



UNIVERSIDADE FEDERAL DO AMAPÁ
COLEGIADO DE MATEMÁTICA
MESTRADO PROFISSIONAL EM MATEMÁTICA

Linovaldo Coêlho dos Santos

**Uma ferramenta computacional para o cálculo e treinamento do método de
escalonamento de Gauss**

MACAPÁ

2016

Linovaldo Coêlho dos Santos

**Uma ferramenta computacional para o cálculo e treinamento do método de
escalonamento de Gauss**

Dissertação apresentada ao Programa de Mestrado Profissional em Matemática em Rede Nacional - PROFMAT da Universidade Federal do Amapá, como requisito parcial para obtenção do título de Mestre em Matemática.

Orientador Prof. Dr. José Walter Cárdenas Sotil

MACAPÁ

2016

Dados Internacionais de Catalogação na Publicação (CIP)
Biblioteca Central da Universidade Federal do Amapá

515.35

S237f Santos, Linovaldo Coêlho dos.

Uma ferramenta computacional para o cálculo e treinamento do método de escalonamento de Gauss / Linovaldo Coêlho dos Santos; orientador, José Walter Cárdenas Sotil. - Macapá, 2016.

103 f.

Dissertação (mestrado) - Fundação Universidade Federal do Amapá, Programa de Pós-Graduação em Matemática (PROFMAT).

1. Método de Gauss. 2. Software. 3. Equações diferenciais lineares. I. Sotil, José Walter Cárdenas, orientador. II. Fundação Universidade Federal do Amapá. III. Título.

Linovaldo Coêlho dos Santos

**Uma ferramenta computacional para o cálculo e treinamento do método de
escalonamento de Gauss**

Aprovado em 05 de Agosto de 2016

Banca examinadora

Prof. Dr. José Walter Cárdenas Sotil
Departamento de Ciências Exata e Tecnológica,
UNIFAP

Prof. Me. Astrogecildo Ubaiara Brito
Instituto Federal do Amapá,
IFAP

Prof. Dr. Guzman Eulálio Isla Chamilco
Departamento de Ciências Exata e Tecnológica,
UNIFAP

Prof. Dr. Erasmo Senger
Departamento de Ciências Exata e Tecnológica,
UNIFAP

MACAPÁ

2016

Agradecimentos

Agradeço primeiramente a Deus, pela vida, "todo aquele que ouve minhas palavras, e as põe em prática, será como um homem prudente que construiu sua casa sobre rocha"(Mt-7,24).

A minha família, que sempre esteve ao meu lado me apoiando, em especial minha esposa Luciana, e aos meus filhos Ingrid e Lucas, que sempre entenderam minhas ausências necessárias para estudo.

A Unifap que através do programa PROFMAT me deu a oportunidade de continuar ampliando meus conhecimentos matemáticos.

Ao Prof. Dr. José Walter Cárdenas Sotil, por toda sua inestimável colaboração neste projeto, sempre acreditando e incentivando.

Aos colegas da turma 2013 - PROFMAT, pela união e colaboração mútua.

A todos que direta ou indiretamente me incentivaram a continuar e finalizar o mestrado.

Por fim, à CAPES, pelo apoio financeiro.

"Verdadeiramente o que
mais prazer me proporciona,
não é o saber, mas o estudar,
não é a posse, mas a conquista,
não é o estar aqui, mas o chegar além."

(Carl Friedrich Gauss)

Resumo

Neste trabalho desenvolvemos o método de escalonamento de Gauss e mostramos a sua eficiência para calcular determinantes, matrizes inversas e sistemas de equações lineares quando comparados com o método de Laplace. O método de escalonamento de Gauss tem complexidade computacional da ordem polinomial, enquanto o método de Laplace segue uma complexidade de ordem exponencial. Foi desenvolvido no ambiente DELPHI 7 um aplicativo chamado MATRIZ, o qual calcula determinantes, matrizes inversas e sistemas de equações lineares algébricas como ferramenta computacional a ser usado em sala de aula. O aplicativo MATRIZ tem como diferencial possuir um ambiente de treinamento, onde o aluno pode decidir qual é o pivô e os multiplicadores, tornando as aulas mais dinâmicas, permitindo fixar os conceitos adquiridos, tomar decisões e obter rapidamente os resultados esperados.

Palavras-chave: Método de Gauss, Escalonar, Software, Matrizes, Determinantes, Matriz Inversa, Equações Lineares.

Abstract

In this work developed the scaling method of Gauss and show your efficiency to calculate determinants, inverse matrices and systems of linear equations compared with Laplacian method. The scaling method of Gauss has computational complexity of the polynomial order, while the method of Laplace follows a complexity of exponential order. It was developed in DELPHI 7 environment application called MATRIZ, which calculates determinants, inverse matrices and systems of algebraic linear equations as a computational tool to be used in the classroom. The application MATRIZ has the differential to have an training environment , where the student can decide which is the pivot and multipliers, making classes more dynamic, allowing fix the acquired concepts, take decisions and get expected results quickly.

