UNIVERSIDADE FEDERAL DE SANTA CATARINA
CAMPUS BLUMENAU
LICENCIATURA EM MATEMÁTICA

Taina da Silva

Resolução de sistemas lineares em $\mathbb{Z}_p$

Blumenau
2022

Taina da Silva

Resolução de sistemas lineares em $\mathbb{Z}_p$

Trabalho de Conclusão de Curso de Graduação em Licenciatura em Matemática do Campus Blumenau da Universidade Federal de Santa Catarina para a obtenção do título de Licenciado em Matemática.
Orientador: Prof. Dr. Luiz Rafael dos Santos
Coorientador: Me. Luiz Fernando Bossa

Blumenau
2022

Ficha de identificação da obra elaborada pelo autor,
através do Programa de Geração Automática da Biblioteca Universitária da UFSC.

Silva, Taina da
Resolução de sistemas lineares em Zp / Taina da Silva ; orientador, Luiz Rafael dos Santos, coorientador, Luiz Fernando Bossa, 2022.
147 p.

Trabalho de Conclusão de Curso (graduação) - Universidade Federal de Santa Catarina, Campus Blumenau, Graduação em Matemática, Blumenau, 2022.

Inclui referências.

1. Matemática. 2. Sistemas lineares. 3. Corpo Zp. 4. Linguagem Julia. 5. Cifra de Hill. I. Santos, Luiz Rafael dos. II. Bossa, Luiz Fernando. III. Universidade Federal de Santa Catarina. Graduação em Matemática. IV. Título.

Taina da Silva

Resolução de sistemas lineares em $\mathbb{Z}_p$

Este Trabalho Conclusão de Curso foi julgado adequado para obtenção do Título de Licenciado em Matemática e aprovado em sua forma final pelo Curso de Licenciatura em Matemática.

Blumenau, 08 de março de 2022.

Documento assinado digitalmente
Julio Faria Correa
Data: 16/03/2022 20:14:13-0300
CPF: 045.818.289-37
Verifique as assinaturas em https://v.ufsc.br

Prof. Dr. Júlio Faria Corrêa
Coordenador do Curso de Licenciatura em Matemática

Banca Examinadora:

Documento assinado digitalmente
Luiz Rafael dos Santos
Data: 16/03/2022 19:22:48-0300
CPF: 034.737.869-26
Verifique as assinaturas em https://v.ufsc.br

Prof. Dr. Luiz Rafael dos Santos
Orientador
Universidade Federal de Santa Catarina – UFSC

Documento assinado digitalmente
Luiz Fernando Bossa
Data: 16/03/2022 20:08:10-0300
CPF: 081.114.689-89
Verifique as assinaturas em https://v.ufsc.br

Me. Luiz Fernando Bossa
Coorientador
Universidade Federal de Santa Catarina – UFSC

Documento assinado digitalmente

Louise Reips
Data: 17/03/2022 09:32:03-0300
CPF: 000.405.570-57
Verifique as assinaturas em https://v.ufsc.br

Profa. Dra. Louise Reips
Universidade Federal de Santa Catarina – UFSC

Documento assinado digitalmente

Rafael Aleixo de Carvalho
Data: 17/03/2022 08:41:50-0300
CPF: 302.494.178-98
Verifique as assinaturas em https://v.ufsc.br

Prof. Dr. Rafael Aleixo de Carvalho
Universidade Federal de Santa Catarina – UFSC

A todos aqueles que como eu apreciam conhecer e compreender as ferramentas matemáticas por trás do funcionamento das coisas cotidianas.

AGRADECIMENTOS

Agradeço a minha família e ao meu noivo, Joel Paludo, pelo apoio, incentivo e por compreenderem minha ausência enquanto me dedicava a graduação e a este trabalho.

Aos meus amigos, João G. Voss, Vantuir D. Junior, Pedro A. T. Junior e Bruna G. Schimitz, pelas boas risadas e por todo apoio e lealdade durante esse período da graduação.

Aos professores Louise Reips e Felipe Vieira, que me possibilitam ser extensionista por dois anos, me proporcionando evolução na carreira da docência e dedicação total a universidade.

Ao professor Luiz Rafael dos Santos, por todo seu apoio durante a graduação, pela paciência e dedicação nas diversas orientações que me permitiram grande crescimento acadêmico e pessoal.

Ao professor Luiz Fernando Bossa, pelo apoio e dedicação na orientação deste trabalho.

A Universidade Federal de Santa Catarina por ter me dado condições de realizar esta graduação e este trabalho.

Ao PIBIC/CNPq e a PIBI/UFSC por financiarem as iniciações científicas que me propiciaram dedicação total a universidade e a este trabalho.

A todo corpo docente da UFSC-Blumenau por todos os conhecimentos e valores adquiridos durante estes anos de graduação.

"Não há nenhum ramo da Matemática, por mais abstrato que seja, que não possa um dia ser aplicado a fenômenos reais."
Nikolai Lobachevsky

RESUMO

Esta monografia visa explorar e compreender o funcionamento métodos para resolução de sistemas lineares, nos quais as componentes que definem as equações lineares pertencem ao corpo $\mathbb{Z}_p$, bem como implementá-los computacionalmente. Para isto, exploramos o algoritmo da divisão euclidiana, e definimos e caracterizaremos o conjunto $\mathbb{Z}_p$ como um corpo. Além disso, caracterizamos sistemas lineares no corpo dos números reais, e analisamos algoritmos para resolução de sistemas lineares em $\mathbb{Z}_p$. Finalmente, construímos uma representação computacional de $\mathbb{Z}_p$ por meio da linguagem de programação `Julia`, implementamos os algoritmos para resolução de sistemas lineares em $\mathbb{Z}_p$, e apresentamos uma aplicação em criptografia utilizando cifra de Hill.

Palavras-chave: Sistemas lineares; Corpo $\mathbb{Z}_p$; Linguagem Julia; Cifra de Hill.

ABSTRACT

This monograph aims to explore and understand the operation of methods for solving linear systems, in which the components defining the linear equations belong to the field $\mathbb{Z}_p$, as well as to computationally implement them. To this end, we exploit the Euclidean division algorithm, and define and characterize the set $\mathbb{Z}_p$ as a field. Furthermore, we characterize linear systems in the field of real numbers, and analyze algorithms for solving linear systems in Zp. Finally, we construct a computational representation of $\mathbb{Z}_p$ by means of the programming language `Julia`, we implement algorithms for solving linear systems in $\mathbb{Z}_p$, and we present an application in cryptography using Hill's cipher.

Keywords: Linear systems; Field $\mathbb{Z}_p$; Julia language; Hill's cipher.

SUMÁRIO

INTRODUÇÃO

A história mostra que, desde o surgimento de linguagens humanas mais complexas — em particular, a escrita —, métodos para codificar uma mensagem de modo que somente seu destinatário legítimo consiga interpretá-la foram pensados. O estudo desses métodos constitui a criptografia: do grego, *cryptos*, secreto ou oculto. Naturalmente, na mesma medida foram pensados métodos para decifrar códigos criptografados.

Um dos códigos de criptografia utilizados durante a história, é a cifra de Hill, que se utiliza de um anel $\mathbb{Z}_n$ em seu processo de cifragem.

Cifra de Hill

A cifra de Hill foi inventada pelo matemático norte-americano Lester S. Hill em 1929. Segundo Brandão [4, p.53], "A Cifra de Hill consiste em um sistema de criptografia por substituição poli-alfabética, baseado em transformações matriciais."

Nesta cifra, primeiro necessitamos definir uma tabela de conversão, para que nossos caracteres sejam transformados em números antes de serem cifrados. A quantidade n de caracteres que a tabela possui, irá definir em que anel iremos trabalhar, como estamos lidando com unidades inteiras, esse anel será $\mathbb{Z}_n$. Em seguida convertemos a mensagem usando a tabela escolhida.

Então, precisamos escolher a chave de criptografia, que deve ser uma matriz quadrada A de ordem $m \times m$ em que m depende do tamanho da mensagem que queremos cifrar. Essa matriz precisa possuir inversa tal que ambas possuam entradas inteiras, já que estamos trabalhando no anel $\mathbb{Z}_n$, caso contrário, não seria possível descriptografar a mensagem.

O próximo passo é quebrar nosso texto cifrado em blocos de tamanho m, e caso a mensagem não seja múltipla de m, precisamos completar a mensagem com algum caractere. Feito isso, convertemos estes blocos em vetores que denotaremos por p.

Para criptografar a mensagem temos que computar o produto $c = Ap$ em que c são os vetores que correspondem aos blocos da mensagem já criptografada. Após, é preciso encontrar as classes de equivalência correspondentes dos elementos no vetor c e, por fim, podemos usar a tabela para verificar a mensagem criptografada.

Por outro lado, dado um vetor criptografado c queremos encontrar o vetor original p, tal que $Ap = c$, isto é, precisamos resolver um sistema linear. Por exemplo, para descriptografar a mensagem, podemos calcular a inversa de A, tal que $A \cdot A^{-1} = I$, depois calcular o produto $A^{-1}c = p$ e encontrar suas correspondências na tabela. Vamos ver como isto funciona na prática.

Pré-codificação

Antes da criptografia, é necessário realizar uma fase de pré-codificação, em que os caracteres da mensagem são convertidos em números através de uma tabela. Observe a tabela de conversão a seguir:

0	a 1	b 2	c 3	d 4	e 5	f 6	g 7	h 8	i 9	j 10	k 11
l 12	m 13	n 14	o 15	p 16	q 17	r 18	s 19	t 20	u 21	v 22	w 23
x 24	y 25	z 26	à 27	á 28	â 29	ã 30	ç 31	é 32	ê 33	í 34	ó 35
ô 36	õ 37	ú 38	A 39	B 40	C 41	D 42	E 43	F 44	G 45	H 46	I 47
J 48	K 49	L 50	M 51	N 52	O 53	P 54	Q 55	R 56	S 57	T 58	U 59
V 60	W 61	X 62	Y 63	Z 64	À 65	Á 66	Â 67	Ã 68	Ç 69	É 70	Ê 71
Í 72	Ó 73	Ô 74	Õ 75	Ú 76	0 77	1 78	2 79	3 80	4 81	5 82	6 83
7 84	8 85	9 86	. 87	, 88	; 89	: 90	? 91	! 92	" 93	% 94	$ 95
- 96	+ 97	= 98	* 99	/ 100	\ 101	# 102	< 103	> 104	(105	) 106	[107
] 108	{ 109	} 110	@ 111	Enter 112							

Tabela 1 – Tabela de conversão para cifragem.

Note que, como este trabalho trata de sistemas lineares em corpos finitos, escolhemos uma tabela com 113 elementos para podermos trabalhar no corpo $\mathbb{Z}_{113}$. Porém, a cifra de Hill original trabalha com uma tabela de 26 elementos e efetua os cálculos necessários no anel $\mathbb{Z}_{26}$.

Suponha para fins de exemplo que desejamos criptografar a frase "Que a força esteja com você". Primeiramente, aplicando a pré-codificação, observamos a tabela de correspondência, e verificamos que a frase corresponde à 55 21 5 0 1 0 6 15 18 31 1 0 5 19 20 5 10 1 0 3 15 13 0 22 15 3 33.

Codificando

Vamos dar início à criptografia de fato. Primeiro devemos escolher uma matriz A, quadrada de ordem $m \times m$, em que m depende do tamanho

da mensagem a ser cifrada; como estamos cifrando uma mensagem pequena, vamos utilizar $m = 3$. Devemos escolher A de modo que seja inversível e que tanto A quanto sua inversa tenham entradas no corpo $\mathbb{Z}_{113}$. Tome então

$$A = \begin{bmatrix} 1 & 3 & 3 \\ 1 & 4 & 110 \\ 1 & 3 & 4 \end{bmatrix}.$$

Agora, vamos separar nossa mensagem em blocos de tamanho m, no caso "Que a força esteja com você" = 55 21 5 0 1 0 6 15 18 31 1 0 5 19 20 5 10 1 0 3 15 13 0 22 15 3 33 ficará como (Que) (a) (for) (ça) (est) (eja) (co) (m v) (ocê) = 55 21 5 0 1 0 6 15 18 31 1 0 5 19 20 5 10 1 0 3 15 13 0 22 15 3 33. Por conseguinte, convertemos nossos blocos em vetores coluna de tamanho m, denotados aqui por p_k em que $k = 1, \cdots, 9$:

$$p_1 = \begin{bmatrix} 55 \\ 21 \\ 5 \end{bmatrix}, p_2 = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, p_3 = \begin{bmatrix} 6 \\ 15 \\ 18 \end{bmatrix},$$

$$p_4 = \begin{bmatrix} 31 \\ 1 \\ 0 \end{bmatrix}, p_5 = \begin{bmatrix} 5 \\ 19 \\ 20 \end{bmatrix}, p_6 = \begin{bmatrix} 5 \\ 10 \\ 1 \end{bmatrix},$$

$$p_7 = \begin{bmatrix} 0 \\ 3 \\ 15 \end{bmatrix}, p_8 = \begin{bmatrix} 13 \\ 0 \\ 22 \end{bmatrix}, p_9 = \begin{bmatrix} 15 \\ 3 \\ 33 \end{bmatrix}.$$

Então, devemos calcular os produtos $c_k = A \cdot p_k$, com $k = 1, \cdots, 9$. O método usual de cifra de Hill efetua as contas com os vetores separadamente, mas, para facilitar e otimizar nossos cálculos, em vez de efetuar nove operações, podemos concatenar todos os vetores p_k em uma matriz P de ordem 3×9, efetuando uma única multiplicação. Por fins estéticos, vamos concatenar os vetores em trios, reescrevendo-os como três matrizes P_{ijk} de ordem 3×3 em que p_1, p_2 e p_3 compõem uma única matriz que denotaremos por P_{123}. Desta forma, quando calculamos $A \cdot P_{123}$ estamos

fazendo o equivalente a calcular $A \cdot p_1$, $A \cdot p_2$ e $A \cdot p_3$. Da mesma maneira, construímos as matrizes P_{456} e P_{789}.

Vamos então aos cálculos com as matrizes concatenadas.

$$C_{123} = AP_{123} = \begin{bmatrix} 1 & 3 & 3 \\ 1 & 4 & 110 \\ 1 & 3 & 4 \end{bmatrix} \cdot \begin{bmatrix} 55 & 0 & 6 \\ 21 & 1 & 15 \\ 5 & 0 & 18 \end{bmatrix} = \begin{bmatrix} 133 & 3 & 105 \\ 689 & 4 & 2046 \\ 138 & 3 & 123 \end{bmatrix}$$
$$= \begin{bmatrix} 20 & 3 & 105 \\ 11 & 4 & 12 \\ 25 & 3 & 10 \end{bmatrix},$$

em que apresentamos a conta intermediária em $\mathbb{Z}$ e após reduzimos a $\mathbb{Z}_{113}$. Doravante, apresentaremos apenas os resultados em $\mathbb{Z}_{113}$. Procedemos então para C_{456} e C_{789}:

$$C_{456} = AP_{456} = \begin{bmatrix} 1 & 3 & 3 \\ 1 & 4 & 110 \\ 1 & 3 & 4 \end{bmatrix} \cdot \begin{bmatrix} 31 & 5 & 5 \\ 1 & 19 & 10 \\ 0 & 20 & 1 \end{bmatrix} = \begin{bmatrix} 34 & 9 & 38 \\ 35 & 21 & 42 \\ 34 & 29 & 39 \end{bmatrix},$$

$$C_{789} = AP_{789} = \begin{bmatrix} 1 & 3 & 3 \\ 1 & 4 & 110 \\ 1 & 3 & 4 \end{bmatrix} \cdot \begin{bmatrix} 0 & 13 & 15 \\ 3 & 0 & 3 \\ 15 & 22 & 33 \end{bmatrix} = \begin{bmatrix} 54 & 79 & 10 \\ 80 & 60 & 41 \\ 69 & 101 & 43 \end{bmatrix}.$$

Por fim, observamos novamente a tabela de conversão e obtemos a mensagem cifrada 20 11 25 3 4 3 105 12 10 34 35 34 9 21 29 38 42 39 54 80 69 79 60 101 10 41 43 = “tkycdc(ljíóíiuâúDAP3Ç2V\jCE”, finalizando assim o processo de encriptação.

Decodificando

Analisemos agora o processo de descriptografia. Primeiramente, necessitamos da chave de decodificação. Para isto, precisamos calcular a inversa de A, no corpo $\mathbb{Z}_{113}$.

Calculemos a inversa de A: Para encontrar a inversa de A basta que encontremos uma matriz tal que $A \cdot A^{-1} = I$. Começamos realizando a

seguinte multiplicação

$$\begin{bmatrix} 1 & 3 & 3 \\ 1 & 4 & 110 \\ 1 & 3 & 4 \end{bmatrix} \cdot \begin{bmatrix} r & s & t \\ u & v & w \\ x & y & z \end{bmatrix}$$
$$= \begin{bmatrix} r+3u+3x & s+3v+3y & t+3w+3z \\ r+4u+110x & s+4v+110y & t+4w+110z \\ r+3u+4x & s+3v+4y & t+3w+4z \end{bmatrix}.$$

Agora basta igualar o resultado da multiplicação à matriz identidade

$$\begin{bmatrix} r+3u+3x & s+3v+3y & t+3w+3z \\ r+4u+110x & s+4v+110y & t+4w+110z \\ r+3u+4x & s+3v+4y & t+3w+4z \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

e resolvendo os três sistemas lineares

$$\begin{cases} r+3u+3x=1 \\ r+4u+110x=0 \\ r+3u+4x=0 \end{cases}, \quad \begin{cases} s+3v+3y=0 \\ s+4v+110y=1 \\ s+3v+4y=0 \end{cases}, \quad \begin{cases} t+3w+3z=0 \\ t+4w+110z=0 \\ t+3w+4z=1 \end{cases}$$

obtemos a inversa de A em $\mathbb{Z}_{113}$ que é dada por

$$A^{-1} = \begin{bmatrix} 25 & 110 & 92 \\ 106 & 1 & 6 \\ 112 & 0 & 1 \end{bmatrix}.$$

Para descriptografar a mensagem "tkycdc(ljíóíiuâúDAP3Ç2V\jCE", precisamos convertê-la novamente nas matrizes C_{123}, C_{456} e C_{789} e calcular os produtos $A^{-1}C_{123}$, $A^{-1}C_{456}$ e $A^{-1}C_{789}$. Portanto:

$$A^{-1}C_{123} = \begin{bmatrix} 4010 & 791 & 4864 \\ 2281 & 340 & 11202 \\ 2265 & 339 & 11770 \end{bmatrix} \equiv \begin{bmatrix} 55 & 0 & 6 \\ 21 & 1 & 15 \\ 5 & 0 & 18 \end{bmatrix} = P_{123},$$

$$A^{-1}C_{456} = \begin{bmatrix} 7828 & 5203 & 9158 \\ 3848 & 1149 & 4304 \\ 3842 & 1037 & 4295 \end{bmatrix} \equiv \begin{bmatrix} 31 & 5 & 5 \\ 1 & 19 & 10 \\ 0 & 20 & 1 \end{bmatrix} = P_{456},$$

$$A^{-1}C_{789} = \begin{bmatrix} 16498 & 17867 & 8716 \\ 6218 & 9040 & 1359 \\ 6117 & 8949 & 1163 \end{bmatrix} \equiv \begin{bmatrix} 0 & 13 & 15 \\ 3 & 0 & 3 \\ 15 & 22 & 33 \end{bmatrix} = P_{789}.$$

Aplicando a tabela de conversão a sequência numérica 55 21 5 0 1 0 6 15 18 31 1 0 5 19 20 5 10 1 0 3 15 13 0 22 15 3 33 temos novamente a frase "Que a força esteja com você", concluindo assim o processo de decodificação.

Vimos que para realizar o processo de criptografia utilizando cifragem de Hill, precisamos calcular uma matriz inversa. Sendo assim, sabe-se que os métodos utilizados para realizar tal computo exigem muitos cálculos. Um dos métodos mais baratos para conseguir obter uma matriz inversa é resolver sistemas lineares do tipo $Av = e_i$, em que v representa a coluna da matriz inversa e e_i representa a coluna correspondente da matriz identidade.

Como estamos tratando de matrizes no corpo $\mathbb{Z}_p$, surge a questão: como resolver um sistema linear no corpo $\mathbb{Z}_p$? É essa a problemática que buscaremos responder no decorrer desta monografia tanto do ponto de vista teórico quanto do computacional.

Por uma questão didática, organizaremos este estudo em sete capítulos que permitem responder a questão acima. O primeiro nos familiariza com conceitos importantes sobre algoritmo euclidiano estendido. No segundo, realizaremos a construção do corpo dos inteiros módulo p. No terceiro, relembraremos conceitos importantes acerca de matrizes. No quarto abordaremos o tema sistemas lineares e métodos utilizados para resolvê-los. No quinto, estudaremos os algoritmos para resolução de sistemas lineares no corpo $\mathbb{Z}_p$. Destacaremos no sexto capítulo, algumas características da linguagem de programação escolhida bem como explicamos as implementações e apresentamos um exemplo resolvendo uma cifra de Hill. Por fim, apresentamos as considerações finais.

1 ALGORITMO EUCLIDIANO E EUCLIDIANO ESTENDIDO

Para que possamos computar cálculos dentro de $\mathbb{Z}_p$, precisamos de algumas ferramentas algébricas, dentre elas estão o Algoritmo da divisão, o Algoritmo Euclidiano e o Algoritmo Euclidiano estendido, os quais estudaremos neste capítulo, tomando como referência o livro de Coutinho [5].

Um algoritmo é uma sequência não ambígua de instruções ou ainda, um processo utilizado para efetuar qualquer cálculo. Os algoritmos são classificados em exatos, — que buscam a melhor solução possível —, ou aproximados, que buscam uma solução satisfatória — que apesar de possuir uma margem de erro, tem resultados suficientemente bons para resolver o problema dado —, isto devido à complexidade de determinados problemas cuja resposta exata seria inviável.

Segundo Coutinho [5, p.18-19], a palavra algoritmo vem do árabe e significa *Al-Khowarazmi* ou, "o homem de Khowarazm", como chamavam o matemático árabe Ben Musa, que viveu no século IX. Foi por seu livro "*Al-jabr wa'l muqabalah*" que os algarismos indo-arábicos chegaram ao ocidente.

Exemplificando, suponha um algoritmo que lhe pede para vestir primeiro as suas meias e sapatos e por último suas calças, enquanto outro algoritmo lhe pede que ponha primeiro as calças e depois as meias e sapatos. Notamos que o processo do primeiro algoritmo é mais difícil e complexo, enquanto o segundo é mais fácil. Nada obstante, ambos atingem o mesmo resultado.

1.1 ALGORITMO DA DIVISÃO

Nesta seção estudaremos o algoritmo da divisão. Aqui, estamos interessados apenas nas divisões no conjunto dos inteiros positivos, ou seja, tais divisões consistem em encontrar o quociente q e o resto r quando estamos dividindo a por b com a, b, r, q inteiros positivos. Formalmente, temos a seguinte definição.

Definição 1.1. Sejam a e b, inteiros positivos, dizemos que b é *divisor* de a, se existir q inteiro positivo, tal que $a = b \cdot q$. Denotaremos este fato por $b \mid a$.

Naturalmente, dados a, b inteiros positivos, quando b não é divisor de a vamos provar adiante (Teoremas 1.2 e 1.3) que existem q e r inteiros positivos, tais que $a = b \cdot q + r$ e $0 < r < b$. Em geral, para qualquer caso, teremos sempre

$$a = b \cdot q + r \text{ e } 0 \leq r < b. \tag{1.1}$$

Teorema 1.2 (Existência)**.** *Dados a, b inteiros positivos, existem q, r inteiros positivos, tais que $a = b \cdot q + r$ e $0 \leq r < b$.*

Demonstração. Por hipótese, $a > 0$ e $b > 0$. Se $0 < a < b$, teremos que $q = 0$ e $r = a$. Se $a = b$, temos que $q = 1$ e $r = 0$. Suponha, então, $a > b > 0$ e considere enquanto fizer sentido, o conjunto S dos inteiros positivos da forma $a - b \cdot q$ em que $q \leq 0$:

$$S = \{a, a - b, a - 2 \cdot b, \ldots a - n \cdot b, \ldots\}.$$

Pelo princípio da boa ordenação, S tem um menor elemento, digamos $r := a - q \cdot b$. Precisamos provar que $r < b$. Se $r = b$, $q = 0$ que é o caso $0 < a < b$. Suponha $r > b$. Com isso existe c inteiro positivo tal que $r = c + b$, logo $r = c + b = a - b \cdot q > b$, por consequência $c = a - b \cdot q - b$ e por fim, $c = a - (q + 1) \cdot b \in S$, com $c < r$, o que é uma contradição com o fato de r ser o menor elemento de S. Portanto, temos que $a = b \cdot q + r$, com $r < b$, o que prova a existência de q e r. ■

Teorema 1.3 (Unicidade)**.** *Sejam a, b inteiros positivos. Se existem q, r inteiros positivos tais que $a = b \cdot q + r$ com $0 \leq r < b$, então tais q, r são únicos.*

Demonstração. Suponha que existam dois inteiros positivos q' e r', tais que $a = b \cdot q + r$ com $0 \leq r < b$ e $a = b \cdot q' + r'$, com $0 \leq r' < b$. Assim, temos que $r = a - b \cdot q$ e que $r' = a - b \cdot q'$.

Suponha $r \geq r'$, além disso, $r - r' = (a - b \cdot q) - (a - b \cdot q') = b(q' - q)$. Por outro lado, sabemos que tanto r como r' são menores que b, à proporção que $r \geq r'$, temos $0 \leq r - r' < b$.

Como $r - r' = b \cdot (q' - q)$, temos que $0 \leq b \cdot (q' - q) < b$ e então $0 \leq b - b \cdot (q' - q)$, de modo que $0 \leq q' - q < 1$. Já que $q' - q \in \mathbb{Z}$, $q' - q = 0$, o que garante $q' = q$ e $r = r'$. ■

A equação (1.1) sugere naturalmente um algoritmo de divisão, isto é, dados a, b inteiros positivos, temos um procedimento para encontrar q, r. Tal algoritmo consiste em três passos:

1. Faça $q = 0$ e $r = a$.

2. Se $r < b$ escreva o quociente q e o resto r e pare, se não vá para o Passo 3.

3. Enquanto $r \geq b$ subtraia b de r, incremente q em 1 e volte ao Passo 2.

O Algoritmo 1.1 apresenta este procedimento em forma de pseudo-código.

Algoritmo 1.1 Algoritmo da divisão

```
1: procedure DIVISAO(a, b)          ▷ Entrada a e b inteiros positivos
2:     q = 0
3:     r = a
4:     while r ≥ b do               ▷ Enquanto r maior ou igual a b, execute:
5:         r ← r − b                ▷ subtraia b de r
6:         q ← q + 1                ▷ Incremente q em uma unidade
7:     return q, r                  ▷ Saída q e r tais que a = b · q + r
```

Vejamos o Algoritmo 1.1 em ação através dos exemplos abaixo.

Exemplo 1.4. Vamos dividir 182 por 12.

Iteração 1

1. $182 = 12 \cdot 0 + 182$.

2. Como $r > b$ prosseguimos para o próximo passo.
3. $182 \div 12 = 1 \cdot 12 + 170$.

Iteração 2

1. $182 = 1 \cdot 12 + 170$.
2. Como $r > b$ prosseguimos para o próximo passo.
3. $182 \div 12 = 2 \cdot 12 + 158$.

Iteração 3

1. $182 = 2 \cdot 12 + 158$.
2. Como $r > b$ prosseguimos para o próximo passo.
3. $182 \div 12 = 3 \cdot 12 + 146$.

Continuamos até...

Iteração 15

1. $182 = 15 \cdot 12 + 2$.
2. como $r < b$ o quociente é 15 e o resto é 2.

Note que o algoritmo precisou fazer 15 iterações até concluir a divisão e nos retornou o quociente $q = 15$ e o resto $r = 2$. Assim, podemos escrever 182 como $182 = 12 \cdot 15 + 2$.

Vamos ver agora como o algoritmo funcionaria para a divisão de 12 por 182.

Exemplo 1.5. Vamos dividir 12 por 182.

Iteração 1

1. $12 = 182 \cdot 0 + 12$.

Note que 12 não é divisível por 182, então a saída são o quociente $q = 0$ e o resto $r = 12$, e escrevemos 12 como $12 = 182 \cdot 0 + 12$.

Observação 1.6. Aplicando o Algoritmo 1.1 repetidas vezes para todos os b's, tais que $0 < b < a$, com a, b inteiros positivos podemos listar todos os divisores de a.

1.2 ALGORITMO EUCLIDIANO

Devido à infinidade de números inteiros positivos, é natural que muitos deles possuam divisores em comum. O maior destes divisores é chamado máximo divisor comum ou, abreviadamente, mdc. A finalidade do algoritmo Euclidiano, é calculá-lo de forma rápida e eficaz. Vamos defini-lo formalmente.

Definição 1.7. Sejam a, b inteiros positivos com pelo menos um deles não-nulo. Dizemos que o inteiro positivo d é o *máximo divisor comum* de a, b, denotado por $\text{mdc}(a, b)$, se, e somente se, são satisfeitas as condições:

i) d é divisor de a e d é divisor de b.

ii) Se k divide ambos a e b , então, $k \leq d$.

Observação 1.8. No caso em que temos um único divisor comum de a e b, isto é, $mdc(a, b) = 1$, então a e b são ditos *primos entre si* ou coprimos.

1.2.1 Como calcular o máximo divisor comum de dois números?

Uma maneira ingênua para calcular o máximo divisor comum de dois números, seria calcular primeiramente todos os seus divisores individualmente, por exemplo, usando o Algoritmo 1.1, e em seguida compará-los, para ver qual o maior divisor comum. Porém, este algoritmo pode prescindir de um grande número de operações para terminar se a diferença entre os dois números envolvidos for um número muito maior que 1 em valor absoluto — visto que o algoritmo testa o quociente começando do zero e acrescentando

1 a cada iteração, até encontrar um quociente que deixe resto menor que b. Portanto, vejamos agora um algoritmo mais adequado.

O algoritmo da divisão nos fornece uma maneira de escrever um inteiro positivo a em função de seu divisor b de maneira que $a = b \cdot q + r$ para q, r inteiros positivos. Agora vamos apresentar o Algoritmo Euclidiano que utiliza o fato de que $\text{mdc}(a, b) = \text{mdc}(b, r)$ (veja Lema 1.9 a seguir), para podermos encontrar de maneira fácil e rápida o $\text{mdc}(a, b)$.

Lema 1.9. *Sejam a, b, q, r inteiros não-negativos, com $a \neq 0$, $b \neq 0$ e $a = b \cdot q + r$, com $0 \leq r < b$. Então,* $\text{mdc}(a, b) = \text{mdc}(b, r)$.

Demonstração. Sejam $d_1 = \text{mdc}(a, b)$ e $d_2 = \text{mdc}(b, r)$. Por definição, existem u, v inteiros positivos, tais que $a = d_1 \cdot u$ e $b = d_1 \cdot v$. Por hipótese, $a = b \cdot q + r$, substituindo os valores de a e b conforme os descritos acima, temos que $(d_1 \cdot u) = (d_1 \cdot v) \cdot q + r$. Assim, $r = d_1 \cdot u - d_1 \cdot v \cdot q$, desta forma obtemos $r = d_1 \cdot (u - v \cdot q)$. Portanto d_1 divide r, mas como $d_2 = \text{mdc}(b, r)$, temos que $d_1 \leq d_2$.

Similarmente, se $d_2 = \text{mdc}(b, r)$, existem inteiros positivos w, z, tais que $b = d_2 \cdot w$ e $r = d_2 \cdot z$. Substituindo estes valores de b e r na equação $a = b \cdot q + r$, temos que $a = (d_2 \cdot w) \cdot q + (d_2 \cdot z)$, e obtemos $a = d_2 \cdot (w \cdot q + z)$. Portanto d_2 divide a, mas como $d_1 = \text{mdc}(a, b)$, temos que $d_2 \leq d_1$.

