[k,s+i]
        else if (k = s+i)
            then P[permuta[k],s+contma-1]:= 1
            else P[permuta[k],s+contma-1]:= 0
        end;
    for s:=1 to mg[j] do tambl[bloco]:=1;
    cont:= mg[j];
    s:= contma;

```



```
while (cont < ma[j]) do
begin
    for k:=1 to m*n do X[k]:= P[k,s];
    prodmatvec(E, X, m*n, m*n, X);
    for k:= 1 to i do
    begin
        Y[k]:= 0;
        for l:=1 to i do
            Y[k]:= Y[k] + B[k,l]*X[l]
        end;
        for k:=1 to i do X[k]:= Y[k];
        for k:= i+1 to m*n do X[k]:= 0;
        for k:=1 to m*n do Y[permuta[k]]:= X[k];
        for k:=1 to m*n do X[k]:= Y[k];
        linindepvect(X, P, over, s+1, contma, contma+cont-1, contma+ma[j]-1);
        if (over) then
        begin
            tambl[bloco]:= tambl[bloco]+1;
            JJ[s,s+1]:= 1;
            cont:= cont+1
        end
        else bloco:= bloco+1;
        s:= s+1
    end;
    bloco:= bloco+1;
    contma:= contma + ma[j];
end; {fim do for j}
bloco:= bloco-1
end;
```

```
Procedure linearindep(P:matrix; n, m:integer; var index:ivector);
Var
    i, j, k, l, cont: integer;
        stop: boolean;
        aux: real;
begin
    i:=1; cont:=0;
    while (i <= n) and (cont <= n) do
        begin
            j:= i;
            l:= i;
            for k:=i+1 to n do
                if (abs(P[k,j]) > abs(P[l,j])) then l:=k;
            stop:= false;
            if (abs(P[l,j]) <= eps)
            then j:= j+1
            else stop:= true;
            while (not stop) do
                begin
                    l:= i;
                    while (abs(P[l,j]) <= eps) and (l <= n) do l:=l+1;
                    if (l > n)
                    then j:=j+1
                    else stop:= true;
                end;
            if (j <> i)
            then begin
                for k:=1 to n do
                    begin
                        aux:= P[k,j];
```

```

        P[k,j] := P[k,i];
        P[k,i] := aux
    end
end;
if (l <> i)
then begin
    for k:=i to m do
    begin
        aux:= P[l,k];
        P[l,k] := P[i,k];
        P[i,k] := aux
    end
end;
cont:= cont+1;
index[cont] := j;
if (cont < n)
then for k:=i+1 to n do
    begin
        aux:= P[k,i]/P[i,i];
        for l:=1 to m do
            P[k,l] := P[k,l] - aux*P[i,l]
        end;
        i:=i+1
    end;
end;
end;

Procedure linindeporder(P:matrix; column:ivector; ini, fi:integer;
                        var indep:boolean);

var
    i, k, l, s: integer;

```

```
over, the_end: boolean;
    aux: real;
    B: matrix;
begin
    indep:= false;
    for i:=1 to n do
        for k:= ini to fi do B[i,k-ini+1]:= P[i,k];
    i:= 1;
    the_end:= false;
    while (i <= n) and (not the_end) do
    begin
        over:= false;
        k:= i;
        while (k <= fi-ini+1) and (not over) do
        begin
            l:= i;
            while (abs(B[l,k]) <= eps) and (l <= n) do l:= l+1;
            if (l > n) then k:= k+1
                else over:= true
            end;
        if (over)
        then begin
            if (k <> i)
            then for s:=1 to n do
                begin
                    aux:= B[s,k];
                    B[s,k]:= B[s,i];
                    B[s,i]:= aux
                end;
            if (l <> i)
```

```

        then for s:=i to fi-ini+1 do
            begin
                aux:= B[i,s];
                B[i,s]:= B[l,s];
                B[l,s]:= aux
            end;
        for k:=i+1 to n do
            begin
                aux:= B[k,i]/B[i,i];
                for l:=i to fi-ini+2 do B[k,l]:= B[k,l]-aux*B[i,l]
            end;
            i:= i+1
        end
        else the_end:= true
    end; {end of while}
    if (i > fi-ini+1) then indep:= true
        else indep:= false
    end;
end;

```

```

Procedure order(var columns:ivector; tambl:ivector; bloco:integer;
                var vli:integer);

```

```

Var

```

```

    contam, cont, cont1, cont2, i, j, k, l, s: integer;
                                lli, li: boolean;
                                index: ivector;

```

```

Procedure linearmenteindep(P: matrix; var ind:ivector;
                            var c:integer; cont:integer);

```