Key words: Gauss method, Scale, Software, Matrices, Determinants, Inverse Matrice, Linear Equations.

Lista de Figuras

5.1	Tela Inicial	50
5.2	Tela Determinante	51
5.3	Tela Equações Lineares	52
5.4	Tela Matriz Inversa	53
5.5	Tela Treinamento Gauss	54
5.6	Primeiro passo - Determinante	55
5.7	Segundo passo - Determinante	55
5.8	Terceiro passo - Determinante	56
5.9	Quarto passo - Determinante	57
5.10	Primeiro passo - Equações Lineares	58
5.11	Segundo passo - Equações Lineares	58
5.12	Terceiro passo - Equações Lineares	59
5.13	Quarto passo - Equações Lineares	60
5.14	Primeiro passo - Matriz Inversa	61
5.15	Segundo passo - Matriz Inversa	61
5.16	Terceiro passo - Matriz Inversa	62
5.17	Quarto passo - Matriz Inversa	63
5.18	Primeiro passo - Treinamento Gauss	64
5.19	Segundo passo - Treinamento Gauss	64
5.20	Terceiro passo - Treinamento Gauss	65
5.21	Terceiro passo - Treinamento Gauss - 2	65
5.22	Quarto passo - Treinamento Gauss	66
5.23	Quinto Passo - Treinamento Gauss	67
5.24	Quinto passo - Treinamento Gauss - 2	67

Sumário

1	Introdução	12
2	Um pouco de História	14
2.1	Gauss, Príncipe da Matemática e seu raciocínio brilhante	14
2.2	Gauss e a Transformação Linear	16
2.3	Gauss e o Método da Eliminação	17
2.4	Sistemas de equações	18
2.5	Como surgiram as matrizes	19
3	Sistemas de Equações Lineares	20
3.1	Matriz	20
3.2	Tipos de matrizes	20
3.3	Operações com matrizes	22
3.4	Determinantes	24
3.4.1	Determinante de matrizes de ordem $n \geq 2$	25
3.4.2	Propriedades dos determinantes	27
3.5	Equação Linear	28
3.6	Sistema de equações lineares	29
3.6.1	Regra de Cramer	30
3.7	Sistema escalonado	31
3.7.1	Resolução de um sistema escalonado	31
3.8	Método de Gauss	32
3.9	Matriz inversa	35
4	Aspectos numéricos	38
4.1	Complexidade Computacional	38
4.1.1	Sistema diagonal	38
4.1.2	Sistema triangular superior	39

4.1.3	Eliminação Gaussiana	40
4.1.4	Determinantes por Cofatores	42
4.1.5	Matriz Inversa	45
4.2	Transformação de decimais em frações	45
4.2.1	Número inteiro em fração	46
4.2.2	Número decimal exato em fração	46
4.2.3	Dízima periódica em fração	46
4.3	Uma abordagem computacional para a representação de uma fração	48
5	Descrição do Aplicativo	50
5.1	Tela Inicial	50
5.2	Determinante	51
5.3	Equações Lineares	52
5.4	Matriz Inversa	53
5.5	Treinamento Gauss	54
5.6	Exemplo de Cálculo de Determinante	55
5.7	Exemplo de Cálculo de Equações Lineares	57
5.8	Exemplo de Cálculo de Matriz Inversa	60
5.9	Exemplo de Treinamento Gauss	63
	Considerações Finais	69
	Referências Bibliográficas	70
A	Programa em Delphi	72

Capítulo 1

Introdução

Os educadores sempre tiveram dificuldade em mostrar a aplicação dos conteúdos matemáticos, as aulas dadas de forma tradicional, com muita teoria e pouca ou nenhuma prática, com fórmulas prontas e sem significado para os alunos, torna a disciplina desinteressante. Precisamos tornar nossas aulas mais dinâmicas e interativas, para atrair o interesse dos nossos alunos. A informática na educação pode ser esse diferencial, pois nossos jovens estão cada vez mais integrados com a informática, tanto na sua disponibilidade quanto no que diz respeito à questão do seu manuseio.

A utilização dos computadores é irreversível, dada a diversidade de suas aplicações e capacidade de armazenar, organizar e processar informações. Na escola, essa realidade não é diferente: o computador renova as práticas docentes e institucionais.

Nos dias de hoje, o conhecimento por simulação está em alta, próprio da cultura informática, faz com que o computador seja também visto como um recurso didático cada dia mais indispensável, pensando nisso, elaboramos este trabalho com o objetivo principal de desenvolver um aplicativo para escalonar utilizando o método de Gauss e também fazer um treinamento desse método de modo a fixar a teoria. O software foi elaborado usando os conhecimentos de programação em cima da plataforma Borland Delphi 7 e os conhecimentos matemáticos de matrizes.