Destarte, como $d_1 \leq d_2$ e $d_2 \leq d_1$, temos que $d_1 = d_2$. Assim, $\text{mdc}(a, b) = \text{mdc}(b, r)$ como queríamos. ■

O algoritmo Euclidiano consiste em um processo de divisões consecu-

tivas

$$\begin{aligned} a &= b \cdot q_1 + r_1, \\ b &= r_1 \cdot q_2 + r_2, \\ r_1 &= r_2 \cdot q_3 + r_3, \\ &\vdots \\ r_{n-4} &= r_{n-3} \cdot q_{n-2} + r_{n-2}, \\ r_{n-3} &= r_{n-2} \cdot q_{n-1} + r_{n-1}, \\ r_{n-2} &= r_{n-1} \cdot q_n + r_n, \text{ com } r_n = 0. \end{aligned}$$

Note que da equação

$$r_{n-2} = r_{n-1} \cdot q_n + r_n,$$

temos que r_{n-1} divide r_{n-2}, portanto, $\text{mdc}(r_{n-2}, r_{n-1}) = r_{n-1}$. Fazendo uso do Lema 1.9 na equação,

$$r_{n-3} = r_{n-2} \cdot q_{n-1} + r_{n-1},$$

obtemos que $\text{mdc}(r_{n-3}, r_{n-2}) = \text{mdc}(r_{n-2}, r_{n-1}) = r_{n-1}$. Se continuarmos o processo, aplicando o Lema 1.9 na equação

$$r_{n-4} = r_{n-3} \cdot q_{n-2} + r_{n-2},$$

obtemos $\text{mdc}(r_{n-4}, r_{n-3}) = \text{mdc}(r_{n-3}, r_{n-2}) = r_{n-1}$. Assim, usando o lema sucessivas vezes, obtemos que $\text{mdc}(a, b) = r_{n-1}$.

Veja abaixo o pseudo-código para o algoritmo Euclidiano.

Algoritmo 1.2 Algoritmo Euclidiano

```
1: procedure EUCLIDIANO(a, b)          ▷ Entrada a e b inteiros positivos
2:     while r ≠ 0 do    ▷ Se r for 0 o mdc é o menor numero entre a e b
3:         r ← DIVISAO(a,b)          ▷ r é resto da divisão inteira entre a e b
4:         a ← b
5:         b ← r
6:     return a                                        ▷ saída o mdc é a
```

Para facilitar, este algoritmo pode ser apresentado em forma de tabela, conforme segue

	q_1	q_2	q_3			q_{n-1}	q_n
a	b	r_1	r_2	...	r_{n-3}	r_{n-2}	r_{n-1}
r_1	r_2	r_3			r_{n-1}	0	

Agora, vejamos através um exemplo o Algoritmo 1.2 em ação.

Exemplo 1.10. Vamos calcular o $\text{mdc}(1234, 54)$:

$$\begin{aligned} 1234 &= 54 \cdot 22 + 46, \\ 54 &= 46 \cdot 1 + 8, \\ 46 &= 8 \cdot 5 + 6, \\ 8 &= 6 \cdot 1 + 2, \\ 6 &= 2 \cdot 3 + 0. \end{aligned}$$

Assim, $\text{mdc}(1234, 54) = 2$. Usando a forma de tabela temos

	22	1	5	1	3
1234	54	46	8	6	2
46	8	6	2	0	

1.2.2 Boa-definição do Algoritmo 1.2

Para calcular o $\text{mdc}(a, b)$, efetuamos uma sequência de divisões:

$$\begin{aligned} a &= b \cdot q_1 + r_1, & 0 \leq r_1 < b \\ b &= r_1 \cdot q_2 + r_2, & 0 \leq r_2 < r_1 \\ r_1 &= r_2 \cdot q_3 + r_3, & 0 \leq r_3 < r_2 \\ r_2 &= r_3 \cdot q_4 + r_4, & 0 \leq r_4 < r_3 \\ &\vdots & \vdots \end{aligned}$$

Então, olhando para os restos, temos que:

$$b > r_1 > r_2 > r_3 > r_4 > \ldots \geq 0.$$

Como entre b e 0 temos finitos inteiros positivos, esta sequência é finita e tal sequência só chega ao fim quando o resto é zero, isto garante que o Algoritmo 1.2 sempre para ao final e, portanto, está bem definido — este fato será formalmente demonstrado mais adiante.

Uma questão que se coloca aqui, além da boa definição do Algoritmo 1.2, é em relação ao máximo de divisões que precisamos realizar para chegar ao máximo divisor comum.

Considere novamente a sequência de restos $b > r_1 > r_2 > r_3 > r_4 > \ldots r_{n-1} > r_n = 0$. Assim, o maior resto possível em uma certa divisão é igual ao resto anterior menos um. Se isto acontecer em cada divisão, teremos que executar b divisões até chegar ao resto igual a zero, e este seria o pior caso possível. Mais formalmente,

$$\begin{aligned} r_{n-2} &= r_{n-1} \cdot 1, \\ r_{n-3} &= r_{n-2} \cdot 1 + r_{n-1}, \\ r_{n-4} &= r_{n-3} \cdot 1 + r_{n-2}, \\ &\vdots \\ a &= b \cdot 1 + r_1. \end{aligned}$$

Note que nesse pior caso, vale a relação $r_k = r_{k-1} + r_{k-2}$, tal qual a relação de recorrência de Fibonacci. Vamos exemplificar estas ideias.

Exemplo 1.11. Digamos $r_{n-1} = 1$ e que $n = 10$, a sequência de restos fornecida pelas equações acima (em ordem decrescente) é 34, 21, 13, 8, 5, 3,

2, 1, 1, 0.

$$\begin{aligned}
34 &= 21 \cdot 1 + 13,\\
21 &= 13 \cdot 1 + \ 8,\\
13 &= \ 8 \cdot 1 + \ 5,\\
8 &= \ 5 \cdot 1 + \ 3,\\
5 &= \ 3 \cdot 1 + \ 2,\\
3 &= \ 2 \cdot 1 + \ 1,\\
2 &= \ 1 \cdot 1 + \ 1,\\
1 &= \ 1 \cdot 1 + \ 0.
\end{aligned}$$

Na forma tabelada, temos:

	1	1	1	1	1	1	1	1
34	21	13	8	5	3	2	1	1
13	8	5	3	2	1	1	0	

Assim, os menores valores de a e b para os quais temos de efetuar n divisões de modo a calcular $\mathrm{mdc}(a, b)$ (e descobrir que vale 1) são $a = 34$ e $b = 21$. Novamente, esta sequência faz parte da famosa *Sequência de Fibonacci*.

Vamos demonstrar agora a boa-definição do Algoritmo 1.2 (Algoritmo Euclidiano).

Teorema 1.12. *Dados a, b inteiros positivos, o Algoritmo 1.2 (Algoritmo Euclidiano) gera a sequência finita $b > r_1 > r_2 > \ldots > r_{n-1} > r_n$ com as seguintes propriedades*

i) $r_n = 0$,

ii) $r_{n-1} = \mathrm{mdc}(a, b)$.

Demonstração. Supondo que o resto nulo ocorre após n divisões e aplicando o algoritmo a a e b, temos:

$$\begin{aligned}
a &= b \cdot q_1 + r_1, & 0 \leq r_1 < b \\
b &= r_1 \cdot q_2 + r_2, & 0 \leq r_2 < r_1 \\
r_1 &= r_2 \cdot q_3 + r_3, & 0 \leq r_3 < r_2 \\
r_2 &= r_3 \cdot q_4 + r_4, & 0 \leq r_4 < r_3 \\
&\vdots \\
r_{n-2} &= r_{n-1} \cdot q_n, & r_n = 0.
\end{aligned}$$

Da última linha, temos que r_{n-1} divide r_{n-2} portanto, $\text{mdc}(r_{n-1}, r_{n-2}) = r_{n-1}$. Aplicando sucessivamente o Lema 1.9, temos que $\text{mdc}(a, b) = r_{n-1}$. ■

Para finalizar essa subseção, como este trabalho também se propõe a implementar computacionalmente os algoritmos estudados, listamos no Código 1 os códigos em `Julia` do algoritmo Euclidiano.

Código 1.1 Código em `Julia` do algoritmo Euclidiano

```
function euclidiano(a, b)
    while b != 0
        resto = a % b
        a = b
        b = resto
    end
    return a
end
```

1.3 ALGORITMO EUCLIDIANO ESTENDIDO

Existe uma relação muito interessante chamada Identidade de Bézout, a qual demonstraremos no Teorema 1.17, que diz que dados inteiros positivos

a e b, existem α, β inteiros tais que

$$\alpha \cdot a + \beta \cdot b = \text{mdc}(a, b).$$

A priori, pensamos que devemos começar encontrando $\text{mdc}(a, b)$ pelo algoritmo Euclidiano, tentando, posteriormente, calcular α e β. Assim, α e β poderiam ser expressos pelos valores obtidos nas divisões que efetuamos. Porém, este método é muito trabalhoso e seria ineficaz para o método computacional, pois o computador teria de armazenar muita informação e números muito grandes, o que torna tudo muito inconveniente ou até impossível.

O algoritmo Euclidiano estendido permite que isto seja feito muito mais facilmente. A ideia consiste em expressar os restos obtidos como $\alpha \cdot a + \beta \cdot b = r_k$, sem esperar chegar ao último resto. A vantagem é que para efetuar os cálculos correspondentes a uma certa divisão, precisamos apenas dos dados referentes às duas últimas divisões. As demais podem ser apagadas liberando memória.

Suponha que, calculando o $\text{mdc}(a, b)$, obtemos a sequência:

$$\begin{aligned}
a &= b \cdot q_1 + r_1, & 0 \leq r_1 < b \\
b &= r_1 \cdot q_2 + r_2, & 0 \leq r_2 < r_1 \\
r_1 &= r_2 \cdot q_3 + r_3, & 0 \leq r_3 < r_2 \\
r_2 &= r_3 \cdot q_4 + r_4, & 0 \leq r_4 < r_3 \\
&\vdots & \vdots \\
r_{n-2} &= r_{n-1} \cdot q_n, & r_n = 0,
\end{aligned}$$

a qual reescrevemos como

$$\begin{aligned} a &= b \cdot q_1 + r_1, & r_1 &= ax_1 + by_1 \\ b &= r_1 \cdot q_2 + r_2, & r_2 &= ax_2 + by_2 \\ r_1 &= r_2 \cdot q_3 + r_3, & r_3 &= ax_3 + by_3 \\ r_2 &= r_3 \cdot q_4 + r_4, & r_4 &= ax_4 + by_4 \\ &\vdots & &\vdots \\ r_{n-2} &= r_{n-1} \cdot q_n, & r_n &= 0 \end{aligned}$$

em que $x_1, x_2, \ldots, x_{n-1}$ e $y_1, y_2, \ldots, y_{n-1}$ são inteiros a determinar. Vamos colocar as informações em um quadro

restos	quocientes	x	y
a	$*$	x_{-1}	y_{-1}
b	$*$	x_0	y_0
r_1	q_1	x_1	y_1
r_2	q_2	x_2	y_2
r_3	q_3	x_3	y_3
$\vdots$	$\vdots$	$\vdots$	$\vdots$
r_{n-2}	q_{n-2}	x_{n-2}	y_{n-2}
r_{n-1}	q_{n-1}	x_{n-1}	y_{n-1}

As duas primeiras linhas da tabela não correspondem a nenhuma divisão, pois nem a e nem b são restos, mas vamos precisar delas. Chamaremonas de linhas -1 e 0.

Queremos descobrir como preencher as colunas x e y. Suponha, então, que já recebemos a tabela preenchida até a $(j-1)$-ésima linha. Começamos a preencher a j-ésima linha dividindo r_{j-2} por r_{j-1} para encontrar r_j e q_j tal que $r_{j-2} = r_{j-1} \cdot q_j + r_j$ e $0 \leq r_j < r_{j-1}$. Assim

$$r_j = r_{j-2} - r_{j-1} \cdot q_j. \tag{1.2}$$

Nas linhas $j-1$ e $j-2$ os valores de x_{j-1}, x_{j-2} e y_{j-1}, y_{j-2} são como:

$$r_{j-2} = ax_{j-2} + by_{j-2} \quad \text{e} \quad r_{j-1} = ax_{j-1} + by_{j-1}.$$

Substituindo em (1.2) temos

$$r_j = (ax_{j-2} + by_{j-2}) - (ax_{j-1} + by_{j-1}) \cdot q_j$$
$$r_j = a(x_{j-2} - q_j \cdot x_{j-1}) + b(y_{j-2} - q_j \cdot y_{j-1}).$$

Assim,

$$x_j = x_{j-2} - q_j \cdot x_{j-1} \quad \text{e} \quad y_j = y_{j-2} - q_j \cdot y_{j-1}.$$

Note que, para calcular x_j e y_j, precisamos apenas das divisões efetuadas nas duas linhas anteriores e de q_j. Só precisamos saber agora como calcular as linhas -1 e 0. Se interpretarmos as linhas como as demais, temos:

$$a = ax_{-1} + by_{-1} \quad \text{e} \quad b = ax_0 + by_0$$

que nos sugere escolher $x_{-1} = 1$, $y_{-1} = 0$, $x_0 = 0$, $y_0 = 1$. Tendo executado o algoritmo e descoberto que o máximo divisor comum é r_{n-1}, obtemos

$$d = r_{n-1} = ax_{n-1} + by_{n-1}.$$

Ou seja, $\alpha = x_{n-1}$ e $\beta = y_{n-1}$.

Observação 1.13. Para fim de facilitar a explicação deste algoritmo suponha $a > b$.

Apresentamos abaixo o pseudo-código do algoritmo Euclidiano estendido.

Algoritmo 1.3 Algoritmo Euclidiano estendido

1: **procedure** ESTENDIDO(a, b)▷ Entrada do algoritmo, inteiros positivos a e b
2: $x = 0$; $x_1 = 1$ ▷ Entradas necessárias para o algoritmo começar a funcionar
3: $y = 1$; $y_1 = 0$
4: $r = b$; $r_1 = a$
5: **while** $r \neq 0$ **do** ▷ Se r for 0, a resposta será $x = 0$ e $y = 1$ pois $\text{mdc}(a, b) = b$
6: $q \leftarrow$ DIVISAO(r_1, r)
7: $(r_1, r) \leftarrow (r, r_1 - q \cdot r)$
8: $(x_1, x) \leftarrow (x, x_1 - q \cdot x)$
9: $(y_1, y) \leftarrow (y, y_1 - q \cdot y)$
10: **return** x_1, y_1 ▷ α e β são x_1, y_1

Vejamos um exemplo da aplicação do Algoritmo 1.3.

Exemplo 1.14. Tome $a = 1234$ e $b = 54$ como no Exemplo 1.10.

restos	quocientes	x	y
1234	*	1	0
54	*	0	1
46	22	$1 - 22 \cdot 0 = 1$	$0 - 22 \cdot 1 = -22$
8	1	$0 - 1 \cdot 1 = -1$	$1 - 1 \cdot (-22) = 23$
6	5	$1 - 5 \cdot (-1) = 6$	$-22 - 5 \cdot 23 = -137$
2	1	$-1 - 1 \cdot 6 = -7$	$23 - 1 \cdot (-137) = 160$
0	3	*	*

Portanto, $\alpha = -7$, $\beta = 160$, assim $(-7) \cdot 1234 + 160 \cdot 54 = 2$, 2 é o $\text{mdc}(1234, 56)$.

Vamos demonstrar agora a Identidade de Bézout, mas para isto, precisaremos de dois resultados auxiliares, estabelecido nos lemas a seguir.

Lema 1.15. *Se b é divisor de a, então, b divide $k \cdot a$, com a, b, $k \in \mathbb{Z}$.*

Demonstração. Suponha que b divide a, então, $\exists\ n \in \mathbb{Z}$ tal que $b = a \cdot n$, assim $k \cdot b = k \cdot (a \cdot n) = a \cdot (k \cdot n) = a \cdot m$, $m = (k \cdot n)$, com $n \in \mathbb{Z}$, portanto, $k \cdot b$ divide a. ■

Lema 1.16. *Sejam a, b e d inteiros positivos, se d é divisor de a e b então d divide a soma $a + b$.*

Demonstração. Se d divide a e b, então $a = d \cdot k$ e $b = d \cdot j$ com k e j inteiros positivos. Assim $a + b = d \cdot k + d \cdot j$, e colocando d em evidência temos que $a + b = d \cdot (k + j)$. Portanto d divide a soma como queríamos. ■

Teorema 1.17 (Identidade de Bézout)**.** *Sejam a e b inteiros positivos e seja $d = \text{mdc}(a, b)$. Existem α e β tais que $\alpha \cdot a + \beta \cdot b = d$.*

Demonstração. Dados a, b inteiros positivos e $b \neq 0$, considere o conjunto

$$S = \{\alpha \cdot a + \beta \cdot b \mid \alpha, \beta \in \mathbb{Z} \text{ e } \alpha \cdot a + \beta \cdot b \geq 0\}.$$

Suponha $d' = \alpha_0 \cdot a + \beta_0 \cdot b$ o menor elemento positivo de S, pelo princípio da boa ordenação. Dado $n \in S$, podemos escrever $n = \alpha_1 \cdot a + \beta_1 \cdot b$. Mas pelo Teorema da divisão, existem q e r inteiros positivos tais que podemos escrever $n = q \cdot d + r$ em que $0 \leq r < d$ e, isolando r, temos que $n - q \cdot b = r$. Portanto, $n - q \cdot d = (\alpha_1 \cdot a + \beta_1 \cdot b) - q \cdot (\alpha_0 \cdot a + \beta_0 \cdot b) = a \cdot (\alpha_1 - q \cdot \alpha_0) + b \cdot (\beta_1 - q \cdot \beta_0) = r$ tal que r pertence a S. Mas como d' é o menor inteiro positivo em S, $r = 0$. Logo, podemos escrever S como

$$S = \{k \cdot d' \mid k \geq 0\}. \tag{1.3}$$

Por hipótese, $d = \text{mdc}(a, b)$. Portanto, pelo Lema 1.15 d divide $\alpha_0 \cdot a$ e d divide $\beta_0 \cdot b$, assim, pelo Lema 1.16, d divide $\alpha_0 \cdot a + \beta_0 \cdot b$, ou seja, d divide d'. Destarte, $d \leq d'$.

Por outro lado, note que a, $b \in S$, pois podem ser escritos como $a = 1 \cdot a + 0 \cdot b$ e $b = 0 \cdot a + 1 \cdot b$. Daí, usando Eq. (1.3), temos que d' divide a e d' divide b. Como d' divide a e b, temos que d' é um divisor comum de a e b. Mas, como $d = \text{mdc}(a, b)$, temos que $d \geq d'$.

Como $d \leq d'$ e $d \geq d'$, concluímos que $d' = d$. Portanto, sempre existem α e β tais que $\alpha \cdot a + \beta \cdot b = d$. ■

Apresentamos agora o código em `Julia` do algoritmo Euclidiano estendido.

Código 1.2 Código em `Julia` do algoritmo Euclidiano estendido.

```
function estendido(a, b)
    x = 0 ; x1 = 1
    y = 1 ; y1 = 0
    r = b ; r1 = a
    while r !=0
        quociente = div(r1, r)
        (r1, r) = (r, r1 - quociente * r)
        (x1, x) = (x, x1 - quociente * x)
        (y1, y) = (y, y1 - quociente * y)
    end
    return x1, y1
end
```

2 O CORPO $\mathbb{Z}_p$

Nesse capítulo definiremos formalmente as estruturas algébricas denominadas anel, anel comutativo, anel comutativo com unidade e corpo. Além disso, provaremos algumas proposições e teoremas necessários. Por fim, mostraremos que $\mathbb{Z}_p$ cumpre as propriedades necessárias para ser um corpo e que suas operações estão bem definidas. Para tal, usaremos como base as obras [5] e [9].

2.1 ANEL

Definição 2.1. Seja $\mathbb{A}$ um conjunto não-vazio, munido de duas operações que chamaremos de adição e multiplicação, dadas respectivamente por:

$$\begin{array}{rccc} + : & \mathbb{A}\times\mathbb{A} & \to \mathbb{A} \\ & (a,b) & \mapsto a+b \end{array} \qquad \begin{array}{rccc} \cdot : & \mathbb{A}\times\mathbb{A} & \to \mathbb{A} \\ & (a,b) & \mapsto a\cdot b. \end{array}$$

Dizemos que $(\mathbb{A}, +, \cdot)$ é um *anel* se $\forall\ a,\ b,\ c \in \mathbb{A}$ valem as seguintes propriedades:

(A1) Associatividade da adição: $(a+b)+c = a+(b+c)$.

(A2) Comutatividade da adição: $a+b = b+a$.

(A3) Existência do elemento neutro da adição: $\exists\ 0_{\mathbb{A}} \in \mathbb{A}$ tal que $a+0_{\mathbb{A}} = a$.

(A4) Existência do elemento oposto da adição: $\exists\ b \in \mathbb{A}$ tal que $a+b = 0_{\mathbb{A}}$.

(A5) Associatividade em relação à multiplicação: $(a\cdot b)\cdot c = a\cdot(b\cdot c)$.

(A6) Distributividade: $a\cdot(b+c) = a\cdot b + a\cdot c$ e $(a+b)\cdot c = a\cdot c + b\cdot c$.

Exemplo 2.2. O conjunto dos inteiros munido das operações de soma e multiplicação usuais $(\mathbb{Z}, +, \cdot)$, é um anel.

Exemplo 2.3. São anéis $(\mathbb{Q}, +, \cdot)$, $(\mathbb{R}, +, \cdot)$ e $(\mathbb{C}, +, \cdot)$.

Definição 2.4. Um anel $(\mathbb{A}, +, \cdot)$, é chamado *anel comutativo*, se $\forall a, b \in \mathbb{A}$ vale a seguinte propriedade:

(A7) Comutatividade na multiplicação: $a \cdot b = b \cdot a$.

Exemplo 2.5. São anéis comutativos $(\mathbb{Z}, +, \cdot)$, $(\mathbb{Q}, +, \cdot)$, $(\mathbb{R}, +, \cdot)$ e $(\mathbb{C}, +, \cdot)$

Definição 2.6. Um anel comutativo $(\mathbb{A}, +, \cdot)$, é chamado *anel comutativo com unidade*, se $\forall a \in \mathbb{A}$, vale a seguinte propriedade:

(A8) Elemento neutro da multiplicação: $\exists\ 1_{\mathbb{A}} \in \mathbb{A}$ tal que $a \cdot 1_{\mathbb{A}} = a$.

Exemplo 2.7. O anel dos inteiros munido das operações usuais de soma e multiplicação é um anel comutativo com unidade.

2.1.1 Relações de equivalência

Definição 2.8. Seja A um conjunto não-vazio e seja $\sim$ uma relação entre os elementos de A. Dizemos que $\sim$ é uma *relação de equivalência* se, e somente se, valem as seguintes propriedades:

i) $a \sim a$, $\forall\ a \in A$. (Reflexiva).

ii) $a \sim b \Rightarrow b \sim a$, $\forall\ a, b \in A$. (Simétrica).

iii) $a \sim b$ e $b \sim c \Rightarrow a \sim c$, $\forall\ a, b, c \in A$. (Transitiva).

Exemplo 2.9. A relação de igualdade no conjunto $\mathbb{R}$ é uma relação de equivalência.

i) $a = a$, $\forall\ a \in \mathbb{R}$.

ii) $a = b \Rightarrow b = a$, $\forall\ a, b \in \mathbb{R}$.

iii) $a = b$ e $b = c \Rightarrow a = c$, $\forall\ a, b, c \in \mathbb{R}$.

Definição 2.10. Seja A um conjunto e $\sim$ uma relação de equivalência definida em A. Se existir $a \in A$, então, a *classe de equivalência* de a é o conjunto dos elementos de A que são equivalentes a a por $\sim$. Denotaremos por $\overline{a}$ a classe de equivalência de a. Em símbolos: $\overline{a} = \{b \in A \mid b \sim a\}$.

Exemplo 2.11. Dado um conjunto B de balas sortidas que possui 11 balas de uva, qualquer uma das balas pode representar a classe das de uva. Isto é, se conhecemos um elemento da classe, podemos reconstruir a classe inteira.

Exemplo 2.12. Dados a e b inteiros, com b não nulo, o conjunto das frações $\frac{a}{b}$ é uma classe de equivalência de números, veja bem:

$$\frac{a}{b} = \frac{2a}{2b} = \frac{3a}{3b} = \frac{4a}{4b} = \dots$$

Proposição 2.13. *Dado um conjunto A com a relação de equivalência $\sim$ e dado $\overline{a}$ uma classe de equivalência de A. Se existir $b \in A$ tal que $b \in \overline{a}$, então, $\overline{a} = \overline{b}$.*

Demonstração. Fixe um $b \in A$, com $b \in \overline{a}$. Por definição, $b \sim a$; pela propriedade simétrica $a \sim b$. Dado $c \in A$ tal que $c \in \overline{a}$, então, $c \sim a$. Logo, pela propriedade transitiva, $c \sim b$. Portanto, $c \in \overline{b}$. Assim, $\overline{a} \subset \overline{b}$.
Por outro lado, dado $d \in A$ tal que $d \in \overline{b}$ então, $d \sim b$. Como visto, $b \sim a$ e pela propriedade transitiva $d \sim a$. Portanto, $d \in \overline{a}$. Assim, $\overline{b} \subset \overline{a}$.
Portanto, como $\overline{a} \subset \overline{b}$ e $\overline{b} \subset \overline{a}$, temos que $\overline{a} = \overline{b}$. ■

Proposição 2.14. *Duas classes de equivalência distintas não podem ter um elemento em comum.*

Demonstração. Dado A um conjunto, munido da relação de equivalência $\sim$, em que $\overline{a}$ e $\overline{b}$ classes de equivalência de A.
Seja $c \in \overline{a} \cap \overline{b}$. Como $c \in \overline{a}$, pelo resultado anterior $\overline{a} = \overline{c}$. Analogamente, se $c \in \overline{b}$, pelo resultado anterior $\overline{b} = \overline{c}$. Assim, $\overline{a} = \overline{c}$ e $\overline{b} = \overline{c}$, portanto, $\overline{a} = \overline{c} = \overline{b}$, ou seja, $\overline{a} = \overline{b}$ como queríamos provar. ■

Proposição 2.15. *Um conjunto A qualquer é a reunião de todas as suas classes de equivalência.*

Demonstração. Queremos mostrar que A é a união de todas as classes de equivalência, ou seja, que

$$A = \bigcup_{a \in A} \overline{a}.$$

Seja $b \in A$, pela propriedade reflexiva $b \sim b$, então, $b \in \overline{b}$, e $\overline{b} \in \bigcup_{a\in A} \overline{a}$, logo, A está contido em $\bigcup_{a\in A} \overline{a}$.
Seja $c \in \bigcup_{a\in A} \overline{a}$, logo, c pertence a alguma classe de equivalência $\overline{c}$, isto implica pela definição de classes que $c \in A$. Assim, $\bigcup_{a\in A} \overline{a}$ está contido em A.
Portanto, como $A \subseteq \bigcup_{a\in A} \overline{a}$ e $\bigcup_{a\in A} \overline{a} \subseteq A$ temos que $A = \bigcup_{a\in A} \overline{a}$. ■

Definição 2.16. O conjunto das classes de equivalência de $\sim$ em A é chamado *conjunto quociente* de A por $\sim$, denotado por $A/\sim$.

Observação 2.17. Os elementos do conjunto quociente são subconjuntos das classes de equivalência de A. Isto é, o conjunto quociente não é um subconjunto de A, mas sim do conjunto das partes de A.

2.1.2 Anel dos inteiros módulo n

Definição 2.18. Dados $a, b \in \mathbb{Z}$, e $n \neq 0$ a e b são congruentes módulo n se $a - b$ é um múltiplo de n, ou seja, $a - b$ é divisível por n, em símbolos $n \mid (a - b)$. Denotaremos a congruência de a e b por n por $a \equiv b \pmod{n}$.

Exemplo 2.19. Dado $n = 5$, $10 \equiv 0 \pmod 5$, pois $\frac{10-0}{5} = 2$ e $14 \equiv 24 \pmod 5$, pois $\frac{14-24}{5} = -2$.

Exemplo 2.20. Dado $n = 11$, $29 \equiv 18 \pmod{11}$, pois $\frac{29-18}{11} = 1$ e $29 \equiv 106 \pmod{11}$, pois $\frac{29-106}{11} = -7$.

Proposição 2.21. *A congruência módulo n é uma relação equivalência sobre $\mathbb{Z}$.*

Demonstração. Para provar uma equivalência, basta que valham as propriedades reflexiva, simétrica e transitiva:

i) Reflexiva: Provaremos que $a \equiv a \pmod{n}$. Seja a um inteiro, assim, $a - a = 0$, logo, $a \equiv a \pmod{n}$, pois 0 é múltiplo de todos os inteiros.

ii) Simétrica: Se $a \equiv b \pmod{n}$, então, $a - b$ é um múltiplo de n. Mas $b - a = -(a - b)$. Logo, $b - a$ também é múltiplo de n. Portanto, $b \equiv a \pmod{n}$.

iii) Transitiva: Dados a, b, c inteiros, suponha $a \equiv b \pmod{n}$ e $b \equiv c \pmod{n}$. Assim, $n \mid (a-b)$ e $n \mid (b-c)$, ou seja, $a-b$ e $b-c$ são múltiplos de n. Então, pelo Lema 1.16 $(a - b) + (b - c) = a - b + b - c = a - c$ é múltiplo de n, logo, $a \equiv c \pmod{n}$.

Portanto, está provado que a congruência módulo n é uma relação de equivalência. ■

O conjunto em que estamos interessados é o conjunto quociente $\mathbb{Z}$ pela relação de equivalência módulo n. Então, vamos defini-lo.

Definição 2.22. O conjunto quociente de $\mathbb{Z}$, pela relação de congruência módulo n, é o *conjunto de inteiros módulo n* sendo denotado por $\mathbb{Z}_n$.

Observação 2.23. Dado $a \in \mathbb{Z}$, $\overline{a} = \{a + k \cdot n, k \in \mathbb{Z}\}$. Em particular $\overline{0}$ é o conjunto dos múltiplos de n. Como $\mathbb{Z}_n$ é o conjunto de todas as classes de equivalência de módulo n, então, $\overline{a} \in \mathbb{Z}_n$.

Considere $a \in \mathbb{Z}$. Pelo algoritmo da divisão temos que:

$$a = n \cdot q + r, \text{ com } q, r \in \mathbb{Z} \text{ e } 0 \leq r \leq n - 1.$$

Então, $a - r = n \cdot q$ por conseguinte, $n \mid r - a$ logo, $r \equiv a \pmod{n}$. Isto é, um inteiro qualquer é congruente módulo n a outro inteiro no intervalo que vai de $\overline{0}$ a $\overline{n-1}$. Além disso, duas destas classes não podem ser iguais: a única maneira de dois números entre 0 e $n - 1$ serem congruentes módulo n é se forem iguais. Ou seja: $\mathbb{Z}_n = \{\overline{0}, \overline{1}, \ldots, \overline{n-1}\}$.

Exemplo 2.24. $\mathbb{Z}_2 = \{\overline{0}, \overline{1}\}$, $\overline{0}$ é a classe dos pares e $\overline{1}$ é a classe dos ímpares, ou seja, $\overline{0} = \{a \in \mathbb{Z} \mid a \equiv \overline{0} \pmod{2}\}$ e $\overline{1} = \{b \in \mathbb{Z} \mid b \equiv \overline{1} \pmod{2}\}$. Assim, $\overline{0} = \overline{2} = \overline{4} = \overline{6} = \ldots$ e $\overline{1} = \overline{3} = \overline{5} = \overline{7} = \ldots$.

Exemplo 2.25. $\mathbb{Z}_7 = \{\overline{0}, \overline{1}, \overline{2}, \overline{3}, \overline{4}, \overline{5}, \overline{6}\}$. Em que:

- $\overline{0} = \overline{7} = \overline{14} = \cdots$
- $\overline{1} = \overline{8} = \overline{15} = \cdots$
- $\overline{2} = \overline{9} = \overline{16} = \cdots$
- $\overline{3} = \overline{10} = \overline{17} = \cdots$
- $\overline{4} = \overline{11} = \overline{18} = \cdots$
- $\overline{5} = \overline{12} = \overline{19} = \cdots$
- $\overline{6} = \overline{13} = \overline{20} = \cdots$

Dado o conjunto $\mathbb{Z}_n$, definimos as operações

$$\begin{array}{rccc} + : & \mathbb{Z}_n \times \mathbb{Z}_n & \to \mathbb{Z}_n & \qquad \times : \quad \mathbb{Z}_n \times \mathbb{Z}_n \quad \to \mathbb{Z}_n \\ & (a,b) & \mapsto \overline{a} + \overline{b} & \qquad\qquad (a,b) \quad\quad \mapsto \overline{a} \cdot \overline{b}. \end{array}$$

em que

i) $\overline{a} + \overline{b} = \overline{a+b}$;

ii) $\overline{a} \cdot \overline{b} = \overline{a \cdot b}$.