```

Var

```

```

    i, j, k, l: integer;
    aux: real;

```

```
B: matrix;
stop: boolean;

begin
  for i:=1 to n do
    for j:=1 to cont do B[i,j]:= P[i,ind[j]];
  vli:=0;
  i:= 1;
  while (i <= n) and (vli < n) do
    begin
      j:= i;
      l:= i;
      for k:=i+1 to n do
        if (abs(B[k,j]) > abs(B[l,j])) then l:=k;
      stop:= false;
      if (abs(B[l,j]) <= eps) then j:= j+1
        else stop:= true;
      while (not stop) and (j <= cont) do
        begin
          l:= i;
          while (abs(B[l,j] <= eps) and (l <= n) do l:=l+1;
          if (l > n) then j:=j+1
            else stop:= true;
        end;
      if (stop)
      then begin
        if (j <> i)
        then begin
          for k:=1 to n do
            begin
              aux:= B[k,j];
```

```
        B[k,j] := B[k,i];
        B[k,i] := aux
    end
end;
if (l <> i)
then begin
    for k:=i to cont do
    begin
        aux:= B[l,k];
        B[l,k] := B[i,k];
        B[i,k] := aux
    end
end;
vli:= vli+1;
index[vli] := ind[j];
for k:=i+1 to n do
begin
    aux:= B[k,i]/B[i,i];
    for l:=1 to cont do
        B[k,l] := B[k,l]-aux*B[i,l]
    end;
    i:=i+1
end
else i:=n+2
end;
for i:=1 to vli do ind[i] := index[i]
end;

begin
    contam:=1;
```

```

cont:= 0;
cont1:= 0;
cont2:= 0;
for i:=1 to bloco do
begin
    s:= contam;
    cont:= cont+1;
    cont1:= 1;
    columns[cont]:= s;
    for j:= 2 to tambl[i] do
    begin
        if (j-columns[cont] = 1)
        then linindeporder(P, columns, s, contam+j-1, li)
        else li:=false;
        if (li)
        then begin
            cont:= cont+1;
            columns[cont]:= contam+j-1;
            cont1:= cont1+1
        end
        else begin
            cont2:= 1;
            k:= j+1;
            lii:= true;
            index[1]:= s+j-1;
            while ((cont2<cont1) and (k<=tambl[i])) and (lii) do
            begin
                linindeporder(P, index, s, s+k-1, lii);
                if (lii)
                then begin

```



```
        cont2:= cont2+1;
        index[cont]:= s+k-1
    end;
    k:= k+1
end;
if (cont2 = cont1)
then begin
    k:=1;
    while (columns[k] < s) do k:= k+1;
    for l:=k to cont2+k-1 do
        columns[l]:=index[l-k+1];
    cont1:= cont2;
    s:= columns[k]
    end
end
end;
contam:= contam+tambl[i]
end;
linearmenteindep(P, columns, vli, cont)
end;

begin {Programa Principal}
    clrscr;
    write('Ordem das matrizes: '); readln(n);
    write('Grau da equação   : '); readln(m);
    make(B, m, n);
    eigenvalues(lambda, B);
    algebraicmult(lambda, lambda, r, nv);
    for i:=1 to m*n do
        for j:=1 to m*n do
```

```
begin
    P[i,j]:=0;
    JJ[i,j]:=0
end;
eigenvectors(lambda, B, P, JJ, bloco, tambl, mg, r);
mgtotal:=0;
for i:=1 to nv do mgtotal:= mgtotal+mg[i];
if (nv = m*n)
then begin
    linearindep(P, n, m*n, columns);
    vli:= n
end
else order(columns, tambl, bloco, vli);
if (vli >= n)
then begin
    for i:=1 to n do
        for j:=1 to n do
            begin
                Q[i,j]:= P[i,columns[j]];
                RR[i,j]:= JJ[columns[i],columns[j]]
            end;
        inverse(Q, n, JJ, exist);
        product(Q, RR, n, n, n, Q);
        product(Q, JJ, n, n, n, JJ);
        writeln('Uma solução da equação é:');
        for i:=1 to n do
            begin
                for j:=1 to n do write(JJ[i,j]);
                writeln
            end
        end
```

```
        end
        else writeln('A equação não tem solução. ');
        readln
    end.
```

A.2 Programação do Algoritmo 3.1

```
Program Point_Fix_Method;
Uses crt;
Const
    precisao= 1E-10;
    max= 30;
Type
    matrix= array[1..max, 1..max] of real;
Var
    aux1, nn, mm, aux: real;
    i, j, m, n: integer;
    nome: string[40];
    A, B, X0, A0: matrix;

Function norma_multi(A: matrix): real;
Var
    sum, max: real;
    i, j: integer;
begin
    max:= 0;
    for j:=1 to n do
        begin
            sum:= 0;
```

```
        for i:=1 to n do sum:= sum + abs(A[i,j]);
        if (sum > max) then max:= sum
    end;
    norma_multi:= max
end;
```