O primeiro capítulo contém a vida e algumas das obras de Carl Friedrich Gauss que, pelo seu trabalho, aplicação e conseqüentemente divulgação dessa forma de escalonar, recebeu seu nome no referido método, o qual mostrou-se muito eficiente em diversas aplicações e mostraremos especificamente o cálculo de determinantes, variáveis de equações lineares e matrizes inversas.

No segundo capítulo temos a fundamentação teórica iniciando com matrizes, dando uma visão geral do conteúdo, ou seja, pretende-se definir o que é uma matriz, alguns tipos de matrizes,

suas operações aritméticas, como também estabelecer algumas de suas propriedades algébricas e seu determinante. Em seguida verificamos as propriedades dos determinantes, equações lineares e sistema de equações lineares. Temos ainda, regra de Cramer, sistemas escalonados, método de Gauss e matrizes inversas com teoria e exemplos.

No capítulo 3 trataremos dos aspectos numéricos com o cálculo do custo computacional de um algoritmo com o objetivo de mostrarmos a eficiência do método de Gauss em relação ao método de Laplace no cálculo de determinantes. Temos também um estudo das transformações numéricas, em especial dos números decimais em frações, onde faremos uma abordagem diferente para essa transformação.

O último capítulo tem descrição do aplicativo com todas as telas, funcionalidades e exemplos com passo a passo da utilização dos recursos do sistema. Deve-se enfatizar a importância do Treinamento Gauss, que no exemplo é feito com uma matriz de quarta ordem, porém pode-se, teoricamente, utilizar matrizes bem maiores para a fixação do método.

Capítulo 2

Um pouco de História

2.1 Gauss, Príncipe da Matemática e seu raciocínio brilhante

De acordo com (BEMFICA, 2015), uma das histórias mais fascinantes envolvendo progressões é a do matemático alemão Carl Friedrich Gauss (1777-1855), considerado o maior matemático do século XIX e um dos maiores de todos os tempos, juntamente com Arquimedes e Isaac Newton. Filho de família pobre, seu pai tentou evitar que recebesse instrução adequada. Gauss, porém, contou com o apoio da mãe, para que pudesse estudar.

A precocidade de Gauss, tido como uma criança prodígio, pode ser exemplificada por um fato interessante ocorrido em sua infância. Aos 10 anos, Gauss frequentava uma escola local, na qual o professor era tido como muito exigente. Certo dia, com a intenção de manter a turma em silêncio, pediu aos alunos que somassem os números naturais de 1 a 100 ($1 + 2 + 3 + \dots + 100$) e, assim que terminassem, colocassem a solução sobre sua mesa. Quase que imediatamente, Gauss colocou sobre a mesa do professor a resposta encontrada. Ele olhou para o menino com pouco-caso, enquanto os demais alunos trabalhavam arduamente. Quando conferiu os resultados, o professor verificou que a única resposta correta era a de Gauss, 5.050, mas sem fazê-la acompanhar de nenhum cálculo.

Gauss havia feito o cálculo mentalmente, observando que a soma do primeiro e do último termo ($1+100$), do segundo e do penúltimo termo ($2+99$), do terceiro e do antepenúltimo ($3+98$), e assim por diante era sempre 101.

A soma dos números naturais de 1 a 100 é dada por $50 \cdot 101 = 5.050$, cálculo que resultou na fórmula da soma dos n primeiros termos de uma PA.

$$S_n = \frac{n(a_1 + a_n)}{2}$$

Verificamos em (USP, 2015), que o menino precoce Gauss, aos 12 anos, criticava os Funda-

mentos da Geometria; com 13 anos, projetava uma Geometria não-euclidiana; por volta dos 15 ou 16 anos de idade, descobriu o Teorema do Número Primo e concebeu a Lei Gaussiana - ou da Distribuição Normal - da Teoria das Probabilidades. Com 18 anos, inventou o método dos mínimos quadrados e, aos 22 anos, determinou as funções elípticas.

Gostava muito de estudar, mas estava indeciso entre tornar-se um filósofo ou um matemático. No dia 30 de março de 1796, ao que parece, essa decisão foi tomada, quando optou pela Matemática. Conseguiu construir, segundo as regras euclidianas, o polígono regular de dezesseis lados. Note-se que já eram conhecidas as construções, com régua e compasso, do triângulo equilátero e do pentágono regular, além de outros polígonos regulares, cujo número de lados fosse múltiplo de 2, 3 e 5, mas de nenhum outro com número primo de lados. A descoberta de Gauss foi publicamente anunciada numa revista literária.

Nesse mesmo dia, Gauss começou a escrever um diário composto por 19 páginas que talvez seja o documento mais importante de toda a História da Matemática. Nele encontram-se 146 breves enunciados de diversos resultados. O último enunciado tem data de 09 de julho de 1814. O conteúdo do diário só foi publicado em 1901 pelo matemático Felix Klein.