Provaremos que estas operações estão bem definidas.

Proposição 2.26. *Dados $\overline{a}, \overline{b} \in \mathbb{Z}_n$, se $\overline{a} = \overline{a'}$ e $\overline{b} = \overline{b'}$, então, $\overline{a+b} = \overline{a'+b'}$.*

Demonstração. Note que, $\overline{a} = \overline{a'}$ nos garante que $\overline{a} - \overline{a'} = n \cdot r_1$ enquanto $\overline{b} = \overline{b'}$ nos diz que $\overline{b} - \overline{b'} = n \cdot r_2$ para r_1, r_2 inteiros apropriados. Isto implica que $(a - a') + (b - b') = n \cdot r_1 + n \cdot r_2 = n \cdot (r_1 + r_2)$ e $(a + b) - (a' + b') = n \cdot (r_1 + r_2)$. Por conseguinte, $(a' - a) + (b' - b) = (a + b) - (a' + b')$. Assim, $\overline{a+b} = \overline{a'+b'}$. ■

Proposição 2.27. *Dados $\overline{a}, \overline{b} \in \mathbb{Z}_n$, se $\overline{a} = \overline{a'}$ e $\overline{b} = \overline{b'}$, então, $\overline{a \cdot b} = \overline{a' \cdot b'}$.*

Demonstração. De fato, $\overline{a} = \overline{a'}$ nos garante que $\overline{a} - \overline{a'} = n \cdot r_1$ enquanto $\overline{b} = \overline{b'}$ nos diz que $\overline{b} - \overline{b'} = n \cdot r_2$ para r_1, r_2 inteiros apropriados. Isto implica que $a = a' + n \cdot r_1$ e $b = b' + n \cdot r_2$. Multiplicando temos: $a \cdot b = (a' + n \cdot r_1) \cdot (b' + n \cdot r_2) = a' \cdot b' + (a' \cdot r_2 + b' \cdot r_1 + r_1 \cdot r_2 \cdot n) \cdot n$. Assim, $a \cdot b - a' \cdot b'$ é um múltiplo de n. Portanto, $\overline{a \cdot b} = \overline{a' \cdot b'}$. ■

Teorema 2.28. *O conjunto $\mathbb{Z}_n$ munido das operações*

i) $\overline{a} + \overline{b} = \overline{a + b}$, *e*

ii) $\overline{a} \cdot \overline{b} = \overline{a \cdot b}$,

denotado por $(\mathbb{Z}_n, +, \cdot)$, é um anel comutativo com unidade.

Demonstração. Já provamos que as operações estão bem definidas, agora provaremos que valem as propriedades da Definição 2.6. Sejam, $\overline{a}$, $\overline{b}$, $\overline{c} \in \mathbb{Z}_n$.

(A1) $(\overline{a} + \overline{b}) + \overline{c} = \overline{(a + b)} + \overline{c} = \overline{(a + b) + c} = \overline{a + (b + c)} = \overline{a} + \overline{(b + c)} = \overline{a} + (\overline{b} + \overline{c})$.

(A2) $\overline{a} + \overline{b} = \overline{a + b} = \overline{b + a} = \overline{b} + \overline{a}$.

(A3) $\overline{a} + \overline{0} = \overline{a + 0} = \overline{a}$ e $\overline{0} + \overline{a} = \overline{0 + a} = \overline{a}$.

(A4) $\overline{a} + (\overline{-a}) = \overline{a + (-a)} = \overline{0}$ e $\overline{-a} + \overline{a} = \overline{-a + a} = \overline{0}$.

(A5) $(\overline{a} \cdot \overline{b}) \cdot \overline{c} = \overline{(a \cdot b) \cdot c} = \overline{a \cdot (b \cdot c)} = \overline{a} \cdot \overline{(b \cdot c)} = \overline{a} \cdot (\overline{b} \cdot \overline{c})$.

(A6) $\overline{a} \cdot (\overline{b} + \overline{c}) = \overline{a} \cdot \overline{(b + c)} = \overline{a \cdot (b + c)} = \overline{a \cdot b + a \cdot c} = \overline{a \cdot b} + \overline{a \cdot c} = \overline{a} \cdot \overline{b} + \overline{a} \cdot \overline{c}$.

(A7) $\overline{a} \cdot \overline{b} = \overline{a \cdot b} = \overline{b \cdot a} = \overline{b} \cdot \overline{a}$.

(A8) $\overline{a} \cdot \overline{1} = \overline{a \cdot 1} = \overline{a}$ e $\overline{1} \cdot \overline{a} = \overline{1 \cdot a} = \overline{a}$.

■

2.1.3 Divisão modular

Nosso objetivo aqui é calcular divisões em $\mathbb{Z}_n$. Nos números reais, dados a e b, $b \neq 0$, a frase "dividir a por b" é equivalente a "multiplicar a pelo inverso de b", denotado por $\frac{1}{b}$, em que $b \cdot \frac{1}{b} = 1$. Como já mencionado, $b \neq 0$, pois 0 não possui inverso. Usaremos o mesmo raciocínio para $\mathbb{Z}_n$.

Definição 2.29. Dado $\overline{a} \in \mathbb{Z}_n$, $\overline{a} \neq \overline{0}$. Diremos que a classe $\overline{\alpha} \in \mathbb{Z}_n$ é o *inverso de* $\overline{a}$ se $\overline{a} \cdot \overline{\alpha} = \overline{1}$.

Vamos à questão de executar divisões em $\mathbb{Z}_n$. Se queremos dividir $\overline{b}$ por $\overline{a}$ precisamos fazer $\overline{b}$ multiplicado pelo inverso de $\overline{a}$. Logo, se o inverso de $\overline{a}$ não existir em $\mathbb{Z}_n$ não é possível efetuar a divisão. Por isso, precisamos que n e a sejam primos entre si.

Proposição 2.30. *A classe $\overline{a}$ possui inverso em $\mathbb{Z}_n$ se, e somente se, a e n são primos entre si.*

Demonstração. Suponhamos que a classe $\overline{a} \neq \overline{0}$ possua inverso $\overline{b} \neq \overline{0}$ em $\mathbb{Z}_n$. Então, $\overline{a} \cdot \overline{b} = \overline{1}$. Logo, $\overline{a} \cdot \overline{b} = \overline{a \cdot b} = \overline{1}$ o que quer dizer que $a \cdot b \equiv 1 \pmod{n}$ então, $a \cdot b - 1 = k \cdot n$ assim, $a \cdot b - k \cdot n = 1$, para algum $k \in \mathbb{Z}$. Seja $d \geq 1$ um divisor de n e de a. Se, $d \mid a$ então, $d \mid a \cdot b$ e se $d \mid n$ então, $d \mid k \cdot n$ e, portanto, $d \mid (a \cdot b - k \cdot n)$, ou seja, $d \mid 1$, logo, $d = 1$. Destarte, conclui-se que $\text{mdc}(a, n) = 1$.
Por outro lado, suponha agora que $\text{mdc}(a, n) = 1$. Pelo Teorema 1.17, existem r e s, inteiros, tais que $r \cdot a + s \cdot n = 1$. Assim, $\overline{1} = \overline{r \cdot a + s \cdot n} = \overline{r} \cdot \overline{a} + \overline{s} \cdot \overline{n}$, mas como estamos em $\mathbb{Z}_n$, $\overline{n} = \overline{0}$. Então, $\overline{1} = \overline{r} \cdot \overline{a} + \overline{s} \cdot \overline{0} = \overline{r} \cdot \overline{a} + \overline{0} = \overline{r} \cdot \overline{a}$. Logo, como $\overline{r} \cdot \overline{a} = \overline{1}$, $\overline{a}$ possui inverso em $\mathbb{Z}_n$. ■

Observação 2.31. A classe $\overline{0}$ não possui inverso, pois para que uma classe a tenha inverso multiplicativo em $\mathbb{Z}_n$, $\text{mdc}(a, p) = 1$ e isso não acontece quando $a = 0$.

Exemplo 2.32. Se queremos dividir $\overline{2}$ por $\overline{3}$ em $\mathbb{Z}_8$. Precisamos encontrar o inverso de $\overline{3}$ em $\mathbb{Z}_8$ e executar a divisão calculando o inverso de $\overline{3}$,

multiplicado por $\overline{2}$. Temos que $\text{mdc}(3,8) = 1$, logo, $\overline{3}$ tem inverso em $\mathbb{Z}_8$. Como neste caso os números são pequenos podemos encontrar o inverso por tentativa e erro, $\overline{3} \cdot \overline{1} = \overline{3}$, $\overline{3} \cdot \overline{2} = \overline{6}$, $\overline{3} \cdot \overline{3} = \overline{1}$, neste caso $\overline{3}$ é o próprio inverso de $\overline{3}$. Assim, o resultado da divisão de $\overline{2}$ por $\overline{3}$ em $\mathbb{Z}_8$ é $\overline{6}$.

Exemplo 2.33. Qual o inverso de $\overline{3}$ em $\mathbb{Z}_{11}$?
Primeiro devemos verificar se o inverso existe, $\text{mdc}(3,11) = 1$, portanto, o inverso existe. Agora aplicando o Algoritmo Estendido temos $3 \cdot 4 - 11 = 1$ que é equivalente a $\overline{3} \cdot \overline{4} = \overline{1}$ em $\mathbb{Z}_{11}$. Assim, $\overline{4}$ é o inverso de $\overline{3}$ em $\mathbb{Z}_{11}$.

Exemplo 2.34. Verificaremos agora o inverso de $\overline{3}$ em $\mathbb{Z}_{12}$. Note que $\text{mdc}(3,12) = 3$, logo, não existe nenhum $\overline{a} \in \mathbb{Z}_{12}$ tal que $\overline{a} \cdot \overline{3} = \overline{1}$. Vamos verificar:

- $\overline{3} \cdot \overline{0} = \overline{0}$
- $\overline{3} \cdot \overline{4} = \overline{0}$
- $\overline{3} \cdot \overline{8} = \overline{0}$
- $\overline{3} \cdot \overline{1} = \overline{3}$
- $\overline{3} \cdot \overline{5} = \overline{3}$
- $\overline{3} \cdot \overline{9} = \overline{3}$
- $\overline{3} \cdot \overline{2} = \overline{6}$
- $\overline{3} \cdot \overline{6} = \overline{6}$
- $\overline{3} \cdot \overline{10} = \overline{6}$
- $\overline{3} \cdot \overline{3} = \overline{9}$
- $\overline{3} \cdot \overline{7} = \overline{9}$
- $\overline{3} \cdot \overline{11} = \overline{9}$

De fato, note que efetuaremos as multiplicações de $\overline{3}$ com todas as classes de equivalência e não encontramos nenhum $\overline{a}$ tal que $\overline{3} \cdot \overline{a} = \overline{1}$.

Observação 2.35. Nem sempre o inverso existe, por exemplo, a classe $\overline{3}$ em $\mathbb{Z}_{12}$ não existe.

Exemplo 2.36. Vamos verificar os inversos de todas as classes de equivalência em $\mathbb{Z}_3$.

- $\overline{0} \cdot \overline{0} = \overline{0}$
- $\overline{0} \cdot \overline{2} = \overline{0}$
- $\overline{1} \cdot \overline{2} = \overline{2}$
- $\overline{0} \cdot \overline{1} = \overline{0}$
- $\overline{1} \cdot \overline{1} = \overline{1}$
- $\overline{2} \cdot \overline{2} = \overline{1}$

Veja que em $\mathbb{Z}_3$ todos os elementos possuem inverso. O inverso de $\overline{1}$ é o próprio $\overline{1}$ e o inverso de $\overline{2}$ é o próprio $\overline{2}$.

Um conjunto como o do Exemplo 2.36 onde todos os elementos possuem inverso recebe o nome de corpo. Vamos, então, defini-lo.

2.2 CORPO

Definição 2.37. Um anel comutativo com unidade $(\mathbb{K}, +, \cdot)$ é chamado *corpo*, se $\forall\ a \in \mathbb{K}$, com $a \neq 0$, vale também a seguinte propriedade:

(A10) Existência do inverso multiplicativo: $\exists\ b \in \mathbb{K}$ tal que $a \cdot b = 1$.

Como vimos nos exemplos anteriores, quando $\overline{a} \in \mathbb{Z}_n$ é tal que $\text{mdc}(a, n) = 1$, então $\overline{a}$ tem inverso em $\mathbb{Z}_n$. De fato, o teorema a seguir confirma que isso vai acontecer, uma vez que uma condição necessária e suficiente para que $\mathbb{Z}_n$ seja um corpo é de que n seja primo. Neste caso, usaremos a notação $\mathbb{Z}_p$, para p primo.

Teorema 2.38. *O conjunto $\mathbb{Z}_p$ é um corpo se, e somente se, p é primo.*

Demonstração. Seja $\mathbb{Z}_n$ um corpo, com n um número composto, digamos $n = a \cdot q$, com $a \neq 0$ e $a \neq 1$. Logo, $\text{mdc}(a, n) = a$. Considere a classe $\overline{a} \in \mathbb{Z}_n$, não nula, e seja $\overline{b} \neq \overline{0}$ em $\mathbb{Z}_n$ seu inverso. Então, $\overline{a} \cdot \overline{b} = \overline{1}$. Logo, $\overline{a} \cdot \overline{b} = \overline{a \cdot b} = \overline{1}$ de modo que, $a \cdot b \equiv 1 \pmod{n}$ então, $a \cdot b - 1 = k \cdot n$. Portanto, $a \cdot b - k \cdot n = 1$, para algum $k \in \mathbb{Z}$. Seja $d \geq 1$ um divisor de n e de a. Destarte, $d \mid (a \cdot b - k \cdot n)$, ou seja, $d \mid 1$, logo, $d = 1$. Portanto, conclui-se que $\text{mdc}(a, n) = 1$. Contradição, pois, a é divisor de n, então, $\text{mdc}(a, n)$ deveria ser a. Logo, concluímos que o único jeito de $\mathbb{Z}_n$ ser um corpo é se n for primo, visto que não ocorreria tal contradição. Pois $\text{mdc}(a, p) = 1$ para todo p primo.
Por outro lado, seja $\overline{a} \in \mathbb{Z}_p$, com p primo e $\overline{a} \neq \overline{0}$. Pelo Teorema 2.28, temos que $\mathbb{Z}_p$ é um anel comutativo com unidade. Basta que provemos que $\mathbb{Z}_p$ cumpre a propriedade $(A10)$.

Provaremos agora a existência do inverso multiplicativo de $\overline{a}$. Note que, como $\text{mdc}(a, p) = 1$ pela Proposição 2.30, $\overline{a}$ tem inverso em $\mathbb{Z}_p$. Destarte, cumpre $(A10)$ portanto, é um corpo. ■

Vamos verificar que o produto de duas classes em $\mathbb{Z}_p$ também possui inverso em $\mathbb{Z}_p$.

Proposição 2.39. *Dados $\overline{a}$, $\overline{b} \in \mathbb{Z}_p$, se $\overline{a}$ e $\overline{b}$ possuem inverso em $\mathbb{Z}_p$, então, $\overline{a} \cdot \overline{b}$ também possui inverso em $\mathbb{Z}_p$.*

Demonstração. Dado $\overline{a}$, $\overline{b} \in \mathbb{Z}$, existem $\overline{\alpha}$, $\overline{\beta}$ tais que $\overline{\alpha} \cdot \overline{a} = 1$ e $\overline{\beta} \cdot \overline{b} = 1$. Assim, $(\overline{a} \cdot \overline{b}) \cdot (\overline{\alpha} \cdot \overline{\beta}) = \overline{(a \cdot b)} \cdot \overline{(\alpha \cdot \beta)} = \overline{(a \cdot b) \cdot (\alpha \cdot \beta)} = \overline{(a \cdot \alpha) \cdot (b \cdot \beta)} = \overline{(a \cdot \alpha)} \cdot \overline{(b \cdot \beta)} = (\overline{a} \cdot \overline{\alpha}) \cdot (\overline{b} \cdot \overline{\beta}) = \overline{1} \cdot \overline{1} = \overline{1}$. ■

Podemos usar o que aprendemos para resolver congruências lineares em $\mathbb{Z}_n$. Uma congruência linear é uma equação do tipo $a \cdot x \equiv b \pmod{n}$, a, $b \in \mathbb{Z}$. Para resolvê-la, precisamos multiplicar a equação pelo inverso de a, para deixar x livre. Para isso, é preciso que $\text{mdc}(n, a) = 1$. Então, existe $\alpha \in \mathbb{Z}$ tal que $\alpha \cdot a \equiv 1 \pmod{n}$. Logo, $\alpha \cdot a \cdot x \equiv \alpha \cdot b \pmod{n}$ consequentemente, $1 \cdot x \equiv \alpha \cdot b \pmod{n}$ portanto, $x \equiv \alpha \cdot b \pmod{n}$.

Observação 2.40. Se $\text{mdc}(a, n) = 1$, então, à congruência linear de $a \cdot x = b \pmod{n}$ tem única solução em $\mathbb{Z}_n$.

3 MATRIZES

Antes de adentrarmos aos sistemas lineares em $\mathbb{R}$, vamos fazer uma pequena revisão de alguns conceitos básicos envolvendo matrizes e operações matriciais. Este capítulo baseia-se nas referências [3, 8, 10, 14, 19].

Dados dois números naturais e não nulos m e n, definimos como *matriz* m por n (escreve-se $m \times n$) toda tabela de números reais com m linhas e n colunas. O número de linhas por colunas de uma matriz é chamado *ordem* da matriz. Representamos uma matriz de m linhas e n colunas (de ordem $m \times n$) por:

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}.$$

Os números a_{ij} que compõem a matriz são chamados *elementos ou entradas* da matriz e são, em geral, escalares que pertencem a algum corpo $\mathbb{K}$. O índice i representa a linha em que o elemento se encontra e o índice j representa a coluna. É convencionado que as linhas sejam numeradas de cima para baixo (de 1 até m) e as colunas sejam numeradas da esquerda para a direita (de 1 até n).

Uma matriz A de ordem $m \times n$ também pode ser indicada por $A = [a_{ij}]_{m \times n}$ em que $i \in \{1, 2, \cdots, n\}$ e $j \in \{1, 2, \cdots, m\}$ ou simplesmente $A = [a_{ij}]$, se a ordem estiver subentendida.

Vejamos agora alguns exemplos de matrizes e suas respectivas ordens. Tome $\mathbb{K}$ um corpo, daí:

a) $A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \end{bmatrix}$ tem ordem 1×4;

b) $B = \begin{bmatrix} 0_{\mathbb{K}} & 0_{\mathbb{K}} \\ 0_{\mathbb{K}} & 0_{\mathbb{K}} \end{bmatrix}$ tem ordem 2×2;

c) $C = \begin{bmatrix} c_{11} & 0_{\mathbb{K}} & 0_{\mathbb{K}} \\ c_{21} & c_{22} & 0_{\mathbb{K}} \\ c_{31} & c_{32} & c_{33} \end{bmatrix}$ tem ordem 3×3;

d) $D = \begin{bmatrix} d_{11} \\ d_{21} \\ d_{31} \end{bmatrix}$ tem ordem 3×1;

e) $E = \begin{bmatrix} e_{11} & e_{12} & e_{13} & e_{14} \\ e_{21} & e_{22} & e_{23} & e_{24} \\ e_{31} & e_{32} & e_{33} & e_{34} \end{bmatrix}$ tem ordem 3×4;

f) $I = \begin{bmatrix} 1_{\mathbb{K}} & 0_{\mathbb{K}} & 0_{\mathbb{K}} \\ 0_{\mathbb{K}} & 1_{\mathbb{K}} & 0_{\mathbb{K}} \\ 0_{\mathbb{K}} & 0_{\mathbb{K}} & 1_{\mathbb{K}} \end{bmatrix}$ tem ordem 3×3.

Observação 3.1. Quando uma matriz tem ordem $n \times n$, diremos apenas que ela tem ordem n.

3.1 TIPOS DE MATRIZES

Algumas matrizes têm características e propriedades que as tornam importantes e que fazem com que sejam muito utilizadas, por isso recebem nomes especiais; vamos conhecer algumas delas.

Matriz linha

Definição 3.2. Uma matriz A é denominada *matriz linha* se possuir apenas uma linha, ou seja, se for de ordem $1 \times n$, em que $n \in \mathbb{N}^*$.

Observação 3.3. Matrizes do tipo linha, também são conhecidas como *vetor linha.*

Exemplo 3.4. Vejamos alguns exemplos:

a) $A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} & a_{15} \end{bmatrix}$;

b) $B = \begin{bmatrix} b_{11} & b_{12} \end{bmatrix}$;

c) $C = \begin{bmatrix} c_{11} & c_{12} & c_{13} & c_{14} \end{bmatrix}$.

Matriz coluna

Definição 3.5. Uma matriz A é denominada *matriz coluna* se possuir apenas uma coluna, ou seja, se for de ordem $m \times 1$, em que $m \in \mathbb{N}^*$.

Exemplo 3.6. Vejamos alguns exemplos:

a) $A = \begin{bmatrix} a_{11} \\ a_{21} \\ a_{31} \\ a_{41} \end{bmatrix}$;

b) $B = \begin{bmatrix} b_{11} \\ b_{12} \\ b_{13} \end{bmatrix}$;

c) $C = \begin{bmatrix} c_{11} \\ c_{12} \\ c_{13} \\ c_{14} \end{bmatrix}$.

Observação 3.7. Matrizes do tipo coluna também são conhecidas como *vetor coluna*. Como futuramente trabalharemos muito com vetores colunas, por convenção, vamos chamá-los apenas de vetores, omitindo a palavra coluna. Neste caso, podemos dizer também que tais vetores tem simplesmente ordem m.

Matriz nula

Definição 3.8. Uma matriz A é denominada *matriz nula* se todos os seus elementos são iguais a zero, ou seja, $a_{ij} = 0_{\mathbb{K}}$ para todo $i \in \{1, 2, \cdots, n\}$ e todo $j \in \{1, 2, \cdots, m\}$. Pode-se denotar uma matriz nula de ordem $m \times n$ por $0_{m \times n}$.

Exemplo 3.9. Vejamos alguns exemplos:

a) $A = \begin{bmatrix} 0_{\mathbb{K}} & 0_{\mathbb{K}} & 0_{\mathbb{K}} \end{bmatrix}$;

b) $B = \begin{bmatrix} 0_{\mathbb{K}} & 0_{\mathbb{K}} \\ 0_{\mathbb{K}} & 0_{\mathbb{K}} \end{bmatrix}$;

c) $C = \begin{bmatrix} 0_{\mathbb{K}} & 0_{\mathbb{K}} \\ 0_{\mathbb{K}} & 0_{\mathbb{K}} \\ 0_{\mathbb{K}} & 0_{\mathbb{K}} \end{bmatrix}$.

Matiz quadrada

Definição 3.10. Uma matriz A é denominada *matriz quadrada* se seu número de linhas igual ao número de colunas, ou seja, $m = n$. Neste caso, quando a matriz é de ordem $n \times n$, dizemos apenas que A é de ordem n.

Exemplo 3.11. Vejamos alguns exemplos:

a) $A = \begin{bmatrix} a_{11} \end{bmatrix}$;

b) $B = \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix}$;

c) $C = \begin{bmatrix} 1_{\mathbb{K}} & 0_{\mathbb{K}} & 0_{\mathbb{K}} \\ 0_{\mathbb{K}} & 1_{\mathbb{K}} & 0_{\mathbb{K}} \\ 0_{\mathbb{K}} & 0_{\mathbb{K}} & 1_{\mathbb{K}} \end{bmatrix}$.

A matriz quadrada possui alguns elementos importantes, dentre os quais estão a diagonal principal, vamos então defini-los.

Definição 3.12. A *diagonal principal* de uma matriz quadrada é o conjunto dos elementos da matriz em que o índice i é igual ao índice j, ou seja a_{11}, a_{22}, $\ldots$, a_{nn}.

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

Matriz triangular superior

Definição 3.13. Uma matriz A é denominada *triangular superior* se é quadrada, e os elementos abaixo da diagonal principal são todos nulos, ou seja, $a_{ij} = 0_{\mathbb{K}}$ se $i > j$.

Exemplo 3.14. Vejamos um exemplo:

$$U = \begin{bmatrix} u_{11} & u_{12} & u_{13} \\ 0_{\mathbb{K}} & u_{22} & u_{23} \\ 0_{\mathbb{K}} & 0_{\mathbb{K}} & u_{33} \end{bmatrix}.$$

Matriz triangular inferior

Definição 3.15. Uma matriz A é denominada *triangular inferior* se é quadrada, e os elementos acima da diagonal principal são todos nulos, ou seja, $a_{ij} = 0_{\mathbb{K}}$ se $i < j$.

Exemplo 3.16. Vejamos um exemplo:

$$L = \begin{bmatrix} \ell_{11} & 0_{\mathbb{K}} & 0_{\mathbb{K}} \\ \ell_{21} & \ell_{22} & 0_{\mathbb{K}} \\ \ell_{31} & \ell_{32} & \ell_{33} \end{bmatrix}.$$

Matriz diagonal

Definição 3.17. Uma matriz A é denominada *matriz diagonal* se for quadrada ($m = n$) e se $a_{ij} = 0_{\mathbb{K}}$ para todo $i \neq j$, ou seja, todos os elementos fora da diagonal principal são nulos, ou seja, se for simultaneamente triangular superior e inferior.

Exemplo 3.18. Vejamos alguns exemplos

a) $A = \begin{bmatrix} a_{11} & 0_{\mathbb{K}} & 0_{\mathbb{K}} \\ 0_{\mathbb{K}} & a_{22} & 0_{\mathbb{K}} \\ 0_{\mathbb{K}} & 0_{\mathbb{K}} & a_{33} \end{bmatrix}$; b) $B = \begin{bmatrix} 0_{\mathbb{K}} & 0_{\mathbb{K}} & 0_{\mathbb{K}} \\ 0_{\mathbb{K}} & 1_{\mathbb{K}} & 0_{\mathbb{K}} \\ 0_{\mathbb{K}} & 0_{\mathbb{K}} & 0_{\mathbb{K}} \end{bmatrix}$.

Matriz Identidade

Definição 3.19. Um matriz A é denominada *matriz identidade* ou *matriz unidade* se é quadrada, diagonal e todos os seus elementos em que $i = j$ (elemento da diagonal principal) são iguais a $1_{\mathbb{K}}$. Denotaremos a matriz identidade por I_n.

Exemplo 3.20. Vejamos alguns exemplos:

a) $I_3 = \begin{bmatrix} 1_{\mathbb{K}} & 0_{\mathbb{K}} & 0_{\mathbb{K}} \\ 0_{\mathbb{K}} & 1_{\mathbb{K}} & 0_{\mathbb{K}} \\ 0_{\mathbb{K}} & 0_{\mathbb{K}} & 1_{\mathbb{K}} \end{bmatrix}$; b) $I_2 = \begin{bmatrix} 1_{\mathbb{K}} & 0_{\mathbb{K}} \\ 0_{\mathbb{K}} & 1_{\mathbb{K}} \end{bmatrix}$.

3.2 OPERAÇÕES MATRICIAIS

Agora que já conhecemos um pouco sobre matrizes, garantiremos que as operações entre matrizes estão bem definidas.

Igualdade

Definição 3.21. Dadas duas matrizes $A = [a_{ij}]_{m\times n}$ e $B = [b_{ij}]_{m\times n}$ de mesma ordem, dizemos que são iguais (escreve-se $A = B$) se $a_{ij} = b_{ij}$ para todo $i \in \{1, 2, \cdots, n\}$ e todo $j \in \{1, 2, \cdots, m\}$.

Exemplo 3.22. Sejam A, B matrizes com coeficientes reais e $x, y \in \mathbb{R}$, as matrizes

$$A = \begin{bmatrix} 1 & x \\ -7 & 0 \end{bmatrix} \text{ e } B = \begin{bmatrix} 1 & 3 \\ y & 0 \end{bmatrix}$$

são iguais se $x = 3$ e $y = -7$.

Adição

Definição 3.23. Dadas duas matrizes $A = [a_{ij}]_{m\times n}$ e $B = [b_{ij}]_{m\times n}$ de mesma ordem, a *soma* de $A + B$ é uma matriz $C = [c_{ij}]_{m\times n}$, tal que $c_{ij} = a_{ij} + b_{ij}$ para todo $i \in \{1, 2, \cdots, n\}$ e todo $j \in \{1, 2, \cdots, m\}$.

Vejamos um exemplo:

Exemplo 3.24. Dadas $A = \begin{bmatrix} 2 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 2 \end{bmatrix}$ e $B = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$, com coeficientes reais, temos que:

$$\begin{aligned} A+B &= \begin{bmatrix} 2 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 2 \end{bmatrix} + \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} \\ &= \begin{bmatrix} 2+1 & -1+2 & 0+3 \\ -1+4 & 2+5 & -1+6 \\ 0+7 & -1+8 & 2+9 \end{bmatrix} \\ &= \begin{bmatrix} 3 & 1 & 3 \\ 3 & 7 & 5 \\ 7 & 7 & 11 \end{bmatrix}. \end{aligned}$$

Subtração

Definição 3.25. Dada uma matriz $A = [a_{ij}]_{m\times n}$, chama-se *inversa aditiva ou oposta de* A e denota-se por $-A = [-a_{ij}]_{m\times n}$ a matriz tal que $A+(-A) = 0$, em que 0 representa a matriz nula.

Definição 3.26. Dadas duas matrizes $A = [a_{ij}]_{m\times n}$ e $B = [b_{ij}]_{m\times n}$ de mesma ordem, a *subtração* de $A-B = A+(-B)$ é uma matriz $C = [c_{ij}]_{m\times n}$, tal que $c_{ij} = a_{ij} - b_{ij} = a_{ij} + (-b_{ij})$ para todo $i \in \{1, 2, \cdots, n\}$ e todo $j \in \{1, 2, \cdots, m\}$.

Segue, abaixo, um exemplo:

Exemplo 3.27. Dadas $A = \begin{bmatrix} 9 & -11 \\ 5 & 7 \\ 0 & -1 \end{bmatrix}$ e $B = \begin{bmatrix} 0 & 2 \\ 7 & 3 \\ 4 & -1 \end{bmatrix}$, com coeficientes

reais, temos que:

$$A - B = \begin{bmatrix} 9 & -11 \\ 5 & 7 \\ 0 & -1 \end{bmatrix} - \begin{bmatrix} 0 & 2 \\ 7 & 3 \\ 4 & -1 \end{bmatrix} = \begin{bmatrix} 9 & -11 \\ 5 & 7 \\ 0 & -1 \end{bmatrix} + \begin{bmatrix} 0 & -2 \\ -7 & -3 \\ -4 & 1 \end{bmatrix}$$
$$= \begin{bmatrix} 9+0 & -11-2 \\ 5-7 & 7-3 \\ 0-4 & -1+1 \end{bmatrix} = \begin{bmatrix} 9 & -13 \\ -2 & 4 \\ -4 & 0 \end{bmatrix}.$$

Proposição 3.28. *Dadas A, B, C matrizes de ordem $m \times n$, valem as seguintes propriedades:*

i) $A + (B + C) = (A + B) + C$ *(associatividade da adição);*

ii) $A + B = B + A$ *(comutatividade da adição);*

iii) $A + 0_{\mathbb{K}} = A$, *em que* $0_{\mathbb{K}}$ *denota a matriz nula* $m \times n$ *(elemento neutro da soma);*

iv) $A + (-A) = 0_{\mathbb{K}}$.

Demonstração. Se $A = [a_{ij}]$, $B = [b_{ij}]$ e $C = [c_{ij}]$, então

i) $A+(B+C) = [a_{ij}]+[b_{ij}+c_{ij}] = [a_{ij}+(b_{ij}+c_{ij})] = [(a_{ij}+b_{ij})+c_{ij}] = [a_{ij}+b_{ij}] + [c_{ij}] = (A+B)+C$.

ii) $A + B = [a_{ij}] + [b_{ij}] = [a_{ij} + b_{ij}] = [b_{ij} + a_{ij}] = [b_{ij}] + [a_{ij}] = B + A$.

iii) $A + 0_{\mathbb{K}} = [a_{ij}] + 0_{\mathbb{K}} = [a_{ij} + 0_{\mathbb{K}}] = [a_{ij}] = A$.

iv) $A + (-A) = [a_{ij}] + [-a_{ij}] = [a_{ij} - a_{ij}] = 0_{\mathbb{K}}$.