```
Function distancia(A, B: matrix): real;
```

```
Var
```

```
    i, j:integer;
```

```
begin
```

```
    for i:=1 to n do
```

```
        for j:=1 to n do A[i,j]:=A[i,j] - B[i,j];
```

```
    distancia:= norma_multi(A)
```

```
end;
```

```
Procedure add(A, B:matrix; m, n:integer; var C:matrix);
```

```
Var
```

```
    i, j:integer;
```

```
begin
```

```
    for i:=1 to m do
```

```
        for j:=1 to n do C[i,j]:=A[i,j]+B[i,j]
```

```
end;
```

```
Procedure Le(var A:matrix; nn:integer);
```

```
Var
```

```
    arquivo: text;
```

```
    name: string[50];
```

```
    ch: string[2];
```

```
    i, j: integer;
```

```
begin
```

```
    str(nn, ch);
    name:= nome+'matriz'+ch+'.dat';
    assign(arquivo, name);
    reset(arquivo);
    for i:=1 to n do
    begin
        for j:=1 to n do read(arquivo, A[i,j]);
        readln(arquivo)
    end;
    close(arquivo)
end;
```

```
Procedure inverse(A: matrix; n:integer; var E:matrix; var exist:boolean);
begin
    {ja foi implementado no algoritmo anterior.}
end;
```

```
Procedure product(A, B: matrix; p, q, s:integer; var C:matrix);
begin
    {ja foi implementado no algoritmo anterior.}
end;
```

```
Function pot(x: real; k:integer): real;
Var
    aux: real;
    i: integer;
begin
    aux:= x;
    for i:=1 to k-1 do aux:= aux*x;
    pot:= aux
```

end;

begin {Programa Principal}

clrscr;

write('Ingrese o grau da equação: '); readln(m);

write('Ingrese a ordem das matrizes: '); readln(n);

write('Ingrese o radio da bola no espaço de matrizes ($0 < nn \leq 1/3$): ');

readln(nn);

writeln('Ingrese o caminho onde estão os coeficientes matriciais: ');

readln(nome);

Le(A, 1);

inverse(A, n, B, exist);

if (exist)

then begin

Le(A0,0);

for i:=1 to n do

for j:=1 to n do A0[i,j] := -A0[i,j];

mm:= norma_multi(A0);

aux1:= mm;

for i:=1 to m do

begin

Le(A, i);

aux:= norma_multi(A);

if (mm < aux) then mm:= aux

end;

if (norma_multi(B) <= nn*(1-nn)/(((1-nn)*aux1+mm*(nn*nn-pot(nn,m+1))))

then begin

for i:=1 to n do

for j:=1 to n do B[i,j] := 0;

repeat

```
        for i:=1 to n do
            for j:=1 to n do X0[i,j]:= B[i,j];
        Le(B, m);
        for i:=m-1 downto 1 do
            begin
                product(B, X0, n, n, n, B);
                Le(A, i);
                add(B, A, n, n, B);
            end;
            inverse(B, n, B, exist);
            product(B, A0, n, n, n, B);
        until (distancia(B, X0) < precisao);
        writeln('Uma solução da equação é:');
        for i:=1 to n do
            begin
                for j:=1 to n do write(B[i,j]:10:5, ' ');
                writeln
            end
        end
    end
else begin
    writeln('A matriz A1 é singular. Tente com outra matriz que seja');
    writeln('próxima de A1 na norma de Frobenius.')
end;
readln
end.
```


Referências Bibliográficas

- [1] E. K. Blum, *Numerical Analysis and Computation Theory and Practice*. Addison-Wesley Publishing Company, Inc., 1972.
- [2] J. D. Botha, *A short treatment of the Jordan canonical form theory*. Expositions Mathematicae, vol. II, no. 4 (1993), 365-367.
- [3] J. Sotomayor, *Lições de Equações Diferenciais Ordinárias*. Instituto de Matemática Pura e Aplicada (Projeto Euclides), 1979.
- [4] R. A. Brualdi, *The Jordan canonical form: an old proof*. Amer. Math. Monthly, 94 (1987), 257-267.
- [5] R. Fletcher and D. C. Sorensen, *An algorithmic derivation of the Jordan canonical form*. Amer. Math. Monthly, 90 (1983), 12-16.
- [6] G. H. Golub and J. H. Wilkinson, *Ill-conditioned eigensystems and the computation of the Jordan canonical form*. SIAM Review, vol. 18, no. 4 (1976), 578-619.
- [7] H. L. Guidorizzi, *Jordan canonical form: an elementary proof*. Trabalhos do Departamento de Matemática, IME-USP, RT-MAT 95-02.
- [8] K. Hoffmann and R. Kunze, *Álgebra Linear*. Livros Técnicos e Científicos Editora S.A., 1976.
- [9] B. Kagstrom and A. Ruhe, *An algorithm for numerical computation of the Jordan normal form of a complex matrix*. ACM Transactions on Mathematical Software, vol. 6, no. 3 (1980), 398-419.

- [10] F. E. Menzaque, *Equações Polinomiais Matriciais: Métodos de Resolução*. Tese de Doutorado. IME-USP., 1993.
- [11] O. Plaat, *Ecuaciones Diferenciales Ordinarias*. Editorial Reverté, S.A., 1974.