Uma das obras mais importantes de Gauss é a *Disquisitiones Arithmeticae* publicada em 1801, considerada o marco inicial da moderna Teoria dos Números, além de ser importantíssimo por trazer uma abordagem rigorosa e moderna da Matemática. Salvo alguns poucos resultados matemáticos antigos, o trabalho é inteiramente original. Na parte inicial se encontra a primeira demonstração do Teorema Fundamental da Aritmética, segundo o qual todo inteiro $n > 1$ pode ser escrito de forma única como um produto de primos. Na parte central fala-se da congruência quadrática, formas e resíduos, e na última seção encontra-se a teoria do polinômio ciclotômico com suas aplicações para a construtibilidade de polígonos regulares.

Em sua tese de doutorado, publicada em 1799 em Helmstädt, encontra-se uma demonstração do Teorema Fundamental da Álgebra. Esse teorema garante que toda equação polinomial de grau n admite n raízes complexas e, em sua demonstração Gauss utilizou números complexos e a geometria do plano complexo com total segurança, além de inaugurar a era das demonstrações de existência, importantes para a Matemática pura. A demonstração apresentada em sua tese baseia-se em parte em considerações geométricas. Em 1816, Gauss publicou duas demonstrações e, em 1850, publicou uma terceira, esforçando-se para encontrar uma prova inteiramente algébrica.

Outro grande trabalho foi um artigo publicado em 1830 sobre resíduos biquadráticos, cujas

idéias inauguraram a Teoria Algébrica dos Números. A partir de 1830, Gauss se ocupou com a Física, realizando estudos em diversos ramos, como a Óptica, onde introduziu o conceito de comprimento focal de um sistema de lentes e inventou as lentes grandes angulares de Gauss para telescópios e objetivas.

Embora tenha publicado muito, vários de seus estudos não o foram, pois Gauss preferia mergulhar em um novo estudo em lugar de escrever sobre as descobertas feitas.

Gauss passou os anos de 1845 a 1851 atualizando os fundos monetários da Universidade de Göttingen. Esse trabalho lhe deu uma experiência em práticas financeiras e, com isso, fez sua fortuna através de investimentos astutos em companhias privadas.

Conforme (UFF, 2015), Gauss atuou em muitas áreas das ciências exatas. Trabalhou em álgebra, teoria dos números, equações diferenciais, teoria de funções elípticas, cartografia, pesquisou o campo magnético terrestre, participou do desenvolvimento do primeiro telégrafo elétrico, contribuiu para a física-matemática com trabalhos em eletromagnetismo e gravitação, além de inúmeros outros tópicos aos quais dedicou suas pesquisas.

Gauss casou-se duas vezes. A primeira esposa, Johanna, deu-lhe três filhos, mas faleceu cedo, em 1809. Seu segundo casamento foi com Minna Waldeck, filha de outro professor de Göttingen, que lhe deu mais três descendentes.

Gauss morreu em Göttingen no dia 23 de Fevereiro de 1855, coincidindo com o incremento da Revolução Industrial. A crença oficial no progresso pacífico começava a ser substituída pela realidade de uma época de crises. Daí em diante, a figura do cientista integral, interessado em todos os aspectos do conhecimento humano, se tornou praticamente uma raridade. Por isso, o desaparecimento de Gauss marcou o fim de uma era.

2.2 Gauss e a Transformação Linear

Segundo (DIEUDONNÉ, 1981), o conceito de transformação linear ou a representação de uma variável como combinação de outras variáveis tornou-se familiar para os matemáticos apenas a partir do século XVIII. Esta técnica teve em Gauss seu maior divulgador. Ele a publicou em seu livro *Disquisitiones generales circa superficies curvas* (1827). Escrito em latim, este trabalho foi resultado dos estudos realizados por Gauss para diversos governos da região que hoje forma a Alemanha. À época, Gauss tinha sido comissionado para supervisionar uma medição precisa do meridiano entre as cidades de Göttingen e Alton, e confeccionar um mapa geodésico do Ducado de Hannover. Tarefas deste tipo levaram este matemático a fundar os princípios de

um novo campo, a Geodésia avançada, e a conduzir medidas geodésicas de campo que consumiram 15 anos de trabalho (KOLMOGOROV and YUSHKEVICH, 1981). Em seu *Disquisitiones generales*, Gauss definiu o que seriam as representações paramétricas de uma superfície e expressões para o cálculo de distâncias. Foi nesse contexto que ganharam utilidade as descrições abreviadas de transformações lineares, baseadas em uma notação que se assemelha a que seria usada futuramente para matrizes (BASHMAKOVA and SMIRNOVA, 2000).