■

Produto por escalar

Definição 3.29. Seja $A = [a_{ij}]_{m\times n}$ uma matriz e $\alpha \in \mathbb{R}$ um escalar, chama-se *produto por escalar* a matriz B tal que $b_{ij} = \alpha \cdot a_{ij}$, para todo $i \in \{1, 2, \cdots, n\}$ e todo $j \in \{1, 2, \cdots, m\}$.

Exemplo 3.30. Sejam $\alpha = 7$ e $A = \begin{bmatrix} 3 & 5 & -6 \\ -1 & 2 & 8 \end{bmatrix}$. Temos que:

$$7A = 7 \cdot \begin{bmatrix} 3 & 5 & -6 \\ -1 & 2 & 8 \end{bmatrix} = \begin{bmatrix} 7\cdot 3 & 7\cdot 5 & 7\cdot -6 \\ 7\cdot -1 & 7\cdot 2 & 7\cdot 8 \end{bmatrix} = \begin{bmatrix} 21 & 35 & -42 \\ -7 & 14 & 56 \end{bmatrix}.$$

Proposição 3.31. *Dados $A = [a_{ij}]_{m\times n}$ e $B = [b_{ij}]_{m\times n}$ matrizes e $\alpha,\ \beta \in \mathbb{K}$ escalares, valem as seguintes propriedades:*

i) $\alpha \cdot (\beta \cdot A) = (\alpha \cdot \beta) \cdot A$;

ii) $\alpha \cdot (A + B) = \alpha \cdot A + \alpha \cdot B$;

iii) $(\alpha + \beta) \cdot A = \alpha \cdot A + \beta \cdot A$;

iv) $1_{\mathbb{K}} \cdot A = A$.

Demonstração. i) $\alpha \cdot (\beta \cdot A) = \alpha \cdot (\beta \cdot [a_{ij}]) = \alpha \cdot \beta \cdot [a_{ij}] = (\alpha \cdot \beta) \cdot [a_{ij}] = (\alpha \cdot \beta) \cdot A$.

ii) $\alpha \cdot (A + B) = \alpha \cdot [a_{ij} + b_{ij}] = [\alpha \cdot (a_{ij} + b_{ij})] = [\alpha \cdot a_{ij} + \alpha \cdot b_{ij}] = [\alpha \cdot a_{ij}] + [\alpha \cdot b_{ij}] = \alpha \cdot [a_{ij}] + \alpha \cdot [b_{ij}] = \alpha \cdot A + \alpha \cdot B$.

iii) $(\alpha + \beta) \cdot A = (\alpha + \beta) \cdot [a_{ij}] = \alpha \cdot [a_{ij}] + \beta \cdot [a_{ij}] = \alpha \cdot A + \beta \cdot A$.

iv) $1_{\mathbb{K}} \cdot A = 1_{\mathbb{K}} \cdot [a_{ij}] = [a_{ij}] = A$.

■

Produto de matrizes

Definição 3.32. Sejam $A = [a_{ij}]_{m\times n}$ e $B = [b_{ij}]_{n\times p}$ matrizes, chama-se *produto de A por B* ou simplesmente AB a matriz $C = [c_{ij}]_{m\times p}$, tal que

$c_{ij} = \sum_{k=1}^{n} a_{ik} \cdot b_{kj} = a_{i1} \cdot b_{1j} + \cdots + a_{in} \cdot b_{nj}$ para todo $i \in \{1, 2, 3 \cdots, m\}$ e todo $j \in \{1, 2, 3 \cdots, p\}$.

Observação 3.33. Note que para que o produto AB seja possível, o número de colunas de A deve ser igual ao número de linhas de B; devido a isso, a multiplicação de matrizes não é comutativa.

Exemplo 3.34. Sejam A e B matrizes com coeficientes reais temos que:

$$\begin{bmatrix} 1 & 2 & 1 \\ 3 & -1 & -3 \\ 2 & 3 & 1 \end{bmatrix} \cdot \begin{bmatrix} 3 \\ -1 \\ 4 \end{bmatrix} = \begin{bmatrix} 1 \cdot 3 + 2 \cdot (-1) + 1 \cdot 4 \\ 3 \cdot 3 + (-1) \cdot (-1) + (-3) \cdot 4 \\ 2 \cdot 3 + 3 \cdot (-1) + 1 \cdot 4 \end{bmatrix} = \begin{bmatrix} 5 \\ -2 \\ 7 \end{bmatrix}.$$

Proposição 3.35. *Sejam A, B e C matrizes e $\alpha \in \mathbb{R}$ escalar. Desde que a ordem das matrizes torne as operações possíveis, temos que:*

i) $A \cdot (B + C) = A \cdot B + A \cdot C$ *(distributividade à esquerda);*

ii) $(A + B) \cdot C = A \cdot C + B \cdot C$ *(distributividade à direita);*

iii) $(A \cdot B) \cdot C = A \cdot (B \cdot C)$ *(associatividade);*

iv) $(\alpha \cdot A) \cdot B = A \cdot (\alpha \cdot B) = \alpha \cdot (A \cdot B)$*;*

v) $A \cdot I = I \cdot A = A$ *(identidade como elemento neutro da multiplicação).*

Demonstração. i) Sejam $A = [a_{ij}]_{m \times n}$, $B = [b_{ij}]_{n \times p}$ e $C = [c_{ij}]_{n \times p}$, temos que

$$\begin{aligned} A \cdot (B + C) &= \sum_{k=1}^{n} a_{ik} \cdot (b_{kj} + c_{kj}) \\ &= \sum_{k=1}^{n} (a_{ik} \cdot b_{kj} + a_{ik} \cdot c_{kj}) \\ &= \sum_{k=1}^{n} a_{ik} \cdot b_{kj} + \sum_{k=1}^{n} a_{ik} \cdot c_{kj} \\ &= A \cdot B + A \cdot C. \end{aligned}$$

ii) Sejam $A = [a_{ij}]_{m\times n}$, $B = [b_{ij}]_{m\times n}$ e $C = [c_{ij}]_{n\times p}$, temos que

$$\begin{aligned}(A+B)\cdot C &= \sum_{k=1}^{n}(a_{kj}+b_{kj})\cdot c_{ik}\\ &= \sum_{k=1}^{n}(a_{kj}\cdot c_{ik}+b_{kj}\cdot c_{ik})\\ &= \sum_{k=1}^{n}a_{kj}\cdot c_{ik}+\sum_{k=1}^{n}b_{kj}\cdot c_{ik}\\ &= A\cdot C+B\cdot C.\end{aligned}$$

iii) Sejam $A = [a_{ij}]_{m\times n}$, $B = [b_{ij}]_{n\times p}$ e $C = [c_{ij}]_{p\times q}$, temos que

$$\begin{aligned}(A\cdot B)\cdot C &= \sum_{k=1}^{p}(A\cdot B)_{ik}\cdot c_{kj}\\ &= \sum_{k=1}^{p}\cdot\left(\sum_{l=1}^{n}a_{il}\cdot b_{lk}\right)\cdot c_{kj},\\ &= \sum_{l=1}^{n}a_{il}\cdot\left(\sum_{k=1}^{p}b_{lk}\cdot c_{kj}\right)\\ &= \sum_{l=1}^{n}a_{il}\cdot(B\cdot C)_{lj}\\ &= A\cdot(B\cdot C).\end{aligned}$$

iv) Sejam $A = [a_{ij}]_{m\times n}$ e $B = [b_{ij}]_{n\times p}$ temos que

$$(\alpha\cdot A)\cdot B = \sum_{k=1}^{n}(\alpha\cdot a_{ik})\cdot b_{kj} = \alpha\cdot\sum_{k=1}^{n}a_{ij}\cdot b_{ij} = \alpha\cdot(A\cdot B)$$

$$A\cdot(\alpha\cdot B) = \sum_{k=1}^{n}a_{ik}\cdot(\alpha\cdot b_{kj}) = \alpha\cdot\sum_{k=1}^{n}a_{ij}\cdot b_{ij} = \alpha\cdot(A\cdot B).$$

v) Sejam $A = [a_{ij}]_{m\times n}$ e $I = [\delta_{ij}]_{n\times n}$, temos que:

$$\begin{aligned} A \cdot I &= \sum_{k=1}^{n} a_{ik} \cdot \delta_{kj} \\ &= a_{i1} \cdot \delta_{1j} + \cdots + a_{ij} \cdot \delta_{jj} + \cdots + a_{in} \cdot \delta_{nj} \\ &= a_{i1} \cdot 0_{\mathbb{K}} \cdots + a_{ij} \cdot 1_{\mathbb{K}} + \cdots + a_{in} \cdot 0_{\mathbb{K}} \\ &= [a_{ij}] \\ &= A. \end{aligned}$$

$$\begin{aligned} I \cdot A &= \sum_{k=1}^{n} \delta_{ik} \cdot a_{kj} \\ &= \delta_{i1} \cdot a_{1j} + \cdots + \delta_{ii} \cdot a_{ij} + \cdots + \delta_{in} \cdot a_{nj} \\ &= 0_{\mathbb{K}} \cdot a_{1j} + \cdots + 1_{\mathbb{K}} \cdot a_{ij} + \cdots + 0_{\mathbb{K}} \cdot a_{nj} \\ &= [a_{ij}] \\ &= A. \end{aligned}$$

■

Observação 3.36. Dados A e B matrizes e α um escalar, faremos um abuso de notação e denotaremos os produtos $A \cdot B$ e $\alpha \cdot A$ por AB e αA, respectivamente.

Proposição 3.37. *Sejam A e B duas matrizes triangulares:*

i) Se A e B são triangulares superiores, então o produto AB é triangular superior.

ii) Se A e B são triangulares inferiores, então o produto AB é triangular inferior.

Demonstração. i) Sejam A e B matrizes triangulares superiores, tome a i-ésima linha de A como $a_{i\bullet}^T = \begin{bmatrix} 0 & \dots & 0 & a_{ii} & \dots & a_{in} \end{bmatrix}$ e a j-ésima

coluna de B como $b_j = \begin{bmatrix} b_{1j} \\ \vdots \\ b_{jj} \\ 0 \\ \vdots \\ 0 \end{bmatrix}$. Veja que o produto $a_{i\bullet}^T \cdot b_j$ é zero para todo $i > j$, logo AB é triangular superior.

ii) Sejam A e B matrizes triangulares inferiores, tome a i-ésima linha de A como $a_{i\bullet}^T = \begin{bmatrix} a_{i1} & \dots & a_{ii}0 & \dots & 0 \end{bmatrix}$ e a j-ésima coluna de B como $b_j = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ b_{jj} \\ \vdots \\ b_{nj} \end{bmatrix}$. Veja que o produto $a_{i\bullet}^T \cdot b_j$ é zero para todo $i < j$, logo AB é triangular inferior.

■

Matriz transposta

Definição 3.38. A transposta de $A = [a_{ij}]_{m\times n}$, denotada por $A^T = [a_{ji}]_{n\times m}$, é a matriz cuja entrada i, j é a_{ji}, ou seja, as linhas são trocadas por colunas e vice-versa.

Exemplo 3.39. Seja $A = \begin{bmatrix} 5 & 3 \\ 2 & 1 \end{bmatrix}$. Então,

$$A^T = \begin{bmatrix} 5 & 2 \\ 3 & 1 \end{bmatrix}.$$

Proposição 3.40. *Sejam $A = [a_{ij}]$ e $B = [b_{ij}]$ matrizes e $\alpha \in \mathbb{R}$. Desde que a ordem das matrizes sejam compatíveis, valem as seguintes propriedades da transposta:*

i) $(A^T)^T = A$;

ii) $(A+B)^T = A^T + B^T$;

iii) $(\alpha A)^T = \alpha A^T$;

iv) $(AB)^T = B^T A^T$.

Demonstração. i) Seja $A = [a_{ij}]_{m\times n}$, temos que, $(A^T)^T = [(a'_{ij})'] = [(a_{ji})'] = [a_{ij}] = A$.

ii) Sejam $A = [a_{ij}]_{m\times n}$ e $B = [b_{ij}]_{m\times n}$ então, $(A+B)^T = [(a_{ij}+b_{ij})'] = [a_{ji}+b_{ji}] = [a'_{ij}] + [b'_{ij}] = A^T + B^T$.

iii) $(\alpha A)^T = [\alpha \cdot a_{ij}]' = \alpha[a_{ji}] = \alpha[a_{ij}]' = \alpha A^T$.

iv) Sejam $A = [a_{ij}]_{m\times n}$, $B = [b_{ij}]_{n\times p}$ e $C = AB = [c_{ij}]_{m\times p}$, então $(AB)^T = [c'_{ij}] = [c_{ji}] = [\sum_{k=1}^{n} a_{jk} \cdot b_{ki}] = \sum_{k=1}^{n} [b'_{ik}] \cdot [a'_{kj}] = B^T A^T$. ■

Matrizes invertíveis

Definição 3.41. Seja A uma matriz quadrada de ordem n. Dizemos que A é uma *matriz invertível* se existir uma matriz B quadrada de ordem n, tal que $AB = BA = I_n$. Se A não é invertível, dizemos que A é uma *matriz singular*.

Teorema 3.42. *Se A é invertível, então é única a matriz B tal que $AB = BA = I$.*

Demonstração. Suponha que B, C sejam inversas de A. Logo, $AC = CA = I_n$ e $AB = BA = I_n$. Daí, $C = I_n C = (BA)C = B(AC) = BI_n = B$. Portanto, $B = C$ é a inversa de A e é única. ■

Definição 3.43. Dada uma matriz invertível A, chama-se *inversa de A* e denota-se por A^{-1} a matriz tal que $AA^{-1} = A^{-1}A = I_n$.

Proposição 3.44. *Sejam A e B matrizes quadradas de ordem n.*

i) Se A é não singular (invertível), então A^{-1} também é não singular e $(A^{-1})^{-1} = A$.

ii) Se A e B são não singulares (invertíveis), então AB também é não singular e $(AB)^{-1} = B^{-1}A^{-1}$.

iii) $(A^T)^{-1} = (A^{-1})^T$.

Demonstração. i) A é invertível, logo $A^{-1}A = AA^{-1} = I$. Então A^{-1} é invertível. Como a inversa é única $A = (A^{-1})^{-1}$.

ii) $(AB)\cdot(B^{-1}A^{-1}) = A(BB^{-1})A^{-1} = AIA^{-1} = AA^{-1} = I$ e $(B^{-1}A^{-1})\cdot(AB) = B^{-1}(A^{-1}A)B = B^{-1}IB = B^{-1}B = I$.

iii) Temos que $I = I^T = (AA^{-1})^T = (A^{-1})^T \cdot A^T$. Mas também temos $I = I^T = (A^{-1}A)^T = A^T \cdot (A^{-1})^T$. Então $(A^T)^{-1} = (A^{-1})^T$.

■

3.3 SUBMATRIZES E MATRIZES POR BLOCO

Definição 3.45. Uma *submatriz* ou *bloco* de uma matriz A de ordem n, é qualquer matriz obtida de A de ordem $r \times s$, suprimindo-se $n - r$ linhas ou $n - s$ colunas, ou é a própria matriz A.

Exemplo 3.46. Determinaremos todas as submatrizes da matriz

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \end{bmatrix}.$$

Tem-se submatrizes de ordens 2×3, 2×2, 2×1, 1×3 1×2 e 1×1. Sendo,

- $2 \times 3 : A$;
- $2 \times 2 : \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}, \begin{bmatrix} a_{11} & a_{13} \\ a_{21} & a_{23} \end{bmatrix}$ e $\begin{bmatrix} a_{12} & a_{13} \\ a_{22} & a_{23} \end{bmatrix}$;
- $2 \times 1 : \begin{bmatrix} a_{11} \\ a_{21} \end{bmatrix}, \begin{bmatrix} a_{12} \\ a_{22} \end{bmatrix}$ e $\begin{bmatrix} a_{13} \\ a_{23} \end{bmatrix}$;

- 1×3 : $\begin{bmatrix} a_{11} & a_{12} & a_{13} \end{bmatrix}$, $\begin{bmatrix} a_{21} & a_{22} & a_{23} \end{bmatrix}$;
- 1×2 : $\begin{bmatrix} a_{11} & a_{12} \end{bmatrix}$, $\begin{bmatrix} a_{12} & a_{13} \end{bmatrix}$, $\begin{bmatrix} a_{11} & a_{13} \end{bmatrix}$, $\begin{bmatrix} a_{21} & a_{22} \end{bmatrix}$, $\begin{bmatrix} a_{22} & a_{23} \end{bmatrix}$, $\begin{bmatrix} a_{21} & a_{23} \end{bmatrix}$;
- 1×1 : $\begin{bmatrix} a_{11} \end{bmatrix}$, $\begin{bmatrix} a_{12} \end{bmatrix}$, $\begin{bmatrix} a_{13} \end{bmatrix}$, $\begin{bmatrix} a_{21} \end{bmatrix}$, $\begin{bmatrix} a_{22} \end{bmatrix}$, $\begin{bmatrix} a_{23} \end{bmatrix}$.

Exemplo 3.47. Seja a matriz

$$A = \begin{bmatrix} a_{11} & a_{12} & 0 & 0 \\ a_{21} & a_{22} & 0 & 0 \\ 1 & 0 & a_{33} & a_{34} \\ 0 & 1 & a_{43} & a_{44} \end{bmatrix},$$

podemos particionar A nos seguintes blocos,

$$A = \left[\begin{array}{cc|cc} a_{11} & a_{12} & 0 & 0 \\ a_{21} & a_{22} & 0 & 0 \\ \hline 1 & 0 & a_{33} & a_{34} \\ 0 & 1 & a_{43} & a_{44} \end{array}\right],$$

obtendo as seguintes submatrizes

$$A_{11} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix},\ A_{12} = 0_2 = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix},$$

$$A_{21} = I_2 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \text{ e } A_{22} = \begin{bmatrix} a_{33} & a_{34} \\ a_{43} & a_{44} \end{bmatrix}.$$

Além disto podemos representar A da seguinte maneira:

$$A = \begin{bmatrix} A_{11} & 0_2 \\ I_2 & A_{22} \end{bmatrix}.$$

Definição 3.48. Seja $A = [a_{ij}]_{n\times n}$. O *menor principal* de ordem k de A é a matriz $A_k = [a_{ij}]_{k\times k}$ que é submatriz de dimensão k extraída de A, isto é

$$A_K = \left[\begin{array}{ccc|c|ccc} a_{11} & \cdots & a_{1k} & a_{1(k+1)} & \cdots & a_{1n} \\ \vdots & & \vdots & \vdots & & \vdots \\ a_{k1} & \cdots & a_{kk} & & & \\ \hline a_{(k+1)1} & \cdots & a_{(k+1)k} & a_{(k+1)(k+1)} & \cdots & a_{(k+1)n} \\ \hline \vdots & & & & \ddots & \vdots \\ a_{n1} & & & \cdots & & a_{nn} \end{array}\right].$$

Observação 3.49. Veja que $A_1 = [a_{11}]$ e $A_n = A$.

As submatrizes ou matrizes por blocos possuem grande importância na realização de cálculos com matrizes muito grandes, pois, algumas vezes, estamos interessados apenas em observar o comportamento de um determinado bloco da matriz. Utilizando este processo de separar a matriz em blocos, conseguimos analisar o comportamento da região desejada da matriz economizando cálculos. Além disso, é possível definir algoritmos de fatoração pensando nos blocos da matriz, o que mais uma vez pode economizar operações, a depender da estrutura de esparsidade da matriz.

4 SISTEMAS DE EQUAÇÕES LINEARES

Para atingir o objetivo deste trabalho que é analisar métodos de resolução de sistemas lineares no corpo $\mathbb{Z}_p$, precisamos entender o que é um sistema linear, e como encontrar sua solução. Para facilitar a compreensão, vamos começar analisando estes, no corpo dos reais $\mathbb{R}$. Neste capítulo, usaremos como referência as obras de [3, 7, 8, 10, 14, 16, 19].

4.1 COMPONENTES DO SISTEMA LINEAR

Um *sistema linear* é um conjunto de duas ou mais equações lineares, da forma $a_1x_1+a_2x_2+\cdots+a_nx_n = b$ em que os x_k's são incógnitas, e os a_k's e b_k's são números reais, para $k = 1, \ldots, n$. Além disto, $a_1^2+a_2^2+\cdots+a_n^2 \neq 0$, ou seja, os coeficientes são não simultaneamente nulos. Um sistema linear com m equações e n incógnitas é dado por

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n = b_1 \\ a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n = b_2 \\ \vdots \\ a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mn}x_n = b_m \end{cases}$$

Observação 4.1. Note que o sistema pode ser representado facilmente por $\sum_{j=1}^{n} a_{ij}x_j = b_i$ para $i = 1, 2, \ldots, m$.

Observação 4.2. Em algumas situações em que há muitos cálculos, usaremos a notação $a_{i\bullet}^T$ para nos referirmos a i-ésima linha da matriz A e usaremos a_j para nos referirmos a j-ésima coluna de A. Assim poderíamos representar a i-ésima linha do sistema por $a_{i\bullet}^T \cdot x$ que é equivalente a $a_{i1}x_1 + a_{i2}x_2 + \ldots + a_{in}x_n$.

Exemplo 4.3. Vejamos alguns exemplos de sistemas lineares no corpo dos números reais.

a) $\begin{cases} x_1 + 2x_2 = 5 \\ 2x_1 + 3x_2 = 8 \end{cases}$ c) $\begin{cases} x_1 + x_2 = 5 \\ 2x_1 \quad\quad = 4 \end{cases}$

b) $\begin{cases} 3x_1 + 2x_2 - x_3 = -2 \\ -3x_1 - x_2 + x_3 = 5 \\ 3x_1 + 2x_2 + x_3 = -2 \end{cases}$ d) $\begin{cases} 2x_1 + x_2 + x_3 = 1 \\ \quad - x_2 - 2x_3 = -4 \\ \quad\quad - 4x_3 = -4 \end{cases}$

Um sistema linear também pode ser visto na forma matricial na qual é expresso como $Ax = b$ em que A é a matriz dos coeficientes, com m linhas e n colunas, x é um vetor de n incógnitas e b é um vetor com m entradas reais.

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix}.$$

Além disso, podemos representar o sistema pela chamada matriz aumentada, (que denotaremos por $[A \mid b]$) como veremos abaixo

$$\left[\begin{array}{cccc|c} a_{11} & a_{12} & \cdots & a_{1n} & b_1 \\ a_{21} & a_{22} & \cdots & a_{2n} & b_2 \\ \vdots & & \ddots & \vdots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} & b_m \end{array}\right].$$

Se retomarmos os sistemas lineares do exemplo Exemplo 4.3 podemos expressá-los na forma $Ax = b$ como a seguir:

Exemplo 4.4.

a) $\begin{bmatrix} 1 & 2 \\ 2 & 3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 5 \\ 8 \end{bmatrix}$;

b) $\begin{bmatrix} 3 & 2 & -1 \\ -3 & -1 & 1 \\ 3 & 2 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} -2 \\ 5 \\ -2 \end{bmatrix}$;

c) $\begin{bmatrix} 1 & 1 \\ 2 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 5 \\ 4 \end{bmatrix}$;

d) $\begin{bmatrix} 2 & 1 & 1 \\ 0 & -1 & -2 \\ 0 & 0 & -4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 1 \\ -4 \\ -4 \end{bmatrix}$.

Tais sistemas tem as respectivas matrizes aumentadas:

a) $\left[\begin{array}{cc|c} 1 & 2 & 5 \\ 2 & 3 & 8 \end{array}\right]$;

b) $\left[\begin{array}{ccc|c} 3 & 2 & -1 & -2 \\ -3 & -1 & 1 & 5 \\ 3 & 2 & 1 & -2 \end{array}\right]$;

c) $\left[\begin{array}{cc|c} 1 & 1 & 5 \\ 2 & 0 & 4 \end{array}\right]$;

d) $\left[\begin{array}{ccc|c} 2 & 1 & 1 & 1 \\ 0 & -1 & -2 & -4 \\ 0 & 0 & -4 & -4 \end{array}\right]$.

4.2 SOLUÇÕES

Solucionar um sistema linear corresponde a encontrar o conjunto de incógnitas que satisfazem simultaneamente todas as equações. Geometricamente falando, quando resolvemos um sistema linear, estamos encontrando a interseção entre todos os hiperplanos dados pelas equações do sistema. Há três possibilidades para o conjunto das soluções de um sistema linear em $\mathbb{R}^n$, a saber:

i) *Solução única:* em que existe um único conjunto que satisfaz simultaneamente todas as equações lineares. Assim, há apenas um ponto de intersecção entre os hiperplanos gerados. Se isso ocorre dizemos que o sistema é possível e determinado.

ii) *Não há solução:* não existe um conjunto que satisfaça simultaneamente todas as equações lineares. Isso significa que não há nenhuma intersecção entre os hiperplanos gerados pelas equações. Nesse caso dizemos que o sistema é impossível ou incompatível.

iii) *Infinitas soluções:* há infinitos conjuntos que satisfazem as equações simultaneamente. Isso significa que os hiperplanos são coincidentes. Caso isso ocorra dizemos que o sistema linear é possível e indeterminado.

Observação 4.5. Quando um sistema possui solução, costumamos dizer que o sistema é consistente.

Observação 4.6. Se A for quadrada e inversível, então a solução única de $Ax = b$ é $x^* = A^{-1}b$.

4.3 SISTEMAS TRIANGULARES

Um caso interessante, de sistemas lineares, são os sistemas triangulares, em que a matriz dos coeficientes do sistema é uma matriz triangular. Este sistema é muito conveniente, pois, sua solução é encontrada facilmente. Os sistemas triangulares são classificados em triangular superior e triangular inferior.

Um sistema linear é triangular inferior se tiver a forma

$$\begin{cases} l_{11}x_1 & = b_1 \\ l_{21}x_1 + l_{22}x_2 & = b_2 \\ l_{31}x_1 + l_{32}x_2 + l_{33}x_3 & = b_3 \\ \vdots & \vdots \\ l_{n1}x_1 + l_{n2}x_2 + \ldots + l_{nn}x_n & = b_n \end{cases}$$

em que $l_{ij} = 0$ se $i < j$ e $l_{ii} \neq 0$, para $i = 1, 2, \ldots, n$.

Um sistema linear é triangular superior se tiver a forma

$$\begin{cases} u_{11}x_1 + u_{12}x_2 + \ldots + u_{1n}x_n & = b_1 \\ u_{22}x_2 + \ldots + u_{2n}x_n & = b_2 \\ \vdots & \vdots \\ u_{nn}x_n & = b_n \end{cases}$$

em que os coeficientes u_{ij} com $i > j$ são nulos e $u_{ii} \neq 0$, para $i = 1, 2, \ldots, n$.

Quando um sistema linear está na forma triangular superior ou inferior, dizemos que ele está na forma triangular estrita. Vamos definir formalmente.

Definição 4.7. Um sistema linear $Ax = b$ é dito na forma *triangular superior estrita* quando $a_{ij} = 0$ para todo $i > j$ e a_{ij} não nulo para todo $i = j$ com $i, j = 1, 2, \cdots, n$ e $A = [a_{ij}]_{n\times n}$.

Definição 4.8. Um sistema linear $Ax = b$ é dito na forma *triangular inferior estrita* quando $a_{ij} = 0$ para todo $i < j$ e a_{ij} não nulo para todo $i = j$ com $i, j = 1, 2, \cdots, n$ e $A = [a_{ij}]_{n\times n}$.

4.3.1 Substituição direta ou avançada

Um algoritmo eficiente para resolver sistemas triangulares estritos do tipo $Lx = b$ em que L é triangular inferior, é o de substituição direta. Neste algoritmo, dado um sistema linear do tipo

$$\begin{cases} l_{11}x_1 & = b_1 \\ l_{21}x_1 + l_{22}x_2 & = b_2 \\ l_{31}x_1 + l_{32}x_2 + l_{33}x_3 & = b_3 \\ \vdots & \vdots \\ l_{n1}x_1 + l_{n2}x_2 + \ldots + l_{nn}x_n & = b_n \end{cases}$$

podemos encontrar sua solução isolando a variável x_1 na primeira equação e substituindo o valor descoberto na segunda equação. Repetimos o processo sucessivas vezes, até obter o valor da variável x_n. Veja que, para descobrir o valor de x_1 efetuamos $x_1 = \frac{b_1}{l_{11}}$. Para descobrir o valor de x_2, substituímos o valor de x_1 encontrado na segunda equação obtendo assim $x_2 = \frac{b_2 - l_{21}x_1}{l_{22}}$. Para resolver x_3, substituímos os valores de x_1 e x_2 de modo a obter $x_3 = \frac{b_3 - (l_{31}x_1 + l_{32}x_2)}{l_{33}}$. Generalizando, temos que

$$x_k = \frac{\left(b_k - \sum_{j=1}^{k-1} l_{kj}x_j\right)}{l_{kk}}, \quad k = 2, \ldots, n. \tag{4.1}$$

Exemplo 4.9. Vamos aplicar o método da substituição direta no sistema

$$\begin{cases} 2x_1 & = 2 \\ -x_1 + 2x_2 & = 3 \\ 3x_1 + \ x_2 + x_3 & = 2 \end{cases}$$

Da equação $2x_1 = 2$ temos que $x_1 = 1$. Agora vamos usar a fórmula (4.1) e substituir o valor de x_1 na equação $-x_1+2x_2 = 3$. Assim, $-x_1+2x_2 = 3 \Rightarrow x_2 = \frac{3+x_1}{2} \Rightarrow x_2 = \frac{3+1}{2} \Rightarrow x_2 = 2$. Por fim, utilizamos a fórmula (4.1) e substituímos os valores de x_1 e x_2 encontrados na equação $3x_1+x_2+x_3 = 2$, então, $3x_1 + x_2 + x_3 = 2 \Rightarrow x_3 = 2 - 3x_1 - x_2 \Rightarrow x_3 = 2 - 3 - 2 \Rightarrow x_3 = -3$. Portanto, a solução do sistema é dada por $x = \begin{bmatrix} 1 & 2 & -3 \end{bmatrix}^T$.

4.3.2 Substituição reversa ou retro-substituição

Um algoritmo eficiente para resolver sistemas triangulares estritos do tipo $Ux = b$ me que U é triangular superior, é o de substituição reversa. Neste algoritmo, dado um sistema linear do tipo

$$\begin{cases} u_{11}x_1 + u_{12}x_2 + \ldots + u_{1n}x_n & = b_1 \\ u_{22}x_2 + \ldots + u_{2n}x_n & = b_2 \\ \vdots \quad \vdots & \\ u_{nn}x_n & = b_n \end{cases}$$

podemos encontrar sua solução isolando a variável x_n na última equação e substituindo o valor descoberto na penúltima equação. Repetimos o processo sucessivas vezes, até obter o valor da variável x_1. Veja que, para descobrir o valor de x_n efetuamos $x_n = \frac{b_n}{u_{nn}}$. Para descobrir o valor de x_{n-1}, substituímos o valor de x_n encontrado na segunda equação obtendo assim $x_{n-1} = \frac{b_{n-1} - u_{n-1,n}x_n}{u_{n-1,n-1}}$. Generalizando, temos que

$$x_k = \frac{\left(b_k - \sum_{j=k+1}^{n} u_{kj} \cdot x_j \right)}{u_{kk}}, k = n-1, \cdots 1. \tag{4.2}$$

Exemplo 4.10. Vamos aplicar o método de retro substituição no sistema

$$\begin{cases} 2x_1 + x_2 + \ x_3 = 1 \\ \quad - x_2 - 2x_3 = -4 \\ \quad\quad - 4x_3 = -4 \end{cases}$$

Da equação $-4x_3 = -4$ temos que $x_3 = 1$. Agora vamos usar a fórmula (4.2) substituir o valor encontrado para x_3 na equação $-x_2 - 2x_3 = -4$. Assim, $-x_2 - 2x_3 \Rightarrow x_2 = \frac{(-2)x_3}{-1} \Rightarrow x_2 = \frac{-2 \cdot 1}{-1} \Rightarrow x_2 = 2$. Por fim, utilizamos a fórmula (4.2) e substituindo os valores de x_2 e x_3 na equação $2x_1+x_2+x_3 = 1$ temos que $2x_1 + x_2 + x_3 = 1 \Rightarrow x_1 = \frac{1-x_2-x_3}{2} \Rightarrow x_1 = \frac{1-2-1}{2} \Rightarrow x_1 = -1$. Assim, a solução do sistema linear é dada por $\begin{bmatrix} -1 & 2 & 1 \end{bmatrix}^T$.

4.4 SISTEMAS EQUIVALENTES

O processo de eliminação de Gauss, que veremos sequência do trabalho, se baseia no fato de que podemos transformar (por operações bem definidas, chamadas elementares) um sistema linear em um outro sistema mais simples, porém equivalente, por sucessivamente eliminar variáveis e chegar, de alguma forma, em um sistema mais fácil de resolver.

A seguir, apresentamos formalmente o conceito de sistemas equivalente antes de procedermos com o delineamento do cerne da eliminação de Gauss: as operações elementares.

Definição 4.11. Dois sistemas lineares são ditos *equivalentes* se possuem o mesmo conjunto solução.

Vejamos um exemplo para ficar mais evidente.

Exemplo 4.12. Os sistemas a) e b) abaixo são equivalentes.

a)

$$\begin{cases} 3x_1 + 2x_2 - \ x_3 = -2 \\ \quad\quad x_2 \quad\quad = 3 \\ \quad\quad\quad\quad 2x_3 = 4 \end{cases}$$

b)

$$\begin{cases} 3x_1 + 2x_2 - x_3 = -2 \\ -3x_1 - x_2 + x_3 = 5 \\ 3x_1 + 2x_2 + x_3 = 2 \end{cases}$$

Com efeito, note que o item a) é facilmente resolvido por substituição reversa, veja bem, de $2x_3 = 4$ temos que $x_3 = 2$ e da segunda equação temos que $x_2 = 3$, assim $x_1 = \frac{-2-(2\cdot 3-1\cdot 2)}{3} = -2$ e portanto $x_1 = -2$. Assim, podemos expressar a solução do sistema linear pelo vetor $[-2, 3, 2]^T$. Já o b) parece ser muito mais complicado de se calcular, mas, na verdade, se trata de um sistema equivalente que possui mesma solução; vamos ver que isso é verdade mostrando que o vetor $[-2, 3, 2]^T$ satisfaz as equações. De fato, efetuando as respectivas substituições na última equação obtemos: $3 \cdot -2 + 2 \cdot 3 + 2 = 6 - 6 + 2 = 2$. Olhando para as equações restante, podemos encontrar: $-3 \cdot -2 - 3 + 2 = 6 - 2 + 3 = 5$ e, por fim, $3 \cdot -2 + 2 \cdot 3 - 2 = -6 + 6 - 2 = -2$ e isso significa que, ao realizarmos todas as substituições, as igualdades são verificadas. Dessa forma, temos dois sistemas equivalentes que possuem o mesmo vetor solução. Perceba que a solução de a) foi facilmente encontrada aplicando o procedimento de substituição reversa. Já para o item b) teríamos que calcular a inversa da matriz associada ao sistema. Salientamos ainda, que não realizamos esse procedimento aqui, apenas utilizamos a solução encontrada em a) para verificar que ambos os sistemas são equivalentes. Aliás, existem algumas operações que podemos utilizar sobre a matriz do sistema linear para transformar um sistema linear em um sistema equivalente.

4.4.1 Operações elementares

Seja $A = [a_{ij}]_{m\times n}$, denotaremos por L_i, $1 \leq i \leq m$ a i-ésima linha da matriz A. Definimos então, as seguintes operações elementares sobre as linhas de A:

i) Permutação das linhas L_i e L_j, denotada por $L_i \leftrightarrow L_j$;

ii) Multiplicar uma linha L_i por um escalar $\alpha \in \mathbb{R}$ (ou no caso geral $\alpha \in \mathbb{K}$ em que $\mathbb{K}$ é um corpo qualquer), denotada por $L_i \leftarrow \alpha L_i$;

iii) Substituição de uma linha L_i pela adição desta mesma linha a α vezes outra linha L_j, denotada por $L_i \leftarrow L_i + \alpha L_j$.

Vamos exemplificar efetuando as operações elementares sobre as linhas da matriz

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

Operação do tipo i):

$$\begin{bmatrix} 2 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 2 \end{bmatrix} \quad L_1 \leftrightarrow L_3 \quad \sim \begin{bmatrix} 0 & -1 & 2 \\ -1 & 2 & -1 \\ 2 & -1 & 0 \end{bmatrix};$$

Operação do tipo ii):

$$\begin{bmatrix} 2 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 2 \end{bmatrix} \quad \sim \begin{bmatrix} 2 & -1 & 0 \\ -1 & 2 & -1 \\ 2 & -2 & 2 \end{bmatrix};$$
$$L_3 \leftarrow L_3 + L_1$$

Operação do tipo iii):

$$\begin{bmatrix} 2 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 2 \end{bmatrix} \quad L_2 \leftarrow L_2 + 2L_3 \sim \begin{bmatrix} 2 & -1 & 0 \\ 3 & 0 & -1 \\ 0 & -1 & 2 \end{bmatrix}.$$

Observação 4.13. Perceba que todas as operações citadas acima, podem ser aplicadas na matriz aumentada de um sistema linear.

Veja que, toda transformação elementar é reversível, isto é, se $\mathcal{E}$ é uma operação elementar, então existe uma operação $\mathcal{E}'$ tal que $\mathcal{E}(\mathcal{E}'(A)) = A$ e $\mathcal{E}'(\mathcal{E}(A)) = A$.

De fato, se $\mathcal{E}$ é uma operação elementar do tipo troca de linhas, basta tomar $\mathcal{E} = \mathcal{E}'$. Se $\mathcal{E}$ é uma operação elementar do tipo multiplicação por um α escalar não nulo, basta que $\mathcal{E}'$ multiplique a mesma linha por $\frac{1}{\alpha}$. Se $\mathcal{E}$ é uma operação do tipo soma de uma linha i com β vezes a uma linha j, basta que $\mathcal{E}'$ subtraia da linha i, β vezes a linha j.

Definição 4.14. Seja A uma matriz e $\mathcal{E}$ uma operação elementar, se B é obtida pela aplicação de uma operação elementar em a $\mathcal{E}(A) = B$, então dizemos que B é *equivalente linha* a A.

Proposição 4.15. *Sejam $Ax = b$ um sistema linear, se pudermos obter $\hat{A}x = \hat{b}$ por sucessivas e finitas aplicações de operações elementares, ou seja, se $[A \mid b]$ é equivalente linha a $[\hat{A} \mid \hat{b}]$, então os sistemas são equivalentes.*

Demonstração. Dois sistemas lineares são equivalentes se possuem o mesmo conjunto solução, vamos então mostrar que as operações elementares não alteram o conjunto solução do sistema linear. Seja $\overline{x} = [x_1, x_2, \ldots, x_n]^T$ a solução de $Ax = b$.

I) Se aplicarmos uma operação do tipo I (troca de linhas) no sistema $Ax = b$, o conjunto solução permanece o mesmo, a menos de uma permutação de elementos, visto que as equações lineares não foram alteradas.

II) Se aplicarmos uma operação do tipo II (multiplicação de uma linha por escalar), sem perda de generalidade, na i-ésima linha do sistema de equações, $a_{i1} \cdot x_1 + a_{i2} \cdot x_2 + \ldots + a_{in} \cdot x_n = b_i$, teremos $\alpha \cdot a_{i1} \cdot x_1 + \alpha \cdot a_{i2} \cdot x_2 + \ldots + \alpha \cdot a_{in} \cdot x_n = \alpha \cdot b_i$ para algum $\alpha \in \mathbb{R}$, utilizando a associatividade, obtemos $(\alpha \cdot a_{i1}) \cdot x_1 + (\alpha \cdot a_{i2}) \cdot x_2 + \ldots + (\alpha \cdot a_{in}) \cdot x_n = \alpha \cdot b_i$, logo o conjunto solução do sistema linear permanece inalterado, portanto, os sistemas são equivalentes.

III) Se aplicarmos uma operação do tipo III (substituição de uma linha pela adição desta mesma linha com um múltiplo de outra) suponha, sem perda de generalidade, que estamos somando a i-ésima linha

$a_{i1} \cdot x_1 + a_{i2} \cdot x_2 + \ldots + a_{in} \cdot x_n = b_i$ com um múltiplo da j-ésima linha $\alpha \cdot a_{j1} \cdot x_1 + \alpha \cdot a_{j2} \cdot x_2 + \ldots + \alpha \cdot a_{jn} \cdot x_n = b_j$ em que $\alpha \in \mathbb{R}$ teremos $(a_{i1} \cdot x_1 + a_{i2} \cdot x_2 + \ldots + a_{in} \cdot x_n) + (\alpha \cdot a_{j1} \cdot x_1 + \alpha \cdot a_{j2} \cdot x_2 + \ldots + \alpha \cdot a_{jn} \cdot x_n) = b_i + (\alpha \cdot b_j)$ ao efetuarmos as somas e colocarmos os x's em evidência obtemos $x_1 \cdot (a_{i1} + \alpha \cdot a_{j1}) + x_2 \cdot (a_{i2} + \alpha \cdot a_{j2}) + \ldots + x_n \cdot (a_{in} + \alpha \cdot a_{jn}) = b_i + (\alpha \cdot b_j)$. Logo, o conjunto solução permanece inalterado portanto, os sistemas são equivalentes.

■

Vejamos um exemplo de aplicação da Proposição 4.15

Exemplo 4.16. Os sistemas

$$\begin{cases} 1x_1 + 4x_2 + 3x_3 = 1 \\ 2x_1 + 5x_2 + 4x_3 = 4 \\ 1x_1 - 3x_2 - 2x_3 = 5 \end{cases} \quad \text{e} \quad \begin{cases} x_1 \qquad\quad = 3 \\ \quad x_2 \quad = -2 \\ \qquad\quad x_3 = 2 \end{cases}$$

representados respectivamente pelas matrizes aumentadas

$$A = \left[\begin{array}{ccc|c} 1 & 4 & 3 & 1 \\ 2 & 5 & 4 & 4 \\ 1 & -3 & -2 & 5 \end{array}\right] \text{ e } B = \left[\begin{array}{ccc|c} 1 & 0 & 0 & 3 \\ 0 & 1 & 0 & -2 \\ 0 & 0 & 1 & 2 \end{array}\right]$$

são equivalentes, visto que

$$
\begin{array}{rl}
& \left[\begin{array}{ccc|c} 1 & 4 & 3 & 1 \\ 2 & 5 & 4 & 4 \\ 1 & -3 & -2 & 5 \end{array}\right] \begin{array}{l} \\ L_2 \leftarrow L_2 + (-2)\cdot L_1 \\ L_3 \leftarrow L_3 + (-1)\cdot L_1 \end{array} \\
\sim & \left[\begin{array}{ccc|c} 1 & 4 & 3 & 1 \\ 0 & -3 & -2 & 2 \\ 0 & -7 & -5 & 4 \end{array}\right] \begin{array}{l} \\ L_2 \leftarrow (-\frac{1}{3})\cdot L_2 \\ \\ \end{array} \\
\sim & \left[\begin{array}{ccc|c} 1 & 4 & 3 & 1 \\ 0 & 1 & \frac{2}{3} & -\frac{2}{3} \\ 0 & -7 & -5 & 4 \end{array}\right] \begin{array}{l} L_1 \leftarrow L_1 - 4\cdot L_2 \\ \\ L_3 \leftarrow L_3 + 7\cdot L_2 \end{array} \\
\sim & \left[\begin{array}{ccc|c} 1 & 0 & \frac{1}{3} & \frac{11}{3} \\ 0 & 1 & \frac{2}{3} & -\frac{2}{3} \\ 0 & 0 & -\frac{1}{3} & -\frac{2}{3} \end{array}\right] \begin{array}{l} L_1 \leftarrow L_1 + L_3 \\ \\ \\ \end{array} \\
\sim & \left[\begin{array}{ccc|c} 1 & 0 & 0 & 3 \\ 0 & 1 & \frac{2}{3} & -\frac{2}{3} \\ 0 & 0 & -\frac{1}{3} & -\frac{2}{3} \end{array}\right] \begin{array}{l} \\ \\ L_3 \leftarrow (-3)\cdot L_3 \end{array} \\
\sim & \left[\begin{array}{ccc|c} 1 & 0 & 0 & 3 \\ 0 & 1 & \frac{2}{3} & -\frac{2}{3} \\ 0 & 0 & 1 & 2 \end{array}\right] \begin{array}{l} \\ L_2 \leftarrow L_2 + (-\frac{2}{3})\cdot L_3 \\ \\ \end{array} \\
\sim & \left[\begin{array}{ccc|c} 1 & 0 & 0 & 3 \\ 0 & 1 & 0 & -2 \\ 0 & 0 & 1 & 2 \end{array}\right].
\end{array}
$$

Veja que ambos os sistemas possuem solução $x = \begin{bmatrix} 3 & -2 & 2 \end{bmatrix}^T$. Vamos verificar, no primeiro sistema de equações temos $1\cdot 3+4\cdot(-2)+3\cdot 2 = 1$, $2\cdot 3+5\cdot(-2)+4\cdot 2 = 4$ e $1\cdot 3-3\cdot(-2)-2\cdot 2 = 5$ e no segundo sistema é fácil ver que x é de fato solução, comprovando assim a equivalência.

4.4.2 Forma escalonada de uma matriz

Definição 4.17. Dada uma matriz $A = [a_{ij}]_{m\times n}$ dizemos que A está na forma *escalonada por linhas*, ou na forma *linha degrau* se:

i) Se a linha k não é composta apenas de zeros, o número de zeros que antecedem o primeiro elemento não-nulo da linha $k+1$ é maior que o da linha k;

ii) Linhas totalmente nulas ficam abaixo das linhas não-nulas.

Observação 4.18. Comumente chamaremos uma matriz na forma escalonada por linhas, de matriz na forma escalonada.

Definição 4.19. O primeiro elemento não nulo de uma linha de uma matriz escalonada recebe o nome de *pivô* da respectiva linha.

Exemplo 4.20. As matrizes abaixo estão na forma escalonada por linhas

$$A = \begin{bmatrix} \mathbf{5} & 3 & 0 & -1 \\ 0 & 0 & \mathbf{13} & 3 \\ 0 & 0 & 0 & \mathbf{-1} \\ 0 & 0 & 0 & 0 \end{bmatrix} \text{ e } B = \begin{bmatrix} \mathbf{1} & 3 & 7 \\ 0 & \mathbf{1} & 11 \\ 0 & 0 & \mathbf{3} \end{bmatrix}.$$

Além disto, os pivôs de A são $a_{11} = 5$, $a_{23} = 13$ e $a_{34} = -1$ e os pivôs de B são $a_{11} = 1$, $a_{22} = 1$ e $a_{33} = 3$.

Os pivôs também recebem o nome de *variáveis principais* enquanto as demais são chamadas variáveis livres.

Definição 4.21. O *grau de liberdade* de um sistema linear é o número de variáveis livres.

Definição 4.22. Seja A uma matriz na forma escalonada. O *rank* ou *posto* de A, denotado por posto(A) é o número de linhas não nulas de A.

Observação 4.23. Veja que dada a matriz dos coeficientes $A = [a_{ij}]_{m\times n}$ e dada $\hat{A} = [\hat{a}_{ij}]_{m\times(n+1)}$ a matriz aumentada de A, por estarmos adicionando uma coluna a matriz A, posto(A) $\leq$ posto($\hat{A}$).

Por exemplo, veja que dadas

$$A = \begin{bmatrix} 5 & 3 & 0 & -1 \\ 0 & 0 & 1 & 3 \\ 0 & 0 & 0 & -8 \\ 0 & 0 & 0 & 0 \end{bmatrix} \text{ e } \hat{A} = \left[\begin{array}{cccc|c} 5 & 3 & 0 & -1 & b_1 \\ 0 & 0 & 1 & 3 & b_2 \\ 0 & 0 & 0 & -8 & b_3 \\ 0 & 0 & 0 & 0 & b_4 \end{array}\right]$$

se $b_4 = 0$, $\text{posto}(A) = \text{posto}(\hat{A}) = 3$ e se $b_4 \neq 0$, $\text{posto}(A) < \text{posto}(\hat{A})$, pois $\text{posto}(\hat{A}) = 4$.

4.5 ELIMINAÇÃO GAUSSIANA

A eliminação gaussiana recorre às três operações elementares para transformar o sistema linear em um sistema equivalente. Vamos defini-la formalmente.

Definição 4.24. O processo de utilizar as operações elementares dos tipos I, II e III para transformar um sistema linear do tipo $Ax = b$ em outro do tipo $Ux = c$ em que U é uma matriz triangular superior é chamado *eliminação gaussiana.*

Vamos descrever o algoritmo da eliminação gaussiana. Inicialmente, escrevemos $A^{(0)} = A$ e $b^{(0)} = b$. A cada etapa $p = 0, 1, \ldots, n-1$, operações elementares são aplicadas no par $[A^{(p)} \mid b^{(p)}]$ para obter um novo par $[A^{(p+1)} \mid b^{(p+1)}]$ com zeros abaixo do elemento $a_{jj}^{(p)}$.

Na primeira etapa, zeramos todos os elementos abaixo do pivô $a_{11}^{(0)}$ subtraindo da j-ésima linha um múltiplo ℓ_{i1} da primeira linha.

$$\left[\begin{array}{cccc|c} a_{11}^{(0)} & a_{12}^{(0)} & \ldots & a_{1n}^{(0)} & b_1^{(0)} \\ a_{21}^{(0)} & a_{22}^{(0)} & \ldots & a_{2n}^{(0)} & b_2^{(0)} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ a_{n1}^{(0)} & a_{n2}^{(0)} & \ldots & a_{nn}^{(0)} & b_n^{(0)} \end{array}\right] \sim \left[\begin{array}{cccc|c} a_{11}^{(1)} & a_{12}^{(1)} & \ldots & a_{1n}^{(1)} & b_1^{(1)} \\ 0 & a_{22}^{(1)} & \ldots & a_{2n}^{(1)} & b_2^{(1)} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & a_{n2}^{(1)} & \ldots & a_{nn}^{(1)} & b_n^{(1)} \end{array}\right].$$

Para $i = 2, \ldots, n$, formalizamos este primeiro passo como

$$\ell_{i1} = \frac{a_{i1}^{(0)}}{a_{11}^{(0)}}, \quad b_i^{(1)} = b_i^{(0)} - \ell_{i1} b_1^{(0)} \quad \text{e} \quad a_{i1}^{(1)} = a_{i1}^{(0)} - \ell_{i1} a_{11}^{(0)}.$$

O próximo passo é zerar todos os elementos abaixo do pivô $a_{22}^{(1)}$, subtraindo da j-ésima um múltiplo ℓ_{i2} da segunda linha.

$$\left[\begin{array}{cccc|c} a_{11}^{(1)} & a_{12}^{(1)} & \dots & a_{1n}^{(1)} & b_1^{(1)} \\ a_{21}^{(1)} & a_{22}^{(1)} & \dots & a_{2n}^{(1)} & b_2^{(1)} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ a_{n1}^{(1)} & a_{n2}^{(1)} & \dots & a_{nn}^{(1)} & b_n^{(1)} \end{array}\right] \sim \left[\begin{array}{cccc|c} a_{11}^{(2)} & a_{12}^{(2)} & \dots & a_{1n}^{(2)} & b_1^{(2)} \\ 0 & a_{22}^{(2)} & \dots & a_{2n}^{(2)} & b_2^{(2)} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & a_{nn}^{(2)} & b_n^{(2)} \end{array}\right].$$

Para $i = 3, \dots, n$, formalizamos este passo como

$$\ell_{i2} = \frac{a_{i2}^{(1)}}{a_{22}^{(1)}}, \quad b_i^{(2)} = b_i^{(1)} - \ell_{i2} b_2^{(1)} \quad \text{e} \quad a_{i2}^{(2)} = a_{i2}^{(1)} - \ell_{i2} a_{22}^{(1)}.$$

De modo geral, para zeramos os elementos abaixo do pivô $a_{jj}^{(p)}$, em que $p = 1, 2, \dots, n$ utilizamos,

$$\ell_{ij} = \frac{a_{ij}^{(p-1)}}{a_{jj}^{(p-1)}}, \quad b_i^{(p)} = b_i^{(p-1)} - \ell_{ij} b_j^{(p-1)} \quad \text{e} \quad a_{ij}^{(p)} = a_{ij}^{(p-1)} - \ell_{ij} a_{jj}^{(p-1)}.$$

Exemplo 4.25. Vamos utilizar o algoritmo para resolver o sistema:

$$\begin{bmatrix} 6 & 2 & -1 \\ 2 & 4 & 1 \\ 3 & 2 & 8 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 7 \\ 7 \\ 13 \end{bmatrix}.$$

Primeiro, montamos a matriz aumentada do sistema:

$$\left[\begin{array}{ccc|c} 6 & 2 & -1 & 7 \\ 2 & 4 & 1 & 7 \\ 3 & 2 & 8 & 13 \end{array}\right].$$

1ª etapa. Temos que:

$$\ell_{21} = \frac{a_{21}^{(1)}}{a_{11}^{(1)}} = \frac{2}{6} = \frac{1}{3}, \quad \text{e} \quad \ell_{31} = \frac{a_{31}^{(1)}}{a_{11}^{(1)}} = \frac{3}{6} = \frac{1}{2}.$$

Assim:

$$a_{21}^{(2)} = a_{21}^{(1)} - a_{11}^{(1)}\ell_{21} \Rightarrow a_{21}^{(2)} = 2 - 6 \cdot \frac{1}{3} \Rightarrow a_{21}^{(2)} = 0;$$
$$a_{22}^{(2)} = a_{22}^{(1)} - a_{12}^{(1)}\ell_{21} \Rightarrow a_{22}^{(2)} = 4 - 2 \cdot \frac{1}{3} \Rightarrow a_{22}^{(2)} = \frac{10}{3};$$
$$a_{23}^{(2)} = a_{23}^{(1)} - a_{13}^{(1)}\ell_{21} \Rightarrow a_{23}^{(2)} = 1 - (-1) \cdot \frac{1}{3} \Rightarrow a_{23}^{(2)} = \frac{4}{3};$$
$$b_{2}^{(2)} = b_{2}^{(1)} - b_{1}^{(1)}\ell_{21} \Rightarrow b_{2}^{(2)} = 7 - 7 \cdot \frac{1}{3} \Rightarrow b_{2}^{(2)} = \frac{14}{3};$$
$$a_{31}^{(2)} = a_{31}^{(1)} - a_{11}^{(1)}\ell_{31} \Rightarrow a_{31}^{(2)} = 3 - 6 \cdot \frac{1}{2} \Rightarrow a_{31}^{(2)} = 0;$$
$$a_{32}^{(2)} = a_{32}^{(1)} - a_{12}^{(1)}\ell_{31} \Rightarrow a_{32}^{(2)} = 2 - 2 \cdot \frac{1}{2} \Rightarrow a_{32}^{(2)} = 1;$$
$$a_{33}^{(2)} = a_{33}^{(1)} - a_{13}^{(1)}\ell_{31} \Rightarrow a_{33}^{(2)} = 8 - (-1) \cdot \frac{1}{2} \Rightarrow a_{33}^{(2)} = \frac{17}{2};$$
$$b_{3}^{(2)} = b_{3}^{(1)} - b_{1}^{(1)}\ell_{31} \Rightarrow b_{3}^{(2)} = 13 - 7 \cdot \frac{1}{2} \Rightarrow b_{3}^{(2)} = \frac{19}{2}.$$

Assim, obtemos a matriz:

$$\left[\begin{array}{ccc|c} 6 & 2 & -1 & 7 \\ 0 & \frac{10}{3} & \frac{4}{3} & \frac{14}{3} \\ 0 & 1 & \frac{17}{2} & \frac{19}{2} \end{array}\right].$$

2ª etapa. Temos que:

$$\ell_{32} = \frac{a_{32}^{(2)}}{a_{22}^{(2)}} = \frac{1}{\frac{10}{3}} = \frac{3}{10}.$$

Assim:

$$a_{32}^{(3)} = a_{32}^{(2)} - a_{22}^{(2)}\ell_{32} \Rightarrow a_{32}^{(3)} = 1 - \frac{10}{3} \cdot \frac{3}{10} \Rightarrow a_{32}^{(3)} = 0;$$
$$a_{33}^{(3)} = a_{33}^{(2)} - a_{23}^{(2)}\ell_{32} \Rightarrow a_{33}^{(3)} = \frac{17}{2} - \frac{4}{3} \cdot \frac{3}{10} \Rightarrow a_{33}^{(3)} = \frac{81}{10};$$
$$b_{3}^{(3)} = b_{3}^{(1)} - a_{2}^{(2)}\ell_{32} \Rightarrow b_{3}^{(3)} = \frac{19}{2} - \frac{14}{3} \cdot \frac{3}{10} \Rightarrow b_{3}^{(3)} = \frac{81}{10}.$$

Portanto, obtemos:

$$\left[\begin{array}{ccc|c} 6 & 2 & -1 & 7 \\ 0 & \frac{10}{3} & \frac{4}{3} & \frac{14}{3} \\ 0 & 0 & \frac{81}{10} & \frac{81}{10} \end{array}\right]$$

ou seja:

$$\begin{bmatrix} 6 & 2 & -1 \\ 0 & \frac{10}{3} & \frac{4}{3} \\ 0 & 0 & \frac{81}{10} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 7 \\ \frac{14}{3} \\ \frac{81}{10} \end{bmatrix}.$$

Este exemplo com riqueza de detalhes, foi apenas para a boa compreensão do algoritmo, nos demais exemplos usaremos apenas a notação das operações elementares como no Exemplo 4.16.

Ao realizarmos o escalonamento, podemos analisar as possíveis soluções com mais facilidade, pois, após o escalonamento é possível constatar o posto da matriz de forma clara. Seja um sistema linear de m equações e n incógnitas dado por $Ax = b$, cuja matriz de coeficientes $A = [a_{ij}]_{m\times n}$ tem posto r e sua respectiva matriz aumentada $\hat{A} = [a_{ij}]_{m\times(n+1)}$ tem posto $\hat{r}$. Então,

i) Se $r = n = \hat{r}$ o sistema tem solução única;

ii) Se $r = \hat{r} < n$ o sistema possui variáveis livres e, portanto, possui infinitas soluções;

iii) Se $r < \hat{r}$ o sistema não possui solução.

Sistemas sobredeterminados

Um sistema é dito *sobredeteminado* se possui mais equações do que incógnitas. Sistemas sobredeterminados geralmente (mas não sempre) não possuem solução. Em um sistema deste tipo, a matriz dos coeficientes possui mais linhas do que colunas ($m > n$), como na figura abaixo.

Exemplo 4.26. Vamos ver alguns sistemas sobredeterminados

a) $$\begin{cases} x_1 + x_2 = 1 \\ x_1 - x_2 = 7 \\ -x_1 + 6x_2 = -6 \end{cases}$$

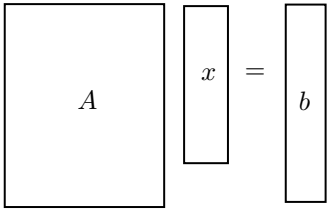


Figura 1 – Representação genérica de um sistema linear sobredeterminado.

b) $\begin{cases} x_1 + 2x_2 + x_3 = 1 \\ 2x_1 - x_2 + x_3 = 2 \\ 4x_1 + 3x_2 + 3x_3 = 4 \\ 2x_1 - x_2 + 3x_3 = 5 \end{cases}$

c) $\begin{cases} x_1 + 2x_2 + x_3 = 1 \\ 2x_1 - x_2 + x_3 = 2 \\ 4x_1 + 3x_2 + 3x_3 = 4 \\ 3x_1 + x_2 + 2x_3 = 3 \end{cases}$

Cujas matrizes aumentadas e respectivas soluções são

a)

$$\left[\begin{array}{cc|c} 1 & 1 & 1 \\ 1 & -1 & 7 \\ -1 & 6 & -6 \end{array}\right] \sim \left[\begin{array}{cc|c} 1 & 1 & 1 \\ 0 & 1 & -3 \\ 0 & 0 & -26 \end{array}\right].$$

Podemos ver pela última linha que o sistema não possui solução, pois não há como a última equação ser satisfeita.

b)

$$\left[\begin{array}{ccc|c} 1 & 2 & 1 & 1 \\ 2 & -1 & 1 & 2 \\ 4 & 3 & 3 & 4 \\ 2 & -1 & 3 & 5 \end{array}\right] \sim \left[\begin{array}{ccc|c} 1 & 2 & 1 & 1 \\ 0 & 1 & \frac{1}{5} & 0 \\ 0 & 0 & 1 & \frac{3}{2} \\ 0 & 0 & 0 & 0 \end{array}\right].$$

Usando a substituição reversa no sistema b), obtemos exatamente a solução $x = \begin{bmatrix} 0.1 & -0.3 & 1.5 \end{bmatrix}^T$. Veja que a solução é única, pois as linhas não nulas da matriz reduzida formam um sistema triangular estrito.

c)

$$\left[\begin{array}{ccc|c} 1 & 2 & 1 & 1 \\ 2 & -1 & 1 & 2 \\ 4 & 3 & 3 & 4 \\ 3 & 1 & 2 & 3 \end{array}\right] \sim \left[\begin{array}{ccc|c} 1 & 2 & 1 & 1 \\ 0 & 1 & \frac{1}{5} & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{array}\right].$$

Pondo x_1 e x_2 em função de x_3, obtemos $x_2 = -0.2x_3$ e $x_1 = 1 - 2x_2 - x_3 = 1 - 0.6x_3$. Portanto, o conjunto solução são todos os termos da forma $x = \begin{bmatrix} 1 - 0.6\alpha & -0.2\alpha & \alpha \end{bmatrix}^T$ em que $\alpha \in \mathbb{R}$. Este sistema é compatível e tem infinitas soluções.

Sistemas subdeterminados

Um sistema é dito *subdeterminado* se possui mais incógnitas do que equações. Em um sistema deste tipo, a matriz dos coeficientes possui mais colunas do que linhas ($m < n$), como na figura abaixo.

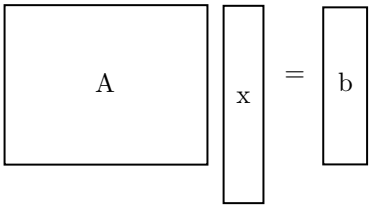


Figura 2 – Representação genérica de um seistema subdeterminado.

Sistemas subdeterminados geralmente possuem soluções infinitas. Isto se dá, pois, a matriz dos coeficientes na forma linha degrau terá $r \leq m$ linhas não nulas. Então, a matriz terá r pivôs e $n - r$ variáveis livres, em que $n - r \geq n - m > 0$.

Exemplo 4.27. Vejamos alguns sistemas subdeterminados

a) $\begin{cases} x_1 + 2x_2 + x_3 = 1 \\ 2x_1 + 4x_2 + 2x_3 = 3 \end{cases}$

b) $\begin{cases} x_1 + x_2 + x_3 + x_4 + x_5 = 2 \\ x_1 + x_2 + x_3 + 2x_4 + 2x_5 = 3 \\ x_1 + x_2 + x_3 + 2x_4 + 3x_5 = 2 \end{cases}$

Cujas matrizes aumentadas e respectivas formas escalonadas são

a)

$$\left[\begin{array}{ccc|c} 1 & 2 & 1 & 1 \\ 2 & 4 & 2 & 3 \end{array}\right] \sim \left[\begin{array}{ccc|c} 1 & 2 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{array}\right]$$

e

b)

$$\left[\begin{array}{ccccc|c} 1 & 1 & 1 & 1 & 1 & 2 \\ 1 & 1 & 1 & 2 & 2 & 3 \\ 1 & 1 & 1 & 2 & 3 & 2 \end{array}\right] \sim \left[\begin{array}{ccccc|c} 1 & 1 & 1 & 1 & 1 & 2 \\ 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & -1 \end{array}\right].$$

Como podemos ver facilmente, o sistema a) não possui solução. Já o sistema b), visto que temos duas variáveis livres, possui infinitas soluções da forma $x = \begin{bmatrix} 1-\alpha-\beta & \alpha & \beta & 2 & -1 \end{bmatrix}^T$, em que $\alpha, \beta \in \mathbb{R}$.

Vamos aqui utilizar um exemplo visto em [10] sobre o fluxo de tráfego de uma cidade, que mostra uma aplicação de sistemas lineares.

Exemplo 4.28. Na região central de uma cidade, dois conjuntos de ruas de mão única se intersetam como na Fig. 3. O volume horário de tráfego entrando e saindo dessa região durante a hora de pique é dado no diagrama. Determine o volume do tráfego entre cada uma das quatro interseções.