2.3 Gauss e o Método da Eliminação

Origem bem diferente teve o Método da Eliminação de Gauss. Tal método, descrito com a nomenclatura atual, consiste em uma sequência de operações aplicadas linha a linha de uma matriz, com a intenção de mudar os coeficientes da matriz e reduzi-la a uma matriz triangular superior (em que os coeficientes abaixo da diagonal são nulos). A solução do sistema linear representado por esta matriz é então obtida por substituição reversa, começando pela última equação do sistema reduzido. Encontramos em (ALTHOEN and MCLAUGHLIN, 1987), no entanto, a informação de que o Método da Eliminação de Gauss foi concebido como uma ferramenta para se chegar ao Método dos Mínimos Quadrados, que por sua vez foi concebido para encontrar a melhor função linear que se aproximava de dados coletados em campo. Por causa disso, o Método dos Mínimos Quadrados despertou interesse dos cientistas e engenheiros ligados à Geodésia, e o próprio Método da Eliminação foi lembrado por muito tempo apenas como um instrumento geodésico. O roteiro de uso do Método da Eliminação foi encontrado em um livro de Gauss publicado em 1810, *Disquisitio de Elementis Ellipticis Palladis*. Nesta obra, Gauss queria determinar detalhes sobre a órbita de Pallas, um dos maiores asteroides do Sistema Solar. No caminho, ele se deparou com um sistema de equações lineares em seis incógnitas, em que nem todas as equações poderiam ser satisfeitas ao mesmo tempo. Isso fez com que fosse preciso determinar valores para as incógnitas que minimizassem o erro quadrático total. Em vez de tentar solucionar o problema diretamente, Gauss introduziu um método para lidar com sistemas de equações lineares em geral. Esse método seria aperfeiçoado mais tarde pelo matemático Wilhelm Jordan (1842-1899). Em seu livro *Handbuch der Vermessungskeunde*, publicado em 1888, Jordan apresenta seu método, batizado de GaussJordan, que resulta em uma matriz diagonal equivalente (com coeficientes nulos acima e abaixo da diagonal) e fornece o valor das incógnitas de forma imediata (ALTHOEN and MCLAUGHLIN, 1987).

2.4 Sistemas de equações

Os primeiros exemplos de resoluções de equações e de sistemas lineares de 1º grau são encontrados nas civilizações egípcia e babilônica, por volta de 2000 a.C.

Já em 250 a.C, encontramos uma importante contribuição chinesa ao desenvolvimento da Matemática: a obra Nove capítulos sobre a arte matemática. Nela são apresentados problemas sobre assuntos diversos, entre os quais soluções de alguns sistemas de equações lineares.

Os gregos também desenvolveram métodos de resolução de alguns sistemas de equações de 1º grau, como aqueles encontrados na obra de Diofanto de Alexandria (200-284). Trabalhos dessa natureza são igualmente encontrados nas obras de diversos matemáticos hindus, como Aryabhata (476-550) e Bramagupta (598-670), e árabes, como Al-Khowarismi (século IX) e Ornar Khayyam (1048-1131).

No entanto, somente na segunda metade do século XVII, tem início um tratamento sistematizado da teoria de sistemas de equações lineares, com os estudos de G. W. Leibnitz (1646-1716), na Alemanha, e Takakazu Seki Kowa (1642-1708), no Japão.

No século XVIII, diversos trabalhos sobre sistemas de equações lineares deram corpo a esse estudo, destacando-se aqueles apresentados por Colin Maclaurin (1698-1746), Gabriel Cramer (1704-1752) e Etienne Bézout (1730-1783), que formularam métodos de resolução de sistemas.

Já no século XIX, são significativas para a consolidação da teoria dos sistemas lineares as contribuições de Carl Gustav Jacob Jacobi (1804-1851), Leopold Kronecker (1832-1891), Eugène Rouché (1832-1910) e Ferdinand Georg Frobenius (1847-1917), que tornaram possível uma abordagem mais simplificada desse assunto.

Ao final do século XVIII, Leibnitz, na Alemanha, e Seki Kowa, no Japão, desenvolveram métodos de resolução de sistemas lineares baseados em tabelas numéricas formadas pelos coeficientes das equações que compunham esses sistemas. Essas tabelas numéricas deram origem ao que hoje denominamos matrizes que, além de serem aplicadas ao estudo dos sistemas lineares, possibilitaram o desenvolvimento de novos ramos da Matemática.

Durante o século XVIII, diversos matemáticos desenvolveram estudos sobre algumas propriedades dessas tabelas. Entre os vários trabalhos desenvolvidos, destacam-se os de Pierre S. Laplace (1749-1827) e Alexandre Théophile Vandermonde (1735-1796).

(MUIR, 1890) aponta que Gauss foi o primeiro a usar o termo determinante, em um texto de 1801, só que em outro contexto. O termo determinante foi posteriormente resgatado por Cauchy (1789-1857), desta vez com o significado que conhecemos hoje.

No início do século XIX, Augustin Louis Cauchy (1789-1857) apresentou um importante trabalho sobre essas tabelas, a partir do qual outros matemáticos, como Carl G. J. Jacobi (1804-1851), Arthur Cayley (1821-1895), James J. Sylvester (1814-1897) e Francesco Brioschi (1824-1897), desenvolveram e sistematizaram definições, propriedades, operações e aplicações até hoje utilizadas.

2.5 Como surgiram as matrizes

Conforme (DANTE, 2014), a necessidade de escrever mensagens sigilosas é muito antiga. Ao longo da história, reis, rainhas, generais, entre outros buscaram meios eficientes de comunicação entre seus aliados.