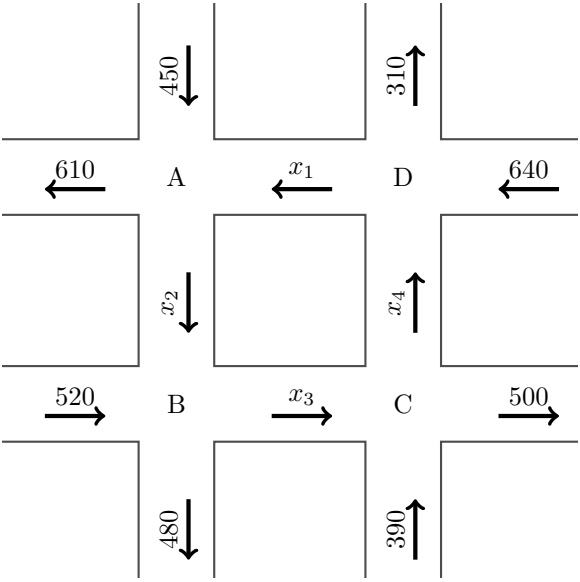


Figura 3 – Diagrama de fluxo de tráfego do exemplo.

Em cada interseção, o número de automóveis entrando deve ser o mesmo número saindo. Por exemplo, na interseção A, o número de automóveis entrando é $x_1 + 450$ e o número saindo é $x_2 + 610$. Então

$$x_1 + 450 = x_2 + 610 \quad \text{(interseção } A\text{)}.$$

Similarmente

$$x_2 + 520 = x_3 + 480 \quad \text{(interseção } B\text{)};$$
$$x_3 + 390 = x_4 + 600 \quad \text{(interseção } C\text{)};$$
$$x_4 + 640 = x_1 + 310 \quad \text{(interseção } D\text{)}.$$

A matriz aumentada para o sistema é

$$\left[\begin{array}{cccc|c} 1 & -1 & 0 & 0 & 160 \\ 0 & 1 & -1 & 0 & -40 \\ 0 & 0 & 1 & -1 & 210 \\ -1 & 0 & 0 & 1 & -330 \end{array}\right].$$

A forma escalonada da matriz é

$$\left[\begin{array}{cccc|c} 1 & 0 & 0 & -1 & 330 \\ 0 & 1 & 0 & -1 & 170 \\ 0 & 0 & 1 & -1 & 210 \\ 0 & 0 & 0 & 0 & 0 \end{array}\right].$$

O sistema possui uma variável livre, devido a isto, tem infinitas soluções possíveis. O diagrama de fluxo do tráfego não fornece informação suficiente para determinar x_1, x_2, x_3 e x_4 de forma unívoca. Se o volume do tráfego fosse conhecido entre qualquer par de interseções, o tráfego nas artérias restantes poderia ser facilmente calculado. Por exemplo, se o total de tráfego entre as interseções C e D é de 200 automóveis por hora em média, então $x_4 = 200$. Usando este valor podemos calcular x_1, x_2 e x_3.

$$x_1 = x_4 + 330 = 530;$$

$$x_2 = x_4 + 170 = 370;$$

$$x_3 = x_4 + 210 = 410.$$

Forma linha degrau reduzida

Definição 4.29. Dizemos que uma matriz está na forma *linha degrau reduzida* se:

i) está na forma linha degrau;

ii) O primeiro elemento não nulo em cada linha é o único elemento não nulo em sua coluna.

Exemplo 4.30. As matrizes a seguir estão na forma linha degrau reduzida:

a) $\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$;

b) $\begin{bmatrix} 6 & 0 & 0 & 3 \\ 0 & -3 & 0 & -2 \\ 0 & 0 & 1 & 1 \end{bmatrix}$;

c) $\begin{bmatrix} 0 & 5 & 7 & 0 \\ 0 & 0 & 0 & \frac{1}{2} \\ 0 & 0 & 0 & 0 \end{bmatrix}$; d) $\begin{bmatrix} -1 & 11 & 0 & -3 \\ 0 & 0 & 11 & 13 \\ 0 & 0 & 0 & 0 \end{bmatrix}$.

Observação 4.31. O processo de transformar a matriz aumentada do sistema linear na forma linha degrau reduzida recorrendo às operações elementares é conhecido como *redução de Gauss-Jordan*.

Exemplo 4.32. Vamos utilizar a redução de Gauss-Jordan no seguinte sistema

$$\begin{cases} -x_1 + x_2 - x_3 + 3x_4 = 0 \\ 3x_1 + x_2 - x_3 - x_4 = 0 \\ 2x_1 - x_2 - 2x_3 - x_4 = 0 \end{cases}$$

Veja que

$$
\begin{array}{rl}
& \left[\begin{array}{cccc|c} -1 & 1 & -1 & 3 & 0 \\ 3 & 1 & -1 & -1 & 0 \\ 2 & -1 & -2 & -1 & 0 \end{array}\right] \begin{array}{l} \\ L_2 \leftarrow L_2 + 3 \cdot L_1 \\ L_3 \leftarrow L_3 + 2 \cdot L_1 \end{array} \\
\sim & \left[\begin{array}{cccc|c} -1 & 1 & -1 & 3 & 0 \\ 0 & 4 & -4 & 8 & 0 \\ 0 & 1 & -4 & 5 & 0 \end{array}\right] \begin{array}{l} L_1 \leftarrow (-1) \cdot L_1 \\ L_2 \leftarrow \frac{1}{4} \cdot L_2 \\ \\ \end{array} \\
\sim & \left[\begin{array}{cccc|c} 1 & -1 & 1 & -3 & 0 \\ 0 & 1 & -1 & 2 & 0 \\ 0 & 1 & -4 & 5 & 0 \end{array}\right] \begin{array}{l} \\ \\ L_3 \leftarrow L_3 + (-1) \cdot L_2 \end{array} \\
\sim & \left[\begin{array}{cccc|c} 1 & -1 & 1 & -3 & 0 \\ 0 & 1 & -1 & 2 & 0 \\ 0 & 0 & -3 & 3 & 0 \end{array}\right] \begin{array}{l} L_1 \leftarrow L_1 + L_2 \\ \\ L_3 \leftarrow -\frac{1}{3} \cdot L_3 \end{array} \\
\sim & \left[\begin{array}{cccc|c} 1 & 0 & 0 & -1 & 0 \\ 0 & 1 & -1 & 2 & 0 \\ 0 & 0 & 1 & -1 & 0 \end{array}\right] \begin{array}{l} \\ L_2 \leftarrow L_2 + L_3 \\ \\ \end{array} \\
\sim & \left[\begin{array}{cccc|c} 1 & 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & -1 & 0 \end{array}\right]
\end{array}
$$

retornando ao sistema temos

$$
\begin{cases} x_1 \qquad\quad - x_4 = 0 \\ \quad x_2 \quad + x_4 = 0 \\ \qquad\quad x_3 - x_4 = 0 \end{cases}
$$

assumindo $x_4 = \alpha$ em que $\alpha \in \mathbb{R}$, temos a solução $x = \begin{bmatrix} \alpha & -\alpha & \alpha & \alpha \end{bmatrix}^T$.

Observação 4.33. O sistema que acabamos de resolver é da forma $Ax = 0$ ($b = 0$). Denominamos tal sistema de *sistema linear homogêneo*. Além disso, $Ax = 0$ tem sempre uma solução trivial $x = 0$.

Teorema 4.34. *Um sistema linear homogêneo $m \times n$ tem solução não trivial ($\hat{x} \neq 0$) se $m < n$ ou seja, se for subdeterminado.*

Demonstração. A forma linha degrau de um sistema linear homogêneo tem no máximo m linhas não nulas. Por consequência, tem no máximo m variáveis principais. Logo, com $m < n$, há $n - m > 0$ variáveis livres, as quais assumem valores arbitrários. Segue que temos infinitas soluções para tal sistema linear homogêneo. ■

4.6 SISTEMAS EQUIVALENTES (VERSÃO MATRICIAL)

Algumas operações como a multiplicação por matrizes não singulares, podem gerar sistemas equivalentes, desde que aplicadas em ambos os lados da igualdade do sistema. Vamos analisar melhor esta afirmação.

Teorema 4.35. *Dado um sistema linear do tipo*

$$Ax = b \tag{4.3}$$

em que A tem ordem $m \times n$, x tem ordem $n \times 1$, b tem ordem $m \times 1$. Dada a matriz C de ordem $m \times m$, não singular temos que o sistema $Ax = b$ é equivalente à

$$CAx = Cb. \tag{4.4}$$

Demonstração. Claro que qualquer solução de Eq. (4.3) é solução de Eq. (4.4). Seja $[x_1, x_2, \ldots, x_n]^T$ a solução de Eq. (4.3), tomando a i-ésima equação do sistema linear temos $a_{i1} \cdot x_1 + a_{i2} \cdot x_2 + \ldots + a_{in} \cdot x_n = b_i$ e se multiplicarmos pela matriz C à esquerda de A obtemos $(c_{i\bullet}^T \cdot a_1) \cdot x_1 + (c_{i\bullet}^T \cdot a_2) \cdot x_2 + \ldots (c_{i\bullet}^T \cdot a_n) \cdot x_n = c_{i\bullet}^T b_i$ veja que a solução x permanece inalterada.

Por outro lado, se $\hat{x}$ for solução de Eq. (4.4), então $CA\hat{x} = Cb \Rightarrow C^{-1}CA\hat{x} = C^{-1}Cb \Rightarrow A\hat{x} = b$ e, portanto $\hat{x}$ é solução de Eq. (4.3). Logo, Eq. (4.3) é equivalente a Eq. (4.4). ■

Para transformar o sistema linear $Ax = b$ em um sistema equivalente $Ux = y$, na forma triangular superior, podemos multiplicá-lo, sucessivas e

finitas vezes por matrizes não singulares $E_1, E_2, E_3, \ldots, E_k$. Em que $U = E_k \cdot \ldots \cdot E_3 \cdot E_2 \cdot E_1 \cdot A$ e $y = E_k \cdot \ldots \cdot E_3 \cdot E_2 \cdot E_1 \cdot b$. O sistema será equivalente ao original desde que $C = E_k \cdot \ldots \cdot E_3 \cdot E_2 \cdot E_1$ seja não singular. Mas, como C é um produto de matrizes não singulares, C é não singular, pelo Teorema 4.35.

4.6.1 Matrizes elementares sobre matrizes

Definição 4.36. As matrizes obtidas a partir da identidade mediante a aplicação de uma única operação elementar são camadas de *matrizes elementares*. Denotaremos $E_k = \mathcal{E}(I_n)$, em que $\mathcal{E}$ é uma operação elementar e I_n é a identidade de ordem n.

Exemplo 4.37. Suponha I_3, assim

Tipo I:

$$E_1 = \mathcal{E}_1(I_3) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

a matriz E_1 é obtida através da permutação das linhas 2 e 3 de I_3, portanto é do tipo I. Se multiplicarmos uma matriz A qualquer por E_1 obtemos

$$E_1 \cdot A = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{31} & a_{32} & a_{33} \\ a_{21} & a_{22} & a_{23} \end{bmatrix}.$$

Tipo II:

$$E_2 = \mathcal{E}_2(I_3) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \alpha & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

a matriz E_2 é obtida pela multiplicação da linha 2 de I_3 por um escalar $\alpha \neq 0$, portanto, é do tipo II. Se multiplicarmos uma matriz A qualquer por

E_2 obtemos

$$E_2 \cdot A = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ \alpha \cdot a_{21} & \alpha \cdot a_{22} & \alpha \cdot a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}.$$

Tipo III:

$$E_3 = \mathcal{E}_3(I_3) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ \alpha & 0 & 1 \end{bmatrix}$$

a matriz E_3 é obtida pela soma da linha 3 de I_3 a linha 1 por um escalar $\alpha \neq 0$, portanto, é do tipo III. Se multiplicarmos uma matriz A qualquer por E_3 obtemos

$$\begin{aligned} E_3 \cdot A &= \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ \alpha & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \\ &= \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} + \alpha \cdot a_{11} & a_{32} + \alpha \cdot a_{12} & a_{33} + \alpha \cdot a_{13} \end{bmatrix}. \end{aligned}$$

Corolário 4.38. *Toda matriz elementar é inversível e sua inversa também é uma matriz elementar.*

Demonstração. Seja E uma matriz do tipo I, então E é obtida permutando as linhas i e j da identidade I. Logo, podemos obter I permutando novamente as mesmas linhas. Portanto $E \cdot E = I$. Destarte, E é sua própria inversa quando é do tipo I.

Seja E uma matriz do tipo II, então E é obtida multiplicando a i-ésima linha da identidade I por um escalar α. Logo, podemos obter a identidade multiplicando E por uma matriz cuja i-ésima linha está sendo multiplicada por um escalar $\frac{1}{\alpha}$. Portanto, E cuja i-ésima linha é $\begin{bmatrix} 0 \cdots \frac{1}{\alpha} \cdots 0 \end{bmatrix}$ é a inversa de E. Destarte, matrizes do tipo II são não singulares.

Seja E uma matriz do tipo III, então E pode ser obtida somando-se m vezes uma linha i a uma linha j. Portanto, para reverter o processo, bata

subtrair m vezes a linha i da linha j. Sendo assim, a inversa de E é uma matriz cuja j-ésima linha é $-m$ vezes a i-ésima linha. Logo, uma matriz do tipo III possui inversa. Conclui-se então, que todas as matrizes elementares são não singulares. ■

Fazendo um apanhado das informações reunidas até aqui sobre matrizes inversíveis, podemos enunciar e demonstrar o teorema abaixo. Trata-se de uma sequência de afirmações equivalentes sobre matrizes inversíveis.

Teorema 4.39. *Seja $A = [a_{ij}]_{n\times n}$ com coeficientes reais, são equivalentes as seguintes afirmações*

i) A é não singular;

ii) $Ax = 0$ tem apenas a solução trivial $x = 0$;

iii) A é equivalente linha à I;

iv) $Ax = b$ tem solução única.

Demonstração. i) $\Rightarrow$ ii) Se A é não singular, existe A^{-1}. Logo, $Ax = 0 \Rightarrow A^{-1}Ax = Ix = 0$. Portanto $x = 0$.

ii) $\Rightarrow$ iii) Suponha que $x = 0$ seja a única solução de $Ax = 0$. Através de operações elementares podemos obter $[U \mid 0]$ em que U está na forma linha degrau. Se algum dos elementos da diagonal de U for zero, a última linha de U será nula, mas daí, $Ax = 0$ teria infinitas soluções. A forma linha degrau reduzida de U neste caso é I. Logo, U é uma matriz estritamente triangular com os elementos da diagonal iguais a 1 (também chamada matriz triangular unitária). Segue-se que I é a forma linha degrau reduzida de A portanto, A é equivalente linha de I.

iii) $\Rightarrow$ i) Se A é equivalente linha a I, existem $E_1, E_2, \ldots, E_k$ tal que $E_k \cdot \ldots \cdot E_2 \cdot E_1 \cdot I = A$. Como cada E_j é não singular, segue que $E_1^{-1} \cdot E_2^{-1} \cdot \ldots \cdot E_k^{-1} \cdot A = I$. Então, $A^{-1} = E_1^{-1} \cdot E_2^{-1} \cdot \ldots \cdot E_k^{-1}$ e, portanto A^{-1} é não-singular.

$i) \Rightarrow iv)$ Se A é não singular, então, existe A^{-1}. Logo, $A^{-1} \cdot Ax = A^{-1} \cdot b \Rightarrow Ix = A^{-1}b \Rightarrow x = A^{-1}b$. Suponha $\tilde{x}$ e x soluções de $A\widetilde{x} = b$ e $Ax = b$. Então $A\widetilde{x} = Ax \Rightarrow A^{-1} \cdot A\tilde{x} = A^{-1} \cdot Ax \Rightarrow \widetilde{x} = x$.

$iv) \Rightarrow (i)$ Se A é não singular e $\hat{\mathbf{x}}$ é qualquer solução de $Ax = b$, então $A\hat{x} = b$. Multiplicando-se ambos os lados dessa equação por A^{-1}, vemos que $\hat{x}$ deve ser igual a $A^{-1}b$. Inversamente, se $Ax = b$ tem uma única solução $\hat{x}$, então podemos dizer que A é não singular. Realmente, se A fosse singular, então a equação $Ax = 0$ teria uma solução $z \neq 0$. mas isto implicaria que $y = \hat{x} + z$ é uma segunda solução de $Ax = b$, já que $Ay = A(\hat{x} + z) = A\hat{x} + Az = b + 0 = b$. Portanto, se $Ax = b$ tem uma única solução, então A deve ser não singular.

■

4.7 FATORAÇÃO LU

Já vimos em seções anteriores, que por meio das operações elementares, podemos transformar um sistema do tipo $Ax = b$ em um sistema equivalente $\hat{A}x = \hat{b}$ através de sucessivas e finitas operações em suas matrizes aumentadas. Porém, se A for uma matriz muito grande e com muitos elementos, este processo se torna muito demorado e exigirá muitos cálculos. Veremos agora uma maneira melhor de transformar um sistema $Ax = b$ em um sistema equivalente mais fácil de se resolver usando uma fatoração na matriz A. Ao fatorarmos a matriz A em duas matrizes L e U, obtemos $A = LU$ em que L é triangular inferior unitária e U é triangular superior. Deste modo, dado um sistema linear $Ax = b$ em que A admite fatoração LU, obtemos então, $(LU)x = b$ e podemos resolver então os sistemas lineares $Ly = b$ e $Ux = y$. Logo, ao invés de resolvermos o sistema original, o que exigiria muitos cálculos, poderíamos apenas resolver $Ly = b$ e depois $Ux = y$.

Teorema 4.40. *Se uma matriz $A = [a_{ij}]_{n\times n}$ pode ser reduzida à forma triangular estrita somente por operações elementares do tipo III (sem troca*

de linhas) então $A = L \cdot U$ em que U é triangular superior e L é triangular inferior com a diagonal de uns.

Demonstração. Se A pode ser reduzida a forma triangular estrita, então $A \sim U$. Logo, existem $E_1, E_2, \ldots, E_k$ matrizes elementares do tipo III de modo que $E_k \cdot E_{k-1} \cdot \ldots \cdot E_1 \cdot A = U$. Manipulando a equação obtemos $A = E_1^{-1} \cdot E_2^{-1} \cdot \ldots \cdot E_k^{-1} \cdot U$. Definindo $L = E_1^{-1} \cdot E_2^{-1} \cdot \ldots \cdot E_k^{-1}$, temos que $A = L \cdot U$. É claro que L é triangular inferior, pois pelo Proposição 3.37, o produto de matrizes triangulares inferiores é triangular inferior. ■

Note que as hipóteses requeridas para o teorema acima são equivalentes a pedir que todas os menores principais (veja Definição 3.48) da matriz em questão são não singulares.

Teorema 4.41. *Seja A uma matriz inversível que admite fatoração LU, então, L e U são únicas.*

Demonstração. Suponha, por absurdo, que L e U não sejam únicas, ou seja, que existem L_1, L_2 e U_1, U_2 tais que $A = L_1 \cdot U_1 = L_2 \cdot U_2$, mas daí, $L_2^{-1} L_1 = U_2 U_1^{-1}$. Mas, o produto de duas matrizes triangulares superiores é triangular superior e o produto de duas matrizes triangulares inferiores é triangular inferior, conforme visto em Proposição 3.37, portanto $L_2^{-1} L_1$ é uma matriz triangular inferior unitária e $U_2 U_1^{-1}$ é triangular inferior. Daí, para que a igualdade seja mantida, $L_2^{-1} L_1 = U_2 U_1^{-1}$ deve ser obrigatoriamente uma matriz diagonal, cujos elementos da diagonal são 1's. Portanto $L_2^{-1} L_1 = U_2 U_1^{-1} = I$ e concluímos que $L_1 = L_2$ e $U_1 = U_2$. ■

A matriz U é obtida ao final do escalonamento da matriz A e a matriz L é obtida a partir da identidade, ao longo do escalonamento de A.

Exemplo 4.42. Seja

$$A = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & 2 \\ 1 & 2 & 3 \end{bmatrix}.$$

Observe que queremos escrever a matriz A como a multiplicação de duas matrizes triangulares e, para isso, vamos desenvolver o exemplo, simultaneamente construindo ambas conforme as operações elementares. Para isto, começamos identificamos o pivô da primeira coluna que é $a_{11} = 1$, então zeramos todos os elementos da primeira coluna abaixo do pivô. Para isso, precisamos encontrar ℓ_{21} e ℓ_{31} de modo que multiplicados pelo pivô da primeira coluna irão zerar respectivamente os elementos a_{21} e a_{31}. Temos que $\ell_{21} = \frac{a_{21}}{a_{11}} = \frac{1}{1} = 1$, de modo similar $\ell_{31} = \frac{a_{31}}{a_{11}} = \frac{1}{1} = 1$. Assim,

$$\begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & 2 \\ 1 & 2 & 3 \end{bmatrix} \quad \begin{matrix} L_2 \leftarrow L_2 - \ell_{21} \cdot L_1 \\ L_3 \leftarrow L_3 - \ell_{31} \cdot L_1 \end{matrix} \sim \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 1 & 2 \end{bmatrix}.$$

Como os elementos a_{21} e a_{31} agora são nulos, podemos guardar os multiplicadores ℓ_{21} e ℓ_{31} em seus lugares, visto que, a matriz L é formada pelo produto das matrizes elementares inversas, assim

$$\left[\begin{array}{ccc} 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ 1 & 1 & 2 \end{array}\right].$$

Na segunda coluna, temos agora que $a_{22} = 1$ é o pivô, assim, precisamos zerar todos os elementos abaixo de a_{22}.

Vamos então calcular ℓ_{32} tal que $\ell_{32} = \frac{a32}{a_{22}} = \frac{1}{1} = 1$. Então,

$$\left[\begin{array}{ccc} 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ 1 & 1 & 2 \end{array}\right] \quad L_3 \leftarrow L_3 - \ell_{32} \cdot L_2 \quad \sim \left[\begin{array}{ccc} 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ 1 & 1 & 1 \end{array}\right].$$

Assim,

$$U = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix}$$

e

$$L = \begin{bmatrix} 1 & 0 & 0 \\ \ell_{21} & 1 & 0 \\ \ell_{31} & \ell_{32} & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

. Veja que, na verdade, o que fizemos foi primeiro multiplicar A por

$$E_{21} = \begin{bmatrix} 1 & 0 & 0 \\ -1 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix},$$

depois por

$$E_{31} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -1 & 0 & 1 \end{bmatrix},$$

e, por fim, multiplicamos pela matriz

$$E_{32} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -1 & 1 \end{bmatrix},$$

obtendo assim

$$U = E_{32} \cdot E_{31} \cdot E_{21} \cdot A \text{ e } L = E_{21}^{-1} \cdot E_{31}^{-1} \cdot E_{31}^{-1}.$$

4.8 FATORAÇÃO PA=LU

Pivoteamento parcial

A fatoração LU depende dos multiplicadores $\ell_{ij} = \frac{a_{ij}}{a_{jj}}$, que só são obtidos se o pivô a_{jj} diferir de zero. Entretanto, o que acontece se o pivô for nulo? Mais ainda, e se a_{jj} for muito menor que a_{ij}, relativamente, como no caso abaixo,

$$A = \begin{bmatrix} 0.0001 & 1 \\ 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 10000 & 1 \end{bmatrix} \begin{bmatrix} 0.0001 & 1 \\ 0 & -9999 \end{bmatrix} = LU?$$

No primeiro caso, a operação é impossível. No segundo caso, pode-se incorrer em erro numérico. Para contornar estes problemas, usamos o que chamamos pivoteamento que nada mais é do que um processo de permutação de linhas de modo a obter um pivô não nulo ou de tamanho comparavelmente menor. Nesta estratégia, realizamos a permutação de linhas da matriz A quando necessário.

Definição 4.43. Uma matriz obtida a partir da identidade, permutando suas linhas (ou colunas) é chamada *matriz de permutação*. Denotaremos matrizes de permutação por P.

As matrizes de permutação são exatamente as matrizes elementares do tipo I da Definição 4.36 e do Exemplo 4.37.

Exemplo 4.44. Sejam

$$P = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix} \text{ e } A = \begin{bmatrix} 30 & 12 & 98 \\ 26 & 5 & 94 \\ 10 & 5 & 81 \end{bmatrix}$$

então

$$P \cdot A = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} 30 & 12 & 98 \\ 26 & 5 & 94 \\ 10 & 5 & 81 \end{bmatrix} = \begin{bmatrix} 26 & 5 & 94 \\ 10 & 5 & 81 \\ 30 & 12 & 98 \end{bmatrix}.$$

Se L e U são fatores de $\hat{A}$ e $\hat{A} = P \cdot A$, dizemos que estamos fazendo a fatoração $PA = LU$.

Exemplo 4.45. Vamos utilizar a fatoração $PA = LU$ no sistema abaixo.

$$\begin{cases} 3x_1 - 4x_2 + x_3 = 9 \\ x_1 + 2x_2 + 2x_3 = 3 \\ 4x_1 - 3x_3 = -2 \end{cases}$$

Veja que

$$A = \left[\begin{array}{ccc|c} 3 & -4 & 1 & 9 \\ 1 & 2 & 2 & 3 \\ 4 & 0 & -3 & -2 \end{array}\right].$$

O primeiro pivô é o elemento $a_{31} = 4$ pois é o elemento que possui maior valor em módulo na primeira coluna. Precisamos então multiplicar A pela matriz P que permuta as linhas 1 e 3

$$P_1 \cdot A = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix} \cdot \left[\begin{array}{ccc|c} 3 & -4 & 1 & 9 \\ 1 & 2 & 2 & 3 \\ 4 & 0 & -3 & -2 \end{array}\right] = \left[\begin{array}{ccc|c} 4 & 0 & -3 & -2 \\ 1 & 2 & 2 & 3 \\ 3 & -4 & 1 & 9 \end{array}\right].$$

Agora, vamos aplicar a fatoração LU na matriz permutada

$$\left[\begin{array}{ccc|c} 4 & 0 & -3 & -2 \\ 1 & 2 & 2 & 3 \\ 3 & -4 & 1 & 9 \end{array}\right] \begin{array}{l} L_2 \leftarrow L_2 - \frac{1}{4}\cdot L_1 \\ L_3 \leftarrow L_3 - \frac{3}{4}\cdot L_1 \end{array} \sim \left[\begin{array}{ccc|c} 4 & 0 & -3 & -2 \\ \frac{1}{4} & 2 & \frac{11}{4} & \frac{7}{2} \\ \frac{3}{4} & -4 & \frac{13}{4} & 12 \end{array}\right].$$

Agora na segunda coluna, o maior elemento em módulo é $a_{32} = -4$, portanto, vamos permutar as linhas 2 e 3

$$P_2 \cdot A = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix} \cdot \left[\begin{array}{ccc|c} 4 & 0 & -3 & -2 \\ \frac{1}{4} & 2 & \frac{11}{4} & \frac{7}{2} \\ \frac{3}{4} & -4 & \frac{13}{4} & \frac{21}{2} \end{array}\right] = \left[\begin{array}{ccc|c} 4 & 0 & -3 & -2 \\ \frac{3}{4} & -4 & \frac{13}{4} & \frac{21}{2} \\ \frac{1}{4} & 2 & \frac{11}{4} & \frac{7}{2} \end{array}\right].$$

Prosseguindo com a fatoração, temos

$$\left[\begin{array}{ccc|c} 4 & 0 & -3 & -2 \\ \frac{3}{4} & -4 & \frac{13}{4} & \frac{21}{2} \\ \frac{1}{4} & 2 & \frac{11}{4} & \frac{7}{2} \end{array}\right] L_3 \leftarrow L_3 + \frac{1}{3}\cdot L_1 \sim \left[\begin{array}{ccc|c} 4 & 0 & -3 & -2 \\ \frac{3}{4} & -4 & \frac{13}{4} & \frac{21}{2} \\ \frac{1}{4} & \frac{1}{2} & \frac{35}{8} & \frac{35}{4} \end{array}\right].$$

Assim, os fatores L e U são

$$U = \begin{bmatrix} 4 & 0 & -3 \\ 0 & -4 & \frac{13}{4} \\ 0 & 0 & \frac{35}{8} \end{bmatrix} \text{ e } L = \begin{bmatrix} 1 & 0 & 0 \\ \frac{3}{4} & 1 & 0 \\ \frac{1}{4} & \frac{1}{2} & \frac{35}{8} \end{bmatrix}$$

e estes são os fatores da matriz $\hat{A} = P \cdot A$ em que $P = P_2 \cdot P_1$, ou seja

$$\hat{A} = P \cdot A = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} 4 & 0 & -3 \\ 1 & 2 & 2 \\ 3 & -4 & 1 \end{bmatrix} = \begin{bmatrix} 4 & 0 & -3 \\ 3 & -4 & 1 \\ 1 & 2 & 2 \end{bmatrix}.$$

Resolvendo os sistemas triangulares, temos que

i) $Ly = Pb$ em que

$$Pb = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 9 \\ 3 \\ -2 \end{bmatrix} = \begin{bmatrix} -2 \\ 9 \\ 3 \end{bmatrix},$$

$$\begin{cases} 3y_1 & = -2 \\ \frac{3}{4}y_1 + y_2 & = 9 \\ \frac{1}{4}y_1 - \frac{1}{2}y_2 + y_3 & = 3 \end{cases} \Rightarrow \quad y = \begin{bmatrix} -2 \\ \frac{21}{2} \\ \frac{35}{4} \end{bmatrix},$$

ii) $Ux = y$

$$\begin{cases} 4x_1 - 3x_3 = -2 \\ -4x_2 + \frac{13}{4}x_3 = \frac{21}{2} \\ \frac{35}{8}x_3 = \frac{35}{4} \end{cases} \Rightarrow \quad x = \begin{bmatrix} 1 \\ -1 \\ 2 \end{bmatrix}.$$

Com efeito, qualquer matriz pode ser fatorada utilizando pivoteamento parcial. Abaixo, enunciamos este resultado, cuja demonstração pode ser encontrada em [14, p. 150-153] e em [19, Theorem 1.8.8].

Teorema 4.46. *Para qualquer matriz A de dimensão $n \times n$ existem matrizes, também de dimensão $n \times n$, L triangular inferior, tais que $l_{ii} = 1$, U triangular superior, e uma matriz de permutação P tal que*

$$PA = LU$$

4.9 CUSTO COMPUTACIONAL

Na cifra de Hill, é necessário calcular a inversa de uma matriz que é chave de encriptação para realizar a decodificação da mensagem. Entretanto, existem métodos que demandam muitos cálculos para realizar esse procedimento. Vamos ver então como seria a melhor forma de calcularmos essa matriz que é a chave de decodificação.

Em computação, quando desejamos o custo de um algoritmo, temos duas opções, ou comparamos a memória gasta para executá-lo, ou comparamos o tempo gasto.

Costuma-se medir o tempo gasto em *flops* que são operações de ponto flutuante, este nome se dá, pois, geralmente os computadores armazenam números reais como tipo ponto flutuante. Assim, quanto mais cálculos são necessários para se resolver um problema, maior sera o número de flops. É

claro que existem outros fatores que influenciam na contagem do tempo, como memória, taxa de transferência, etc. Porém, aqui, nos concentraremos apenas nos flops.

Classificamos a dificuldade de uma operação com o que chamamos de *ordem*, utilizando como base a quantidade de flops, neste caso, a quantidade de trabalho é igualada linearmente com a dimensão. Por exemplo, se em um cálculo são contabilizados n flops, dizemos que a operação é de ordem n e denotamos por $\mathcal{O}(n)$. Se forem contabilizados n^2 flops, dizemos que a operação é de ordem $\mathcal{O}(n^2)$.

Vamos nos basear na referência [6], para classificar algumas operações entre vetores e matrizes e contabilizar seus flops, para esclarecer melhor.

Vamos analisar o custo para realizar operações entre vetores pertencentes a um espaço vetorial de dimensão n, digamos, por exemplo, o $\mathbb{R}^n$. Veja abaixo.

a) Multiplicação de um escalar por um vetor de tamanho n,

$$\alpha \cdot v = \begin{bmatrix} \alpha \cdot v_1 & \alpha \cdot v_2 & \dots & \alpha \cdot v_n \end{bmatrix}^T.$$

São necessários n produtos, portanto tem ordem $\mathcal{O}(n)$.

b) Soma de dois vetores v e w de tamanho n,

$$v + w = \begin{bmatrix} v_1 + w_1 & v_2 + w_2 & \dots & v_n + w_n \end{bmatrix}^T.$$

São necessárias n somas, portante tem ordem $\mathcal{O}(n)$.

c) Soma de um múltiplo de um vetor com outro vetor

$$\alpha \cdot v + w = \begin{bmatrix} \alpha \cdot v_1 + w_1 & \alpha \cdot v_2 + w_2 & \dots & \alpha \cdot v_n + w_n \end{bmatrix}^T.$$

São necessários n produtos e n somas, portanto tem ordem $\mathcal{O}(n)$.

d) Produto escalar de dois vetores de tamanho n,

$$v^T w = v_1 \cdot w_1 + v_2 \cdot w_2 + \cdot + v_n \cdot w_n.$$

São necessários n produtos e $n-1$ somas, temos então $2n-1$ operações, portanto ordem $\mathcal{O}(n)$.

Operações de ordem $\mathcal{O}(n)$, como estas que acabamos de ver, são chamadas operações nível 1.

Vamos analisar agora, o custo de operações a multiplicação de uma matriz por um vetor $A \cdot x = b$ em que $A \in \mathbb{R}^{m \times n}$ e $x \in \mathbb{R}^n$.

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} a_{11} \cdot x_1 + a_{12} \cdot x_2 + \cdots + a_{1n} \cdot x_n \\ a_{21} \cdot x_1 + a_{22} \cdot x_2 + \cdots + a_{2n} \cdot x_n \\ \vdots \\ a_{m1} \cdot x_1 + a_{m1} \cdot x_2 + \cdots + a_{mn} \cdot x_n \end{bmatrix}.$$

Veja que para obtermos um elemento de b é necessário multiplicar uma linha de A pelo vetor coluna x, isto é equivalente à operação de produto escalar de dois vetores, portanto, possui $2n-1$ operações, que como já vimos anteriormente tem ordem $\mathcal{O}(n)$. Mas, veja que para obter b precisamos realizar este processo m vezes, teremos o custo de $m \cdot (2n-1) = (2n \cdot m - m)$ flops. Se supormos $m = n$ teremos o custo de $(2n^2 - n)$ flops, o que nos dá ordem $\mathcal{O}(n^2)$.

Calculemos o custo de multiplicação de duas matrizes. Sejam $A \in \mathbb{R}^{m \times p}$ e $B \in \mathbb{R}^{p \times n}$, queremos calcular o custo de $A \cdot B = C$.

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1p} \\ a_{21} & a_{22} & \cdots & a_{2p} \\ \vdots & & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mp} \end{bmatrix} \cdot \begin{bmatrix} b_{11} & b_{12} & \cdots & b_{1n} \\ b_{21} & b_{22} & \cdots & b_{2n} \\ \vdots & & \ddots & \vdots \\ b_{p1} & b_{p2} & \cdots & b_{pn} \end{bmatrix} = \begin{bmatrix} c_{11} & c_{12} & \cdots & c_{1n} \\ c_{21} & c_{22} & \cdots & c_{2n} \\ \vdots & & \ddots & \vdots \\ c_{m1} & c_{m2} & \cdots & c_{mn} \end{bmatrix},$$

em que $c_{ij} = \sum_{k=1}^{p} a_{ik} \cdot b_{kj}$ expressa um elemento de C dado pela multiplicação de uma linha de A por uma coluna de B.

O custo de multiplicar uma linha de A por uma coluna de B é equivalente à operação de produto escalar de dois vetores, e possui $2n-1$ flops e ordem $\mathcal{O}(n)$. Veja que isto gera apenas um elemento de C, então

precisamos multiplicar por n pra obter toda uma linha de C e posteriormente multiplicar por n para obter todas as colunas de C. Assim, são necessários $n \cdot n \cdot (2n-1) = (2n^3 - n^2)$ flops, e temos ordem $\mathcal{O}(n^3)$.

Vamos analisar agora, o custo para resolver um sistema usando a fatoração LU, veja que além da fatoração LU, temos que implementar também os algoritmos de substituição avançada e o de retro-substituição. Começaremos calculando o custos destes e depois analisaremos o custo da fatoração LU.

Seja L uma matriz triangular inferior $n \times n$ e b um vetor $n \times 1$. Podemos expressar o algoritmo de substituição direta por

Código 4.1 Código do algoritmo de substituição direta. Adaptado de [6]

```
1 function subdir(L,b)
2 n = size(L)
3 b[1] = b[1]/L[1,1]
4     for i = 2:n
5         b[i]=(b[i] - L[i, 1:(i-1)] * b[1:(i-1)])/L[i,i]
6     end
7 return b
8 end
```

Vamos contabilizar os flops. Na linha 3 temos uma operação de divisão, portanto $\mathcal{O}(1)$.

Contabilizaremos, agora, as operações do loop na linha 5. Quando tomamos $i = 2$ temos $b[2] = (b[2] - L[2, 1:1] * b[1:1])/L[2,2]$, que envolve uma operação de divisão, uma operação de multiplicação e uma operação de subtração, totalizando 3 flops. Quando $i = 3$, temos $b[3] = (b[3] - L[3, 1:2] * b[1:2])/L[3,3]$, temos uma subtração, duas multiplicações, uma soma e uma divisão, ou seja, $1 + 2 + 1 + 1 = 5$ flops. De modo geral, se tomarmos $i = n$, temos $b[n] = (b[n] - L[n, 1:n-1] * b[1:n-1])/L(n,n)$ temos uma subtração, $n-1$ multiplicações, $n-2$ somas uma divisão, ou seja, $1 + (n-1) + (n-2) + 1 = (2n-1)$ flops.

Somando todos os custos, obtemos

$$\sum_{k=1}^{n} 2(k-1) = 1 + 3 + 5 + 7 + \cdots + 2n - 1.$$

Usando a fórmula da soma da progressão aritmética temos que $S_{flops} = \frac{n\cdot(1+(2n-1))}{2} = \frac{n\cdot 2n}{2} = n^2$. Portanto, para aplicar o algoritmo da substituição direta, usamos n^2 flops, sendo assim, uma operação de ordem $\mathcal{O}(n^2)$.

Vamos agora calcular o custo de uma substituição reversa. Seja U uma matriz triangular superior $n \times n$ e b um vetor $n \times 1$. podemos expressar o algoritmo de substituição reversa no código abaixo.

Código 4.2 Código do algoritmo de substituição reversa. Adaptado de [6]

```
function subrev(U, b)
n = size(L)
    b[n] = b[n]/U[n,n]
    for i = (n-1):-1:1
        b[i] = (b[i] - U[i,(i+1):n]*b[(i+1):n])/U[i,i]
    return x
end
```

Veja que os dois algoritmos são bastantes similares e possuem a mesma quantia de flops, a explicação é análoga. Portanto, temos n^2 flops e ordem $\mathcal{O}(n^2)$.

Por fim, vejamos o quão custosa é, computacionalmente, a fatoração LU. Seja A $n \times n$, temos o seguinte código em `Julia` para fatoração LU.

Código 4.3 Código em `Julia` do algoritmo da fatoração LU

```
function LU(A::Matrix)
    n = size(A, 1)
    U = copy(A)
    L = diagm(fill(1, n))
    for k = 1:n-1
        for i = k+1:n
            L[i, k] = U[i, k] / U[k, k]
            for j = k:n
                U[i, j] -= L[i, k] * U[k, j]
            end
        end
    end
    return L, U
end
```

Para zerar os termos da primeira coluna, temos que calcular os coeficientes $\ell_{i1} = \frac{a_{i1}}{a_{11}}$, $i \in \{2, 3, 4, \ldots, n\}$ então multiplicá-lo com os termos $a_{12}, \ldots, a_{1n}$, da primeira linha e somar cada um com o seu termo respectivo da i-ésima linha. Totalizando então n multiplicações e $n-1$ subtrações, por linha. Como temos $n-1$ linhas, são realizadas $n \cdot (n-1)$ multiplicações contabilizando o custo dos multiplicadores e o custo da multiplicação pela linha que contém o pivô. São realizadas também, $(n-1) \cdot (n-1) = (n-1)^2$ subtrações.

Para zerar os termos da segunda coluna, realizamos um processo similar. Efetuamos $(n-1)$ produtos e $(n-2)$ subtrações. Como desejamos aplicar em todas as $n-2$ linhas da coluna 2, temos então $(n-1) \cdot (n-2)$ produtos e $(n-2) \cdot (n-2)$ subtrações.

Repetimos o processo até o termo $a_{n,(n-1)}$, o último termo a ser zerado. Daí, precisamos calcular o coeficiente $\ell_{n,(n-1)}$ e multiplicá-lo pelo elemento $a_{(n-1),n}$ e depois realizar a subtração com o termo $a_{n,(n-1)}$, totalizando então dois produtos e uma subtração. Veja que estamos contabilizando o cálculo do multiplicador como um produto pelo inverso multiplicativo do pivô.

Somando todas as subtrações que precisamos realizar, obtemos então

$$(n-1)^2 + (n-2)^2 + \cdots + 1^2 = \sum_{i=1}^{n-1} i^2,$$

e, manipulando o somatório obtemos $\sum_{i=1}^{n-1} i^2 = \sum_{i=1}^{n} i^2 - n^2$. Perceba que esse somatório é fácil de calcular, pois, $\sum_{i=1}^{n} i^2$ é o número piramidal quadrado. Assim,

$$\begin{aligned}
\sum_{i=1}^{n} i^2 - n^2 &= \frac{n \cdot (n+1) \cdot (2n+1)}{6} - n^2 \\
&= \frac{2n^3 + 3n^2 + n}{6} - n^2 \\
&= \frac{2n^3 + 3n^2 + n - 6n^2}{6} \\
&= \frac{2n^3 - 3n^2 + n}{6}.
\end{aligned}$$

Somando as multiplicações que precisamos efetuar, obtemos

$$\begin{aligned}
n \cdot (n-1) + \cdots + 2 \cdot 1 &= \sum_{i=1}^{n-1} (i \cdot (i+1)) \\
&= \sum_{i=1}^{n-1} \left(i^2 + i\right) \\
&= \sum_{i=1}^{n-1} i^2 + \sum_{i=1}^{n-1} i
\end{aligned}$$

veja que, $\sum_{i=1}^{n-1} i$ refere-se aos multiplicadores e $\sum_{i=1}^{n-1} i^2$ refere-se a multiplicação

destes pelas linhas que contêm o pivô. Assim,

$$\begin{aligned}\sum_{i=1}^{n-1} i^2 + \sum_{i=1}^{n-1} i &= \frac{n\cdot(n+1)\cdot(2n+1)}{6} - n^2 + \frac{n\cdot(n-1)}{2} \\ &= \frac{2n^3+3n^2+n}{6} - n^2 + \frac{n^2-n}{2} \\ &= \frac{2n^3+3n^2+n-6n^2+3n^2-3n}{6} \\ &= \frac{2n^3-2n}{6} \\ &= \frac{n^3-n}{3}.\end{aligned}$$

Assim o custo da fatoração LU é

$$\begin{aligned}&= \frac{2n^3-3n^2+n}{6} + \frac{n^3-n}{3} \\ &= \frac{2n^3-3n^2+n+2n^3-2n}{6} \\ &= \frac{4n^3-3n^2-n}{6}.\end{aligned}$$

Após calcularmos a LU, precisamos resolver dois sistemas triangulares, o $Ly = b$ e o $Ux = y$, já vimos que o custo para isto seria $2n^2$. Portanto, o custo total para resolver um sistema linear utilizando fatoração LU seria:

$$\begin{aligned}&= \frac{4n^3-3n^2-n}{6} + 2n^2 \\ &= \frac{4n^3-3n^2-n+12n^2}{6} \\ &= \frac{4n^3+9n^2-n}{6}.\end{aligned}$$

Vamos calcular quanto custa para inverter uma matriz, suponha que desejamos inverter uma matriz 2×2, a matriz $A = \begin{bmatrix}1 & 2\\ 3 & 4\end{bmatrix}$ por exemplo. Então, precisamos encontrar uma matriz do tipo $B = \begin{bmatrix}a & b\\ c & d\end{bmatrix}$ tal que

$A \cdot B = I$. Para isto, efetuamos a seguinte multiplicação

$$A \cdot B = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \cdot \begin{bmatrix} a & b \\ c & d \end{bmatrix} = \begin{bmatrix} 1a + 2c & 1b + 2d \\ 3a + 4c & 3b + 4d \end{bmatrix}$$

e depois igualamos a identidade

$$\begin{bmatrix} 1a + 2c & 1b + 2d \\ 3a + 4c & 3b + 4d \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}.$$

Assim obtemos dois sistemas lineares de duas incógnitas e duas variáveis

$$\begin{cases} 1a + 2c = 1 \\ 3a + 4c = 0 \end{cases} \text{ e } \begin{cases} 1b + 2d = 0 \\ 3b + 4d = 1 \end{cases}.$$

Similarmente, se desejamos inverter uma matriz 3×3, precisamos resolver 3 sistemas lineares de 3 equações cada e 3 incógnitas cada, e assim por diante. Portanto, para inverter uma matriz $n \times n$ precisaríamos resolver n sistemas lineares com n equações e n incógnitas. Assim, o custo final para inverter uma matriz utilizando a fatoração LU seria o custo de uma fatoração LU que é $\frac{4n^3-3n^2-n}{6}$ mais o custo de resolver n sistemas lineares triangulares utilizando a substituição avançada e reversa que seria $n \cdot 2n^2 = 2n^3$. Portanto, o custo final para se inverter uma matriz utilizando a fatoração LU seria $\frac{4n^3-3n^2-n}{6} + 2n^3 = \frac{16x^3-3x^2-x}{6}$.

O fazemos é calcular a fatoração LU e resolver estes n sistemas lineares utilizando a fatoração computada. Assim, dividimos nosso problema de inversão da matriz em n cálculos de sistemas do tipo $Av = e_i$ em que v é uma coluna da matriz inversa que queremos encontrar e e_i é uma coluna da identidade de ordem n.

5 RESOLUÇÃO DE SISTEMAS LINEARES EM $\mathbb{Z}_p$

Já conceituamos algumas definições, propriedades e teoremas a respeito do corpo $\mathbb{Z}_p$, de matrizes e de sistemas lineares. Vamos entender agora como se comportam as matrizes e os sistemas lineares no corpo dos inteiros módulo p. Para elaboração deste capítulo, foram usadas as referências [1, 15, 17].

5.1 O ESPAÇO VETORIAL $\mathbb{Z}_p^n$ SOBRE $\mathbb{Z}_p$

Por uma questão didática, retomaremos alguns conceitos iniciais e elementares a respeito de espaços vetoriais para, posteriormente, estabelecer algumas características de um espaço vetorial sobre um corpo finito $\mathbb{Z}_p$.

A terna $(\mathbb{E}, +, \cdot)$ é dita um *espaço vetorial* sobre um corpo $\mathbb{K}$ se $\mathbb{E}$ contém o elemento $0_{\mathbb{K}}$ que denominaremos elemento neutro e se as duas operações $+ : \mathbb{E} \to \mathbb{E}$ que chamaremos de soma e $\cdot : \mathbb{E} \to \mathbb{E}$ que chamaremos de multiplicação por escalar satisfizerem as seguintes propriedades:

i) $\forall\ u, v \in \mathbb{E},\ u + v = v + u$;

ii) $\forall\ u, v, w \in \mathbb{E},\ u + (v + w) = (u + v) + w$;

iii) $\forall\ u \in \mathbb{E}$, existe um único $0_{\mathbb{E}}$ tal que $u + 0_{\mathbb{E}} = 0_{\mathbb{E}} + u = u$;

iv) $\forall\ u \in \mathbb{E}$, existe $(-u) \in \mathbb{E}$ de modo que $u + (-u) = (-u) + u = 0_{\mathbb{E}}$;

v) $\forall\ \alpha \in \mathbb{K}$ e $\forall\ u, v \in \mathbb{E},\ \alpha \cdot (u + v) = (\alpha \cdot u) + (\alpha \cdot v)$;

vi) $\forall\ \alpha, \beta \in \mathbb{K}$ e $\forall\ u, v \in \mathbb{E},\ (\alpha \cdot \beta) \cdot u = \alpha \cdot (\beta \cdot u)$;

vii) $\forall\ \alpha, \beta \in \mathbb{K}$ e $\forall\ u, v \in \mathbb{E},\ (\alpha + \beta) \cdot u = (\alpha \cdot u) + (\beta \cdot u)$;

viii) $\forall\ u \in \mathbb{E},\ 1 \cdot u = u$ em que $1 \in \mathbb{K}$ é o escalar dito como elemento neutro da multiplicação.

Ao invés de dizer que a terna $(\mathbb{E}, +, \cdot)$ é um espaço vetorial, faremos um abuso de notação e diremos que $\mathbb{E}$ é um espaço vetorial.

Considere $\mathbb{E}$ um espaço vetorial sobre um corpo $\mathbb{K}$ qualquer.

Sejam $v_1, v_2, \cdots, v_n \in \mathbb{E}$, dizemos que o conjunto $\{v_1, v_2, \cdots, v_n\}$ é composto de vetores *linearmente independentes* se existirem $\alpha_1, \alpha_2, \cdots, \alpha_n \in \mathbb{K}$, em que $\mathbb{K}$ é um corpo qualquer, tal que a combinação linear nula

$$\alpha_1 \cdot v_1 + \alpha_2 \cdot v_2 + \cdots + \alpha_n \cdot v_n = 0_{\mathbb{E}}$$

admita uma única solução $\alpha_1 = \alpha_2 = \cdots = \alpha_n = 0_{\mathbb{K}}$. Caso contrário, dizemos que $v_1, v_2, \cdots, v_n$ são *linearmente dependentes*.

Seja $S \subset \mathbb{E}$ um subconjunto, o *conjunto gerado por* S ou *span de* S, denotado por $\text{span}(S)$, é o conjunto de todas as possíveis combinações lineares finitas dos elementos de S.

Perceba que um subconjunto $\mathbb{F} \subset \mathbb{E}$ é um espaço vetorial de $\mathbb{E}$ se for ele mesmo um espaço vetorial com as operações herdadas de $\mathbb{E}$.

Dizemos que $\mathcal{B} \subset \mathbb{F}$, $\mathcal{B} = \{v_1, v_2, \ldots, v_n\}$ é uma *base* para $\mathbb{F}$ se valem as seguintes propriedades:

i) $\mathcal{B}$ é linearmente independente;

ii) $\text{span}(\mathcal{B}) = \mathbb{F}$.

Assim, dada uma base $\mathcal{B}$ de $\mathbb{E}$, como $\mathcal{B}$ gera $\mathbb{E}$, todo elemento de $\mathbb{E}$ pode ser escrito como combinação linear finita dos vetores da base, ou seja,

$$\alpha_1 \cdot v_1 + \alpha_2 \cdot v_2 + \ldots + \alpha_n \cdot v_n = \sum_{i=1}^{n} \alpha_i \cdot v_i,$$

com $\alpha_i \in \mathbb{K}$.

A *dimensão* de um espaço vetorial $\mathbb{E}$, denotada por $\dim(\mathbb{E})$, é o número de elementos de uma base de $\mathbb{E}$.

Para garantirmos uma linguagem comum com o leitor, vamos relembrar os conceitos de núcleo e imagem de uma matriz.

Seja $A \in \mathbb{K}^{m \times n}$, o *espaço-coluna* ou *imagem de* A é o conjunto $Im(A) = \{b \in \mathbb{K}^m \mid b = \alpha_1 a_1 + \cdots + \alpha_n a_n\}$ em que $a_1, a_2, \ldots, a_n \in \mathbb{K}^m$ são as colunas de A. O *espaço nulo* ou *núcleo* ou *kernel* de A é o conjunto de vetores que zeram A, isto é, $\mathcal{N}(A) = \{x \in \mathbb{K}^n \mid Ax = 0_{\mathbb{E}}\}$.

Com efeito, $\dim(\mathcal{N}(A)) = n$ é o número de variáveis livres de A e $\dim(Im(A)) = r$ é o número de pivôs de A; veja mais sobre a relação entre variáveis livres, pivôs e os espaços fundamentais de uma matriz em [18, Sec. 3.2-3.4].

Se x^* é uma solução de $Ax = b$, com $A \in \mathbb{K}^{m\times n}$ e $u \in \mathcal{N}(A)$, então, $x^* + u$ é solução de $Ax = b$. De fato, $A(x^* + u) = Ax^* + Au = b + 0 = b$.

Também é fácil provar que o sistema linear $Ax = b$ possui solução se, e somente se, $b \in Im(A)$. Mais que isso, para que a solução seja única, $\mathcal{N}(A) = \{0_{\mathbb{K}^n}\}$. Isto se dá pelo Teorema 4.39.

Suponha $\mathcal{N}(A) \neq 0$, então existe uma base $\mathcal{B} = \{u_1, u_2, \ldots, u_k\}$. Se $u \in \mathcal{N}(A)$, então podemos escrever $u = \alpha_1 u_1 + \cdots + \alpha_k u_k$, em que u é solução geral de $Ax = 0$. Se o sistema linear $Ax = b$ possui solução, existe x_p uma solução particular tal que $Ax_p = b$. Então, $x_p + (\alpha_1 u_1 + \cdots + \alpha_k u_k)$ é a solução geral de $Ax = b$.

Estes resultados enunciados acima são fundamentais. O próximo resultado, conhecido como teorema do núcleo e da imagem de matrizes está formalmente enunciado e sua demonstração pode ser encontrada em [14, p.198-199].

Teorema 5.1. *Seja $A \in \mathbb{K}^{m\times n}$, $k = \dim(\mathcal{N}(A))$ o número de variáveis livres de A e $r = \dim(Im(A))$ o número de pivôs de A então, $n = k + r$.*

Tendo agora revisado alguns conceitos essenciais sobre espaços vetoriais, vamos na sequência, analisar alguns resultados acerca de espaços vetoriais sobre $\mathbb{Z}_p$. Aqui, utilizaremos o livro de Shokranian [17], o qual é uma referência interessante com vários exemplos de álgebra linear em corpos finitos.

Teorema 5.2. *Um espaço vetorial $\mathbb{E}$ sobre um corpo $\mathbb{K}$ tem um número finito de elementos (vetores) se, e somente se, $\dim_{\mathbb{K}}(\mathbb{E})$ é finita e $\mathbb{K}$ é um corpo finito.*

Demonstração. Suponha que $\mathbb{E}$ possua um número finito de elementos. Isto implica que $\dim_{\mathbb{K}}(\mathbb{E})$ é finito. Suponha também que $\mathbb{K}$ não seja finito,

o que implica que existem infinitos vetores em $\mathbb{E}$, pois existem infinitas combinações lineares de vetores da base de $\mathbb{E}$ com coeficientes da combinação linear que variam no corpo infinito $\mathbb{K}$. Contradição! Logo, $\mathbb{K}$ é finito.

Suponha agora que $\dim_{\mathbb{K}}(\mathbb{E})$ seja finita e que $\mathbb{K}$ seja um corpo finito. Provaremos que $\mathbb{E}$ possui um número finito de vetores. Suponha que $\mathbb{E}$ possui infinitos vetores e que $\dim_{\mathbb{K}}(\mathbb{E}) = k$, $k \in \mathbb{N}$, então $\mathbb{K}$ terá que ser um corpo infinito, pois todos os vetores de $\mathbb{E}$ são obtidos como uma combinação linear $\alpha_1 v_1 + \cdots + \alpha_k v_k$ em que $\alpha_1, \ldots \alpha_k \in \mathbb{K}$ e $v_1, \ldots, v_k$ são vetores da base. Novamente chegamos a uma contradição, demonstrando assim, o teorema em questão. ■

Lema 5.3. *Seja $\mathbb{K}_q$ um corpo finito com q elementos. Então, o espaço vetorial $\mathbb{K}_q^n$ possui q^n vetores.*

Demonstração. Vamos provar por indução. Tome $n = 1$, neste caso é verdade, pois $\mathbb{K}_1^n$ possui 1 elemento, pois $\mathbb{K}_q$ é um espaço vetorial sobre si e possui exatamente q vetores. Assuma ser verdade para $n = k - 1$, ou seja, o espaço vetorial $\mathbb{K}_q^{k-1}$. Neste caso, considere todos os vetores $v_j = (x_1, x_2, \ldots, x_{k-1})$ de $\mathbb{K}_q^{k-1}$, em que as entradas x_i são de $\mathbb{K}_q$, para $i = 1, 2, \ldots, q^{k-1}$. Tome agora o caso $n = k$, acrescentando a cada vetor v_j todos os elementos de $\mathbb{K}_q$, assim é gerado $q \cdot q^{k-1} = q^k$, em que $v_j(t) = (x_1, x_2, \ldots, x_{k-1}, t)$ são todos os vetores de $\mathbb{K}_q^k$ quando t varia em $\mathbb{K}_q$. ■

A partir daqui, os resultados apresentados serão todos considerando o corpo finito $\mathbb{Z}_p$.

Considere agora o espaço vetorial $\mathbb{Z}_p^n$, que é o conjunto de todas as n-uplas com componentes em $\mathbb{Z}_p$. Este conjunto é um espaço vetorial sobre $\mathbb{Z}_p$ com as operações de adição por coordenadas $+$ e multiplicação escalar $\cdot$, ou seja, se

$$x = \begin{bmatrix} \overline{x_1} \\ \overline{x_2} \\ \vdots \\ \overline{x_n} \end{bmatrix} \text{ e } y = \begin{bmatrix} \overline{y_1} \\ \overline{y_2} \\ \vdots \\ \overline{y_n} \end{bmatrix} \in \mathbb{Z}_p^n \text{ e } \overline{\alpha} \in \mathbb{Z}_p,$$

então,

$$x + y = \begin{bmatrix} \overline{x_1} + \overline{y_1} \\ \overline{x_2} + \overline{y_2} \\ \vdots \\ \overline{x_n} + \overline{y_n} \end{bmatrix} \text{ e } \overline{\alpha} \cdot x = \begin{bmatrix} \overline{\alpha} \cdot \overline{x_1} \\ \overline{\alpha} \cdot \overline{x_2} \\ \vdots \\ \overline{\alpha} \cdot \overline{x_n} \end{bmatrix}.$$

Exemplo 5.4. Observe que o espaço vetorial $\mathbb{Z}_2^3$ é finito e possui $2^3 = 8$, elementos que são:

- $v_0 = \begin{bmatrix} \overline{0} & \overline{0} & \overline{0} \end{bmatrix}^T$;
- $v_1 = \begin{bmatrix} \overline{1} & \overline{0} & \overline{0} \end{bmatrix}^T$;
- $v_2 = \begin{bmatrix} \overline{0} & \overline{1} & \overline{0} \end{bmatrix}^T$;
- $v_3 = \begin{bmatrix} \overline{1} & \overline{1} & \overline{0} \end{bmatrix}^T$;
- $v_4 = \begin{bmatrix} \overline{0} & \overline{0} & \overline{1} \end{bmatrix}^T$;
- $v_5 = \begin{bmatrix} \overline{1} & \overline{0} & \overline{1} \end{bmatrix}^T$;
- $v_6 = \begin{bmatrix} \overline{0} & \overline{1} & \overline{1} \end{bmatrix}^T$;
- $v_7 = \begin{bmatrix} \overline{1} & \overline{1} & \overline{1} \end{bmatrix}^T$.

5.2 MATRIZES E SISTEMAS LINEARES EM $\mathbb{Z}_p$

Como já vimos no Capítulo 2, $\mathbb{Z}_p$ é um corpo, assim como os reais. Logo, grande parte do que foi feito no Capítulo 4, é válido também para $\mathbb{Z}_p$.

Uma matriz sobre um corpo finito $\mathbb{Z}_p$ é uma matriz $m \times n$ de entradas em $\mathbb{Z}_p$. Um sistema linear em $\mathbb{Z}_p$ é um sistema cujas variáveis e coeficientes pertencem a $\mathbb{Z}_p$.

Exemplo 5.5. Tome a equação linear $x_1 + x_2 + x_3 = \overline{1}$, em que x_1, x_2 e x_3 pertencem a $\mathbb{Z}_2$. Vemos que essa equação possui exatamente quatro soluções:

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} \overline{1} \\ \overline{0} \\ \overline{0} \end{bmatrix}, \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} \overline{0} \\ \overline{1} \\ \overline{0} \end{bmatrix}, \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} \overline{0} \\ \overline{0} \\ \overline{1} \end{bmatrix} \text{ e } \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} \overline{1} \\ \overline{1} \\ \overline{1} \end{bmatrix}.$$

Os sistemas lineares funcionam de modo similar no corpo dos inteiros módulo p: podemos também usar os métodos abordados no capítulo Capítulo 4 para resolver sistemas lineares no corpo $\mathbb{Z}_p$, visto que todas as operações estão bem definidas, como mostrado no Capítulo 2.

Apresentemos então, primeiramente, um exemplo mais detalhado.

Exemplo 5.6. Seja $A = [a_{ij}]_{m\times n}$ com entradas em $\mathbb{Z}_{11}$

$$A = \begin{bmatrix} \overline{2} & \overline{1} & \overline{4} \\ \overline{3} & \overline{2} & \overline{1} \\ \overline{8} & \overline{3} & \overline{5} \end{bmatrix}.$$

Vamos fatorar A utilizando a fatoração LU.

Primeiro identificamos o pivô da primeira coluna, no caso $a_{11} = \overline{2}$, e zeramos todos os elementos abaixo do pivô $a_{21} = \overline{3}$ e $a_{31} = \overline{8}$. Para isto, precisamos encontrar ℓ_{21} de modo que $a_{21} + \ell_{21} \cdot a_{11} = \overline{0}$. Neste caso, encontramos o inverso aditivo de a_{21} em $\mathbb{Z}_{11}$, sendo assim, o inverso aditivo de $\overline{3}$ que é $\overline{8}$, pois $\overline{3} + \overline{8} = \overline{11} = \overline{0}$. Então, precisamos encontrar ℓ_{21} de modo que $\ell_{21} \cdot a_{11} = \overline{8}$. Para isso, calculamos o inverso multiplicativo de a_{11} em $\mathbb{Z}_{11}$, que em nosso caso seria o inverso multiplicativo de $\overline{2}$ que é $\overline{6}$ e efetuamos a operação $\overline{6}\cdot\overline{8} = \overline{4}$, deste modo, $\ell_{21} = \overline{4}$. Assim, $a_{21} + \ell_{21} \cdot a_{11} \Rightarrow \overline{3} + \overline{4} \cdot \overline{2} = \overline{3} + \overline{8} = \overline{0}$. Por fim, aplicamos as mesmas operações nos elementos a_{22} e a_{23}. Assim, $a_{22} = \overline{2} + \overline{4} \cdot \overline{1} = \overline{6}$ e $a_{23} = \overline{1} + \overline{4} \cdot \overline{4} = \overline{6}$.

Agora precisamos encontrar ℓ_{31} de modo que $a_{31} + \ell_{31} \cdot a_{11} = \overline{0}$. Sabemos que o inverso aditivo de $\overline{8}$ em $\mathbb{Z}_{11}$ é $\overline{3}$. Deste modo, precisamos encontrar ℓ_{31} tal que $\ell_{31}\cdot\overline{2} = \overline{3}$. Para isto, utilizamos o inverso multiplicativo

de a_{11}, já calculado, que é $\overline{6}$ e efetuamos a operação $\overline{6} \cdot \overline{3} = \overline{7}$, deste modo $\ell_{31} = \overline{7}$. Assim, $a_{31} + \ell_{31} \cdot a_{11} = \overline{8} + \overline{7} \cdot \overline{2} = \overline{8} + \overline{3} = \overline{0}$. Por fim, aplicamos as mesmas operações nos elementos a_{32} e a_{33}. Assim, $a_{32} = \overline{3} + \overline{7} \cdot \overline{1} = \overline{10}$ e $a_{33} = \overline{5} + \overline{7} \cdot \overline{4} = \overline{0}$.

$$A = \begin{bmatrix} \overline{2} & \overline{1} & \overline{4} \\ \overline{3} & \overline{2} & \overline{1} \\ \overline{8} & \overline{3} & \overline{5} \end{bmatrix} \begin{array}{l} \\ L_2 \leftarrow L_2 + \overline{4} \cdot L_1 \\ L_3 \leftarrow L_3 + \overline{7} \cdot L_1 \end{array} \sim \begin{bmatrix} \overline{2} & \overline{1} & \overline{4} \\ \overline{0} & \overline{6} & \overline{6} \\ \overline{0} & \overline{10} & \overline{0} \end{bmatrix}.$$

Novamente, como zeramos as entradas a_{12} e a_{13}, podemos guardar os coeficientes ℓ_{12} e ℓ_{13}, respectivamente. Assim,

$$\left[\begin{array}{ccc} \overline{2} & \overline{1} & \overline{4} \\ \hline \overline{4} & \overline{6} & \overline{6} \\ \overline{7} & \overline{10} & \overline{0} \end{array}\right].$$

Agora, identificamos o pivô da segunda coluna $a_{22} = \overline{6}$ e desejamos zerar o elemento $a_{32} = \overline{10}$. Para isso, precisamos encontrar ℓ_{32} de modo que $a_{32} + \ell_{32} \cdot a_{22} = \overline{0}$. Primeiramente encontramos o inverso aditivo de a_{32} em $\mathbb{Z}_{11}$, no caso, o inverso aditivo de $\overline{10}$ é $\overline{1}$, pois $\overline{10} + \overline{1} = \overline{11} = \overline{0}$. Então, precisamos encontrar ℓ_{32} de modo que $\ell_{32} \cdot a_{22} = \overline{1}$. Para isto, calculamos o inverso multiplicativo de a_{22} em $\mathbb{Z}_{11}$, que em nosso caso, seria o inverso multiplicativo de $\overline{6}$ que é $\overline{2}$, e efetuamos a operação $\overline{2} \cdot \overline{1} = \overline{2}$, deste modo, $\ell_{32} = \overline{2}$. Assim, $a_{32} + \ell_{32} \cdot a_{22} \Rightarrow \overline{10} + \overline{2} \cdot \overline{6} = \overline{10} + \overline{1} = \overline{0}$.