Segundo (IEZZI et al., 2010), as matrizes teriam surgido com a escola inglesa Trinity College, em um artigo do matemático Arthur Cayley(1821-1895), datado de 1858. Vale lembrar, no entanto, que, bem antes, no século III a.C., os chineses já desenvolviam um processo de resolução de sistemas lineares em que apareciam implícita a ideia das matrizes.

Cayley criou as matrizes no contexto de estrutura algébrica, sem pensar em suas aplicações práticas que apareceriam posteriormente.

As transformações geométricas no plano (ou transformações 2D - duas dimensões) são muito usadas pela computação gráfica para a construção de figuras e produção de imagens. Tais imagens podem ser percebidas nos efeitos especiais utilizados no cinema, na TV e nos sistemas multimídia em geral, além de servir de ferramenta de auxílio em várias áreas da atividade humana. As três transformações básicas são: translação, rotação e escala.

Capítulo 3

Sistemas de Equações Lineares

Neste capítulo apresentamos os conceitos teóricos e práticos associados ao cálculo de determinantes, matrizes inversas e resolução de equações lineares algébricas pelo método de Gauss e o método de Laplace.

3.1 Matriz

Denomina-se matriz toda tabela retangular de valores dispostos ordenadamente em linhas e colunas. As matrizes são indicadas por letras maiúsculas do alfabeto latino e representadas utilizando-se parênteses ou colchetes.

Um elemento genérico de uma matriz A é simbolizado por a_{ij} , em que i indica a linha e j a coluna a que o elemento pertence. Se a matriz tem m linhas e n colunas, dizemos que ela é de ordem $m \times n$ e é representada assim:

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

Essa matriz $m \times n$ possui $m \cdot n$ elementos. Podemos também expressá-la de forma mais reduzida, por meio de uma lei de formação para seus elementos:

$$A = (a_{ij})_{m \times n}, \quad \text{onde } i \in \{1, 2, 3, \dots, m\}, \text{ e } j \in \{1, 2, 3, \dots, n\}.$$

3.2 Tipos de matrizes

Algumas matrizes recebem nomes especiais em função de suas características:

a) Matriz linha

É toda matriz de ordem $1 \times n$ ($n \in \mathbb{Z}^+$):

$$\left[a_{11} \ a_{12} \ a_{13} \ \dots \ a_{1n} \right]$$

Se A é uma matriz de ordem $m \times n$, denotamos a i -ésima linha de A como A_i , isto é

$$A_i = \left[a_{i1} \ a_{i2} \ a_{i3} \ \dots \ a_{in} \right]$$

b) Matriz coluna

É toda matriz de ordem $m \times 1$ ($m \in \mathbb{Z}^+$):

$$\begin{bmatrix} a_{11} \\ a_{21} \\ a_{31} \\ \vdots \\ a_{m1} \end{bmatrix} .$$

Se A é uma matriz de ordem $m \times n$, denotamos a j -ésima coluna de A como A^j , isto é

$$A^j = \begin{bmatrix} a_{1j} \\ a_{2j} \\ a_{3j} \\ \vdots \\ a_{mj} \end{bmatrix} .$$

c) Matriz quadrada

É toda matriz cujo número de linhas é igual ao número de colunas. Assim, chamamos matriz quadrada de ordem n a toda matriz de ordem $n \times n$.

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

Toda matriz quadrada possui duas diagonais:

i) Diagonal principal: composta por elementos a_{ij} tais que $i = j$, isto é,

$$a_{11}, a_{22}, a_{33}, a_{44}, \dots, a_{nn};$$

ii) Diagonal secundária: em que os elementos a_{ij} são tais que $i + j = n + 1$, isto é,

$$a_{1n}, a_{2(n-1)}, a_{3(n-2)}, a_{4(n-3)}, \dots, a_{n1}.$$

d) Matriz nula

É toda matriz de ordem $m \times n$ cujos elementos são todos nulos. Para indicar uma matriz nula utiliza-se a notação $O_{m \times n}$.

e) Matriz diagonal

É toda matriz quadrada em que os elementos não pertencentes à diagonal principal são todos nulos.

$$A = \begin{bmatrix} a_{11} & 0 & \cdots & 0 \\ 0 & a_{22} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & a_{nn} \end{bmatrix}$$

f) Matriz identidade

É toda matriz diagonal em que os elementos da diagonal principal são iguais a 1. Para indicar uma matriz identidade de ordem n , utilizamos a notação I_n .