Por fim, aplicamos as mesmas operações no elemento a_{33}. Assim, $a_{33} = \overline{0} + \overline{2} \cdot \overline{4} = \overline{1}$.

$$\left[\begin{array}{ccc} \overline{2} & \overline{1} & \overline{4} \\ \hline \overline{4} & \overline{6} & \overline{6} \\ \overline{7} & \overline{10} & \overline{0} \end{array}\right] \quad L_3 \leftarrow L_3 + \overline{2} \cdot L_1 \quad \sim \left[\begin{array}{ccc} \overline{2} & \overline{1} & \overline{4} \\ \hline \overline{4} & \overline{6} & \overline{6} \\ \overline{7} & \overline{2} & \overline{1} \end{array}\right].$$

Desta forma, obtemos as seguintes matrizes L e U

$$L = \begin{bmatrix} \overline{1} & \overline{0} & \overline{0} \\ \overline{4} & \overline{1} & \overline{0} \\ \overline{7} & \overline{2} & \overline{1} \end{bmatrix}, U = \begin{bmatrix} \overline{2} & \overline{1} & \overline{4} \\ \overline{0} & \overline{6} & \overline{6} \\ \overline{0} & \overline{0} & \overline{1} \end{bmatrix}.$$

Provaremos a seguir um resultado importante sobre o número de soluções de sistemas lineares em $\mathbb{Z}_p$, que é uma consequência do famoso teorema do núcleo e da imagem. Como vimos, pelo fato de os reais serem um corpo infinito, um sistema linear que possui solução e para o qual a matriz dos coeficientes tem núcleo diferente do zero, terá infinitas soluções. Porém, isso não ocorre em $\mathbb{Z}_p$; o Teorema 5.7 garante que o número de soluções será sempre finito.

Teorema 5.7. *Seja A uma matriz $m \times n$ com elementos em $\mathbb{Z}_p$. Qualquer sistema de equações lineares consistente, com matriz dos coeficientes igual a A, tem exatamente $p^{n-\text{posto}(A)}$ soluções sobre $\mathbb{Z}_p$.*

Demonstração. Seja A uma matriz $m \times n$ com elementos em $\mathbb{Z}_p$. Suponha que A possui variáveis livres, então pelo Teorema do núcleo e da imagem, o número de variáveis livres será exatamente $\dim(\mathcal{N}(A)) = n - \dim(Im(A))$ em que n é o número total de variáveis, $\dim(\mathcal{N}(A)) = k$ é o número de variáveis livres de A e $\dim(Im(A)) = r$ o número de pivôs de A. Se $[A \mid b]$ for consistente a solução será dada por

$$x^* = x_P + x_N,$$

em que x_P é uma solução particular e x_N pertence ao $\mathcal{N}(A)$.

Seja $\{v_1, v_2, \ldots, v_k\}$ uma base do núcleo, logo, $x_N = \sum_{i=1}^{k} \alpha_i v_i$. Veja que se $\alpha_i \in \mathbb{R}$, o número de soluções é infinito, pois α_i pode assumir qualquer valor de $\mathbb{R}$. Porém, considere o corpo finito $\mathbb{Z}_p$; neste caso, α_i pode assumir apenas p valores. Então, temos apenas p^k combinações possíveis para vetores do núcleo de A. Portanto, temos $p^{n-\dim(Im(A))}$ soluções para um sistema linear consistente em $\mathbb{Z}_p$. ■

Note que se a matriz tiver posto completo, o teorema anterior ainda é válido, visto que neste caso $\dim(Im(A)) = n$ e logo, como no caso dos reais, tal sistema tem solução única.

Além disso, aplicando o Teorema 5.7 no sistema linear do Exemplo 5.5, vemos que como $p = 2$, $n = 3$ e $\dim(Im(A)) = 1$, o número de soluções do

sistema apresentado é dado por $2^{3-1} = 4$, o que condiz com o que havíamos apresentado no exemplo.

6 IMPLEMENTAÇÕES

Neste capítulo apresentamos as implementações dos algoritmos e estruturas que permitem usarmos o computador para realizarmos operações com objetos que utilizam elementos de $\mathbb{Z}_p$.

Para realizar as implementações, utilizamos a linguagem de programação `Julia`. Todos os códigos encontram-se disponíveis em https://github.com/TainaSilva3012/IntZp.jl.

A escolha dessa linguagem foi feita por conta de algumas de suas características que serão explanadas a seguir. Apresentaremos primeiro alguns conceitos de ciência da computação, tendo como referência [11–13].

Um *objeto* é um valor na memória referenciado por um identificador que possua atributos e consiga realizar ações. Pode ser uma variável, uma estrutura de dados, uma função ou um método.

Um *método* é um procedimento do objeto, ou seja, como esse objeto se comporta.

Um *tipo* em programação representa a natureza de objetos reais. Por exemplo, em `Julia`, o tipo `Int` que simboliza os números inteiros, ou o tipo `String` que representa uma palavra ou coleção de caracteres, ou o tipo `Bool` que caracteriza as variáveis booleanas de verdadeiro ou falso.

Um fato curioso sobre `Julia` é que "Os objetos carregam etiquetas de tipo e os próprios tipos são objetos `Julia` que podem ser criados e inspecionados em tempo de execução" [2, p.14, tradução nossa].

Vamos exemplificar os três conceitos recém aprendidos, se definirmos `num = 23`, teremos o objeto `num` com valor numérico 23. Podemos dizer que o objeto possui o método "pode ser somado a outros números inteiros". Suponha que pedimos ao programa para realizar a soma `num + 5.3`, o método não permite que a soma seja realizada, pois o método definido não permite a soma de um tipo inteiro com um tipo ponto flutuante (no caso, 5.3). Porém, se definirmos o método "pode ser somado a números ponto flutuante", conseguimos realizar a operação `num + 5.3` sem problemas. Além disto, a existência do segundo método não compromete o primeiro.

O *polimorfismo paramétrico* permite que um código seja escrito de forma genérica, utilizando variáveis no lugar de tipos, ou seja, o mesmo código pode funcionar em diferentes tipos. Por exemplo, se tenho as variáveis `a = 10` e `b = 26` do tipo `Int` e as variáveis `c = 7.2` e `d = 1.9` do tipo `Float64`, a operação de multiplicação · possui polimorfismo paramétrico, pois é possível realizar `a * b = 260` e `c * d = 13.68` sem haver nenhuma diferenciação da função · com relação ao tipo.

A linguagem `Julia` utiliza-se de tipagem opcional — ou seja, o comportamento padrão em `Julia` quando os tipos são omitidos é permitir que os valores sejam de qualquer tipo —, da compilação *just-time-in* — compila o programa enquanto o executa — e do que chamamos despacho múltiplo, do inglês *multiple dispatch* — que se tata de um polimorfismo paramétrico. Isto é, que permite definir uma função de forma genérica de modo que o próprio compilador decida como irá prosseguir na execução da função.

> O principal meio de abstração de Julia é o despacho múltiplo dinâmico. Grande parte de uma linguagem consiste em mecanismos para selecionar o código a ser executado em diferentes situações — desde a seleção de métodos até a seleção de instruções. Utilizamos apenas despacho múltiplo dinâmico para este fim, o que é possível através de regras de despacho suficientemente expressivas.[2, p.2, tradução nossa].

Por exemplo, se definimos uma função `length_string(x::String)`, cuja entrada é uma variável do tipo `String`, ou seja, uma palavra e a saída é a quantidade de caracteres que a `String` contém. Assim, se chamássemos a função com a entrada `length_string("linear")`, a saída seria 6, pois a palavra linear contem 6 caracteres. Suponha agora que desejamos criar a função `length(x::Array)`, em que a entrada é um vetor e a saída é o número de elementos do vetor. Assim, se chamássemos a função com a entrada `length([1 2 3])`, teríamos a saída 3, que é o comprimento do vetor indicado.

Com o despacho múltiplo, podemos simplesmente chamar a função `length` e a linguagem diferenciará as variáveis automaticamente, dando a

resposta de acordo. Sem haver a necessidade da tipagem ou de várias funções que fazem a mesma coisa, mas para tipos diferentes de variáveis. Pois, há uma única chamada que está associada a vários métodos.

Veja um exemplo de `Julia`

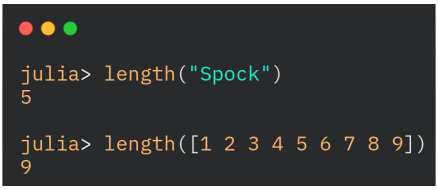


Figura 4 – Exemplo em `Julia` do despacho múltiplo.

Na verdade, o que ocorre é uma espécie de banco de dados em que há várias definições para a mesma função com diferentes tipos de métodos. Por exemplo, a função soma possui 190 métodos dentro do `Julia`, permitindo que o usuário some diversos tipos, inclusive tipos distintos.

6.1 CONSTRUINDO UM CORPO FINITO EM JULIA

Apresentaremos agora a construção do corpo $\mathbb{Z}_p$ em `Julia`. Tal linguagem possui outra característica bastante útil para este fim, os chamados construtores. Essa ferramenta nos permite criar um novo tipo de variável, o que nos possibilita adicionar um novo conjunto numérico à linguagem, além dos já existentes (inteiros, pontos flutuantes, etc.). Além disso, os construtores nos permitem importar funções já existentes na linguagem para podermos criar um método para a função, permitindo, assim, que a função seja executada no novo corpo. No nosso caso, decidimos adicionar o corpo dos inteiros módulo p, em que p é um número primo. Abaixo exibimos o código usado para definir o tipo inteiro módulo p.

Código 6.1 Código em `Julia` do construtor do corpo $\mathbb{Z}_p$

```
1  struct IntZp <: Number
2      n::Int
3      p::Int
4      function IntZp(n, p)
5          a = n % p
6      if a < 0
7          a += p
8      end
9          return new(a, p)
10     end
11 end
```

Note que na linha 1 do código acima, declaramos `IntZp <: Number`. Isso significa que `IntZp` é um subtipo associado ao tipo `Number`, e logo, todos os métodos, objetos e tipos que se utilizam de `Number`, também podem ser definidos e utilizados com `IntZp`. Além disso, optamos por trabalhar apenas com os inteiros positivos módulo p. Deste modo, sempre que um número negativo módulo p é inserido, já é realizada automaticamente a conversão para o menor representante positivo módulo p da classe (linhas 6-8). Por exemplo, se inserirmos `IntZp(-4, 11)`, este é automaticamente convertido para `IntZp(7, 11)`.

Definimos como `IntZp` os números pertencentes a $\mathbb{Z}_p$. Para exemplificar, vamos definir $a = \overline{5} \pmod{11}$ e $b = \overline{7} \pmod{11}$. Assim, `a.n` corresponde ao número ou classe de equivalência de `a` e `a.p` corresponde ao primo p.

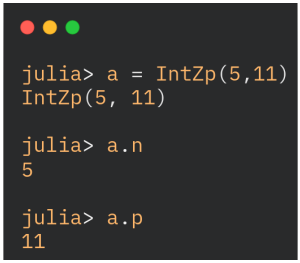


Figura 5 – Exemplo em `Julia` do tipo inteiro módulo p.

Para realizar as operações básicas de soma, subtração, multiplicação e divisão, importamos as funções da biblioteca base de `Julia` e criamos um método para `IntZp`. Para as funções de soma e subtração temos o seguinte código.

Código 6.2 Código em `Julia` dos métodos para as funções soma e subtração $\mathbb{Z}_p$.

```
import Base: +, -

function +(a::IntZp, b::IntZp)
    a.p != b.p && error("$a e $b não estão no mesmo corpo")
    x = (a.n + b.n) % a.p
    return IntZp(x, a.p)
end

function -(a::IntZp, b::IntZp)
    a.p != b.p && error("$a e $b não estão no mesmo corpo")
    x = (a.n - b.n) % a.p
    return IntZp(x, a.p)
end
```

Veja o novo método em execução na figura abaixo.

```
julia> a = IntZp(5,11)
IntZp(5, 11)

julia> b = IntZp(7,11)
IntZp(7, 11)

julia> a + b
IntZp(1, 11)

julia> a - b
IntZp(9, 11)
```

Figura 6 – Exemplo em `Julia` da soma e subtração módulo p.

Para a função de divisão, precisamos do inverso multiplicativo de um elemento em $\mathbb{Z}_p$. Para encontrar o inverso, utilizamos o algoritmo euclidiano estendido, estudado na Seção 1.3. Primeiro, implementamos o algoritmo euclidiano estendido com o código:

Código 6.3 Código em `Julia` do algoritmo euclidiano estendido em $\mathbb{Z}_p$.

```
function estendido(num::IntZp)
    a = num.n
    b = num.p
    x = 0
    x1 = 1
    y = 1
    y1 = 0
    r = b
    r1 = a
    while r != 0
        quociente = div(r1, r)
        (r1, r) = (r, r1 - quociente * r)
        (x1, x) = (x, x1 - quociente * x)
        (y1, y) = (y, y1 - quociente * y)
    end
    return x1
end
```

Em seguida, podemos definir as operações de multiplicação e divisão com os códigos abaixo:

Código 6.4 Código em `Julia` dos métodos para as funções multiplicação e divisão em $\mathbb{Z}_p$.

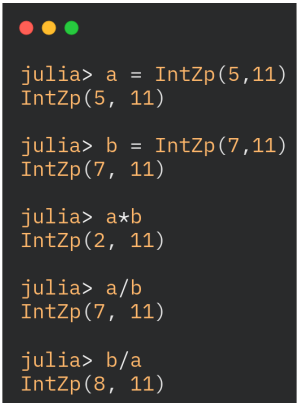
```
import Base: *, inv, /

function *(a::IntZp, b::IntZp)
    a.p != b.p && error("$a e $b não estão no mesmo corpo")
    x = (a.n * b.n) % a.p
    return IntZp(x, a.p)
end

function inv(num::IntZp)
    x = estendido(num)
    return IntZp(x, num.p)
end

function /(a::IntZp, b::IntZp)
    a.p != b.p && error("$a e $b não estão no mesmo corpo")
    b = inv(b)
    x = a.n * b.n
    return IntZp(x, a.p)
end
```

Veja o novo método definido em execução.



```
julia> a = IntZp(5,11)
IntZp(5, 11)

julia> b = IntZp(7,11)
IntZp(7, 11)

julia> a*b
IntZp(2, 11)

julia> a/b
IntZp(7, 11)

julia> b/a
IntZp(8, 11)
```

Figura 7 – Exemplo em `Julia` da multiplicação e divisão módulo p.

Em geral, é importante ressaltar que podemos usar o `Constructor` para importar funções de qualquer pacote existente e criar métodos para que funcionem em nosso novo tipo `IntZp`, o que faz com que $\mathbb{Z}_p$ possa ser usado como um conjunto numérico em `julia`. Assim, qualquer método programado para resolver sistemas lineares com números inteiros ou de ponto flutuante poderá também ser usado no corpo $\mathbb{Z}_p$. É isso que faremos a seguir.

6.2 APLICAÇÃO À CIFRA DE HILL

Como visto anteriormente, usando o `Constructor`, construímos um novo objeto `IntZp(n,p)` e definimos novos métodos para que as funções necessárias para a implementação pudessem atuar sobre o corpo $\mathbb{Z}_p$.

Vamos nos ater agora a implementação do algoritmo da cifra de Hill desenvolvido para um corpo finito $\mathbb{Z}_p$. Em nosso caso, utilizamos o corpo $\mathbb{Z}_{113}$. Como vimos, o processo de codificação depende de encontrar a inversa de uma matriz. Faremos isso usando a fatoração LU, em $\mathbb{Z}_p$. Para explicar toda nossa implementação, utilizaremos de um exemplo para ilustrar com mais clareza e didática o procedimento.

Dando início a codificação, começamos definindo a entrada do programa. Optamos por definir como um arquivo de texto que será lido e codificado. Do mesmo modo, a saída será um arquivo contendo o texto codificado. O mesmo acontece para a decodificação.

A mensagem que iremos cifrar é o poema de José Saramago, exibido abaixo.

```
Processo

As palavras mais simples, mais comuns,
As de trazer por casa e dar de troco,
Em língua doutro mundo se convertem:
Basta que, de sol, os olhos do poeta,
Rasando, as iluminem.

SARAMAGO, José. "Os poemas possíveis". 3 ed., Lisboa: Editorial Caminho, 1981.
```

Figura 8 – Poema de José Saramago utilizado nos testes.

O primeiro passo é escolher uma chave de criptografia que precisa ser uma matriz não singular, ou seja, inversível em $\mathbb{Z}_p$. Para isto, utilizamos a função exibida abaixo que sorteia os elementos que serão entradas da matriz no corpo $\mathbb{Z}_p$.

Código 6.5 Código em `Julia` da função que cria aleatoriamente uma chave.

```
function chave(n::Int,p::Int)
    C= diagm(fill(1, n))
    for j = 1:n
        C[j,j] = rand(1:p-1)
        for i = 1:n
            C[j, i] += rand(0:p-1)
        end
    end
    return IntZp(C, p)
end
```

Para realizar o experimento de maneira aleatória e em simultâneo, para não ficarmos trocando a matriz chave o tempo todo, usamos o pacote `Random` e o comando `Random.seed!(23)`, com o intuito de fixar a matriz aleatória obtida. Visto que a mensagem a ser cifrada contém 267 caracteres, utilizaremos $n = 7$. Como estamos trabalhando com 113 correspondências na Tabela 1, tomamos $p = 113$. Assim, utilizando o comando `chave_codifica = chave(7, 113)`, obtemos a matriz abaixo.

$$\texttt{chave\_codifica} = \begin{bmatrix} \overline{90} & \overline{82} & \overline{14} & \overline{24} & \overline{108} & \overline{68} & \overline{39} \\ \overline{15} & \overline{80} & \overline{67} & \overline{39} & \overline{112} & \overline{69} & \overline{48} \\ \overline{49} & \overline{79} & \overline{81} & \overline{7} & \overline{47} & \overline{22} & \overline{53} \\ \overline{15} & \overline{18} & \overline{6} & \overline{81} & \overline{103} & \overline{73} & \overline{95} \\ \overline{34} & \overline{56} & \overline{110} & \overline{79} & \overline{21} & \overline{110} & \overline{54} \\ \overline{96} & \overline{15} & \overline{82} & \overline{102} & \overline{19} & \overline{17} & \overline{15} \\ \overline{35} & \overline{48} & \overline{86} & \overline{49} & \overline{15} & \overline{59} & \overline{3} \end{bmatrix}.$$

Para garantir que a matriz escolhida como chave tenha inversa, aplicamos a função abaixo que utiliza a fatoração LU para calcular a inversa de uma matriz em $\mathbb{Z}_p$. Note que estamos utilizando o despacho múltiplo e definindo a função `inv` para matrizes com entradas do em $\mathbb{Z}_p$, usando o tipo `Matrix{IntZp}`.

Código 6.6 Código em `Julia` da função que calcula a inversa da matriz chave.

```
function inv(A::Matrix{IntZp})
    Ainv = copy(A)
    m, n = size(A)
    @assert m == n "A matriz dada não é quadrada"
    L, U = LU(A)
    a = A[1]
    for colA in eachcol(A)
        j = colA.indices[2]
        ej = zeros(a, n)
        ej[j] = IntZp(1, a.p)
        y = subdir(L, ej)
        Ainv[:,j] = subrev(U, y)
    end
    return Ainv
end
```

Utilizando o comando `chave_decodifica = inv(chave_codifica)`, geramos a seguinte matriz inversa:

$$\texttt{chave\_decodifica} = \begin{bmatrix} \overline{11} & \overline{28} & \overline{72} & \overline{44} & \overline{67} & \overline{17} & \overline{48} \\ \overline{67} & \overline{3} & \overline{70} & \overline{63} & \overline{84} & \overline{5} & \overline{0} \\ \overline{39} & \overline{81} & \overline{49} & \overline{13} & \overline{79} & \overline{42} & \overline{109} \\ \overline{45} & \overline{9} & \overline{98} & \overline{109} & \overline{101} & \overline{66} & \overline{76} \\ \overline{108} & \overline{2} & \overline{85} & \overline{27} & \overline{105} & \overline{37} & \overline{46} \\ \overline{42} & \overline{110} & \overline{107} & \overline{14} & \overline{6} & \overline{39} & \overline{67} \\ \overline{63} & \overline{68} & \overline{1} & \overline{21} & \overline{97} & \overline{66} & \overline{43} \end{bmatrix}.$$

Como a chave foi sorteada aleatoriamente, é interessante verificarmos se as matrizes que obtemos como chaves são de fato inversas. Para isto, computamos os produtos `chave_codifica * chave_decodifica` e `chave_decodifica * chave_codifica`. Se em ambos os casos obtemos a matriz identidade no corpo $\mathbb{Z}_{113}$, então a inversa está correta.

Agora que temos as chaves escolhidas, utilizamos a função abaixo para ler os arquivos, e utilizar a Tabela 1 para converter os caracteres do texto em números.

Para ler o arquivo `.txt` utilizamos a função abaixo.

Código 6.7 Código em `Julia` da função que lê arquivos `.txt`.

```
function lerarquivo(caminho)
    pre_string = open(f->read(f, String), caminho)
    string = Unicode.normalize(pre_string)
    return string
end
```

Para converter o arquivo em um vetor numérico, salvamos a tabela no formato `.txt` e a importamos com a função `lerarquivo` exibida abaixo.

Código 6.8 Código em `Julia` das funções utilizadas para conversão.

```
pre_tabela = lerarquivo("precodificacao.txt")
TABELA = split(pre_tabela,"")

function converte_char(letra)
    posicao = findall(x-> x==letra, TABELA)
    return posicao[1] - 1
end

function converte_texto(texto)
    texto_array = split(texto,"")
    array_convertido = converte_char.(texto_array)
    return array_convertido
end
```

O próximo passo no algoritmo de Hill é concatenar o vetor contendo os símbolos convertidos em uma matriz, cuja ordem seja compatível com a matriz que escolhemos como chave. Utilizamos a função abaixo, que tem como entrada o inteiro n, que é o mesmo da ordem da matriz, que é a chave de codificação. A função então cria uma matriz com n linhas. Em relação

ao número de colunas, efetuamos o cálculo do tamanho do vetor obtido e convertido e dividimos por n. O resto r obtido é a quantidade de colunas da matriz.

Quando o vetor convertido não contém exatamente $n \times r$ elementos, completamos a matriz com um caractere qualquer conforme mencionado anteriormente. Em nosso caso, escolhemos completar com o caractere @, mas poderíamos utilizar qualquer outro, ou mesmo, poderíamos sortear caracteres na tabela.

Código 6.9 Código em `Julia` da função que concatena o vetor.

```
function concatena_matrix(array::Array, n::Int)
    tamanho = length(array)
    resto = tamanho % n
    falta = n - resto
    finalzinho = repeat([111],falta)
    array_completado = [array; finalzinho]
    return reshape(array_completado,n,:)
end
```

Como nossa mensagem contém 267 caracteres e escolhemos `n`=7, temos uma matriz de tamanho 7×39, totalizando 273 elementos. Assim, nossa mensagem será completada com 6 caracteres.

Para finalizar o processo de criptografia, fazemos o produto da chave obtida pela matriz concatenada, obtendo uma matriz com a mensagem cifrada. Efetuamos então o processo de converter a matriz com a mensagem cifrada em um vetor novamente. Para tal, utilizamos a tabela de correspondência para converter os números em caracteres e escrevemos um arquivo de texto que será a saída. Para isto, utilizamos as funções dadas nos códigos abaixo.

Código 6.10 Código em `Julia` da função que converte uma matriz para um vetor.

```
function converte_matrix(matrix::Matrix)
    return reshape(matrix,1,:)
end
```

Código 6.11 Código em `Julia` da função que converte números para letras conforme a tabela de correspondência.

```
function converte_num(n::Int)
    return TABELA[n+1]
end

function converte_array(array)
    array_texto = converte_num.(array)
    return join(array_texto,"")
end
```

Código 6.12 Código em `Julia` da função que escreve um arquivo `.txt`.

```
function escrevearquivo(caminho, texto)
    open(caminho, "w") do io
        write(io, texto)
    end
end
```

Após aplicarmos as funções listadas acima, obtemos um arquivo de texto com a seguinte mensagem cifrada:

```
ntójGX0l-D5=ôDVséÇ6êÉ;Hb@@FoY(ÚÁ.ÇÊç\jYzÍ=iÃSlÇP
lbUy>úarVFPWFloGÔ%+Í7q4jâÁVXô-Y(#ó]
Nà
l;B\)qDDlbtq),fÕU>lj{V-E:BZõ?<Ê:=nÍÚ\(ágâ4zjh)8gjtÕ{8íRNg7Ôé=yH"tóCu7iMy,Gs5O/=3t1=Â
s79@gçEàD8F29ã].í3Mz#nÇIêHÔ.pÚ,ÓMAwó"l16táYEbZa3bÇ!
PÂ4ÀK46ÍçEmT?t ô1m/;Ád9];óCpz ÊHÚ)T;wSv [zÂnÀI{.
```

Figura 9 – Poema codificado.

Para o processo de descriptografar a mensagem, utilizamos o mesmo processo para ler o arquivo com a mensagem cifrada e convertê-la em números de acordo com a Tabela 1. Realizamos então a conversão da mensagem em uma matriz $n \times r$. Utilizando a matriz inversa, calculada com a função `inv`, multiplicamos a chave de descriptografia pela mensagem cifrada obtendo, assim, a mensagem descriptografada. Por fim, realizamos todo o processo de conversão para que a saída seja um arquivo de texto, conforme o exibido abaixo, finalizando, assim, o processo de descriptografia.

```
Processo

As palavras mais simples, mais comuns,
As de trazer por casa e dar de troco,
Em língua doutro mundo se convertem:
Basta que, de sol, os olhos do poeta,
Rasando, as iluminem.

SARAMAGO, José. "Os poemas possíveis". 3 ed., Lisboa: Editorial Caminho, 1981.@@@@@@
```

Figura 10 – Poema decodificado.

Veja que o caractere que colocamos para completar a matriz aparece no final da mensagem. Não utilizamos nenhuma função para removê-lo por que não sabíamos o tamanho da mensagem original, visto que temos acesso apenas a mensagem cifrada e a chave de criptografia. Sendo assim, só é possível excluir esses caracteres após a mensagem ser descriptografada.

7 CONSIDERAÇÕES FINAIS

Neste estudo, apresentamos alguns métodos para resolver sistemas lineares em $\mathbb{Z}_p$, bem como construímos uma estrutura capaz de reproduzir as propriedades do corpo finito $\mathbb{Z}_p$ computacionalmente incluindo algoritmos para solução destes sistemas lineares. Além disto, aplicamos nossa estrutura computacional para resolver cifras de Hill.

Para o desenvolvimento desta monografia, foram necessários os conhecimentos adquiridos durante o período de graduação nas disciplinas de álgebra — em que estudamos estruturas como anéis e corpos —, álgebra linear — na qual investigamos sistemas lineares e espaços vetoriais –, álgebra linear aplicada — em que realizamos implementações de métodos para resolução de sistemas lineares — e, métodos numéricos — quando compreendemos questões acerca de custo computacional. Além disto, outro fator contribuinte, foram as diversas iniciações científicas realizadas durante todo o período de graduação, que serviram de grande motivação para este trabalho.

Em nossa opinião, o resultado teórico mais importante que estudamos é o Teorema 5.7, que garante que num corpo finito $\mathbb{Z}_p$ caso um sistema seja consistente, possui um número finito de soluções. Fora isso, notamos que os sistemas lineares no corpo $\mathbb{Z}_p$ se comportam de modo muito similar ao corpo dos reais, ao menos para os métodos estudados.

O objetivo computacional pretendido com o trabalho também foi cumprido dada a construção do objeto `IntZp` e seus métodos relacionados na linguagem `Julia`, além de sua aplicação à criptografia de mensagens.

Considerando o escopo deste trabalho, nos atemos apenas a estudar o método da eliminação gaussiana, a fatoração LU e a fatoração $PA = LU$, e implementamos o método da fatoração LU. Sendo assim, é evidente que muito estudo ainda pode ser desempenhado em torno das questões apontadas aqui. Para um possível continuação destes estudos poderíamos seguir a partir de métodos diretos de fatoração, tais como fatoração de Cholesky, além de estudar a possibilidade de implementação de métodos iterativos

para sistemas lineares em corpos finitos. Outro tema possível é explorar outras aplicações que envolvam sistemas lineares em corpos finitos.

REFERÊNCIAS

[1] Aishah Ibraheam Basha. “Linear algebra over finite fields”. Tese de dout. Washington: Washington State University, dez. de 2020.

[2] Jeff Bezanson, Stefan Karpinski, Viral B Shah e Alan Edelman. “Julia: A fast dynamic language for technical computing”. Em: *arXiv preprint arXiv:1209.5145* (2012).

[3] José Luiz Boldrini, Sueli I Rodrigues Costa, Vera Lúcia Figueiredo e Henry G Wetzeler. *Álgebra linear*. São Paulo: Harbra-Harper & Row do Brasil, 1980.

[4] Mariana Martins Durães Brandão. “Uma adaptação da Cifra de Hill para estudo de matrizes.” Diss. de mestr. Ouro Preto: Universidade Federal de Ouro Preto. Programa de Pós-Graduação em Matemática em Rede Nacional, 2017.

[5] Severino Collier Coutinho. *Números Inteiros e Criptografia RSA*. Rio de Janeiro: IMPA/SBM, 2000, p. 213.

[6] Gene H Golub e Charles F Van Loan. *Matrix computations*. 4º edição. Johns Hopkins University Press, 2013.

[7] Alexandre Hartung. “Matrizes e resolução de problemas”. Diss. de mestr. São Carlos: Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, jun. de 2017.

[8] Abramo Hefez e Cecília de Souza Fernandez. “Introdução à álgebra linear”. Em: *Rio de Janeiro: SBM* (2012).

[9] Oscar Ricardo Janesch e Inder Jeet Taneja. *Álgebra I*. 2 ed. rev. Florianópolis: UFSC/EAD, 2011, p. 215.

[10] Steven J. Leon. *Algebra linear com aplicações*. Rio de Janeiro: LTC, 2011, p. 464.

[11] *Manual da linguagem Julia de programação / Methods*. Acesso em: 14 fevereiro 2022. Disp. em: https://docs.julialang.org/en/v1/manual/methods/#Methods.

[12] *Manual da linguagem Julia de programação / Types*. Acesso em: 14 fevereiro 2022. Disp. em: https://docs.julialang.org/en/v1/manual/types/#Composite-Types.

[13] *Manual da linguagem Julia de programação /Constructors*. Acesso em: 15 fevereiro 2022. Disp. em: https://docs.julialang.org/en/v1/manual/constructors/.

[14] Carl D. Meyer. *Matrix analysis and Applied linear algebra*. Philadelphia: SIAM, 2000, p. 869.

[15] David Poole. *Álgebra Linear: uma introdução moderna*. Tradução da 4º edição norte-americana. Cengage Learning, 2017, p. 722.

[16] Márcia A Gomes Ruggiero e Vera Lúcia da Rocha Lopes. *Cálculo numérico: aspectos teóricos e computacionais*. Makron Books do Brasil, 1997.

[17] Salahoddin Shokranian. *Exemplos de álgebra linear sobre corpos*. Rio de Janeiro: Editora Ciência Moderna, 2015, p. 315.

[18] Gilbert Strang. *Álgebra Linear e suas Aplicações*. 4ª ed. São Paulo: Cengage Learning, 2010.

[19] David S. Watkins. *Fundamentals of Matrix Computations. Pure and Applied Mathematics: A Wiley Series of Texts, Monographs and Tracts.* Hoboken: Wiley, 2010, p. 644.