$$I_n = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix}$$

3.3 Operações com matrizes

Para que seja possível aplicar o conceito de matrizes em diversas situações de resolução de problemas práticos, é necessário que sejam definidos os seguintes processos de operações com matrizes:

a) Transposição de uma matriz

Dada uma matriz A do tipo $m \times n$, chama-se transposta de A e indica-se por A^t a matriz que se obtém trocando-se ordenadamente as linhas pelas colunas de A . A operação de obtenção de uma matriz transposta de A é denominada transposição da matriz A . Assim, se A é uma matriz de ordem 3×2 , então A^t é uma matriz de ordem 2×3 como mostra o seguinte exemplo,

$$A = \begin{bmatrix} a & d \\ b & e \\ c & f \end{bmatrix} \Rightarrow A^t = \begin{bmatrix} a & b & c \\ d & e & f \end{bmatrix}$$

onde, a primeira linha de A^t corresponde à primeira coluna de A e a segunda linha de A^t corresponde à segunda coluna de A .

b) Igualdade de matrizes

Duas matrizes, A e B , serão iguais se forem da mesma ordem e se os elementos correspondentes forem iguais. Assim, se $A = (a_{ij})$ e $B = (b_{ij})$ são matrizes de ordem $m \times n$, então:

$$A = B \Leftrightarrow (a_{ij} = b_{ij}, \quad \forall 1 \leq i \leq m, \quad \forall 1 \leq j \leq n).$$

c) Adição de matrizes

Dadas duas matrizes de mesmo tipo, $A = (a_{ij})_{m \times n}$ e $B = (b_{ij})_{m \times n}$, denomina-se matriz soma ($A + B$) a matriz obtida adicionando-se os elementos correspondentes de A e B :

$$A + B = (a_{ij} + b_{ij})_{m \times n}.$$

d) Matriz oposta

Seja a matriz $A = (a_{ij})$ de ordem $m \times n$. Denomina-se a matriz oposta de A , à matriz denotada por $-A$, da mesma ordem que A e cuja soma com A resulta na matriz nula. Isto é,

$$A + (-A) = 0 \Leftrightarrow -A = (-a_{ij}).$$

e) Subtração de matrizes

Dadas duas matrizes A e B de ordem $m \times n$, denomina-se matriz diferença $A - B$ à matriz de ordem $m \times n$, obtida subtraindo-se os elementos correspondentes de A e B :

$$A - B = (a_{ij} - b_{ij})_{m \times n}.$$

f) Multiplicação de número real por matriz

Dada uma matriz $A = (a_{ij})_{m \times n}$ e um número real k , denomina-se matriz produto do número real k por A , à matriz denotada por $k \cdot A$ de ordem $m \times n$, obtida multiplicando-se cada um dos elementos de A por k ,

$$k \cdot A = (k \cdot a_{ij})_{m \times n}.$$

g) Multiplicação de matrizes

Dadas as matrizes $A = (a_{ij})_{m \times n}$ e $B = (b_{ij})_{n \times p}$, definimos a matriz produto de A e B , de ordem $m \times p$ e denotada $C = A \cdot B = (c_{ij})_{m \times p}$, onde

$$\begin{aligned} c_{ij} = A_i \cdot B^j &= \sum_{k=1}^n a_{ik} \cdot b_k \\ &= a_{i1}b_{1j} + a_{i2}b_{2j} + \cdots + a_{in}b_{nj}, \quad i = 1, \dots, m, \quad j = 1, \dots, p. \end{aligned}$$

Da definição observe que o produto $C = A \cdot B$ seja possível, é necessário que o número de colunas de A seja igual ao número de linhas de B

$$C_{m \times p} = A_{m \times n} \cdot B_{n \times p}$$

onde, $c_{ij} = A_i \cdot B^j$ é o produto da i -ésima linha de A e a j -ésima coluna de B a qual é definido como a soma dos produtos de cada elemento da linha pelo seu correspondente elemento da coluna.

3.4 Determinantes

Dada uma matriz A , quadrada de ordem n , podemos associar a ela um número real chamado determinante da matriz A , obtido a partir de operações realizadas com seus elementos. Para se indicar o determinante de uma matriz A utiliza-se a abreviatura $\det(A)$ ou $|A|$.

O cálculo do determinante de uma matriz A pode ser feito por uma lei geral que estabelece as operações a serem realizadas com os elementos da matriz. Antes de estudarmos essa lei geral, vamos estabelecer regras práticas para determinantes de matrizes quadradas de ordens 1, 2 e 3.

a) Matriz 1×1

Dada a matriz $A = [a_{11}]$, o seu determinante é o próprio número a_{11} :

$$\det(A) = |a_{11}| = a_{11}$$

b) Matriz 2×2

Dada a matriz $A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}$, o seu determinante é igual ao produto dos elementos da diagonal principal menos o produto dos elementos da diagonal secundária.

$$\det(A) = a_{11}a_{22} - a_{12}a_{21}$$

c) Matriz 3×3

Dada a matriz $A = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$, o seu determinante é igual a:

$$\det(A) = a_{11}a_{22}a_{33} + a_{12}a_{23}a_{31} + a_{13}a_{21}a_{32} - a_{13}a_{22}a_{31} - a_{11}a_{23}a_{32} - a_{12}a_{21}a_{33}$$

Podemos obter esses seis produtos de uma forma prática, conhecida como regra de Sarrus, fazendo o seguinte:

em que D_{ij} é o menor complementar de a_{ij} .

Exemplo: Dada a matriz:

$$A = \begin{bmatrix} 4 & -3 & 9 \\ 2 & 0 & 5 \\ 3 & 5 & 1 \end{bmatrix},$$

os cofatores A_{11} , A_{21} e A_{31} são:

$$A_{11} = (-1)^{1+1} \cdot \begin{vmatrix} 0 & 5 \\ 5 & 1 \end{vmatrix} = 1 \cdot (0 - 25) = -25;$$

$$A_{21} = (-1)^{2+1} \cdot \begin{vmatrix} -3 & 9 \\ 5 & 1 \end{vmatrix} = (-1) \cdot (-3 - 45) = 48;$$

$$A_{31} = (-1)^{3+1} \cdot \begin{vmatrix} -3 & 9 \\ 0 & 5 \end{vmatrix} = 1 \cdot (-15 - 0) = -15$$

A partir do conceito de cofator, podemos enunciar a lei geral para o cálculo do determinante de uma matriz quadrada de ordem $n \geq 2$, conhecida como Teorema de Laplace (Pierre-Simon Laplace, 1749-1827) oferece uma solução prática no cálculo dos determinantes.

- **Teorema de Laplace**

Dada uma matriz A , quadrada de ordem $n \geq 2$, o valor de $\det(A)$ é o número real obtido pela soma dos produtos dos elementos de uma fila qualquer (linha ou coluna) por seus respectivos cofatores. Assim, se considerarmos os elementos de uma linha i , podemos escrever:

$$\det(A) = a_{i1}A_{i1} + a_{i2}A_{i2} + \cdots + a_{in}A_{in}$$

Exemplo: Dada a matriz

$$A = \begin{bmatrix} 4 & -3 & 9 \\ 2 & 0 & 5 \\ 3 & 5 & 1 \end{bmatrix},$$

calculamos seu determinante usando o Teorema de Laplace.

Escolhendo a 3ª coluna como referência, temos,

$$\begin{aligned} \det(A) &= a_{13}A_{13} + a_{23}A_{23} + a_{33}A_{33} \\ &= 9 \cdot (-1)^{1+3} \cdot \begin{vmatrix} 2 & 0 \\ 3 & 5 \end{vmatrix} + 5 \cdot (-1)^{2+3} \cdot \begin{vmatrix} 4 & -3 \\ 3 & 5 \end{vmatrix} + 1 \cdot (-1)^{3+3} \cdot \begin{vmatrix} 4 & -3 \\ 2 & 0 \end{vmatrix} \\ &= 9 \cdot 1 \cdot (10 - 0) + 5 \cdot (-1) \cdot (20 + 9) + 1 \cdot 1 \cdot (0 + 6) \\ &= -49. \end{aligned}$$

3.4.2 Propriedades dos determinantes

Para uma matriz quadrada A , de ordem n , podem ser demonstradas algumas propriedades que simplificam o cálculo de seu determinante.

1. Se A tem uma linha (ou uma coluna) com todos os elementos nulos, seu determinante é nulo.
2. Se uma matriz A tem duas linhas (ou duas colunas) proporcionais, seu determinante é nulo.
3. Se trocarmos entre si a posição de duas linhas (ou duas colunas) de A , obtemos uma matriz A' tal que:

$$\det A' = -\det A$$

4. Se multiplicarmos os elementos de uma fila (ou de uma coluna) de A por um número real k , obtemos uma matriz cujo determinante é $k \cdot \det(A)$. Isto é,

$$\det \begin{bmatrix} A_1 \\ \vdots \\ A_{i-1} \\ k \cdot A_i \\ A_{i+1} \\ \vdots \\ A_n \end{bmatrix} = k \cdot \begin{bmatrix} A_1 \\ \vdots \\ A_{i-1} \\ A_i \\ A_{i+1} \\ \vdots \\ A_n \end{bmatrix}$$

5. De uma forma geral, podemos dizer que, para uma matriz A , quadrada de ordem n , temos:

$$\det(k \cdot A) = k^n \cdot \det(A)$$

6. O determinante de uma matriz é igual ao determinante de sua transposta,

$$\det(A) = \det(A^t)$$

7. O determinante do produto de duas matrizes quadradas de mesma ordem é igual ao produto dos determinantes dessas matrizes,

$$\det(A \cdot B) = \det(A) \cdot \det(B)$$

8. Se somarmos a uma fila de A uma outra fila previamente multiplicada por um número real, obtemos uma matriz com determinante $\det(A)$,

$$\det \begin{bmatrix} A_1 \\ \vdots \\ A_i + k \cdot A_j \\ \vdots \\ A_j \\ \vdots \\ A_n \end{bmatrix} = \det \begin{bmatrix} A_1 \\ \vdots \\ A_i \\ \vdots \\ A_j \\ \vdots \\ A_n \end{bmatrix}$$

Quando fazemos uma transformação dessa natureza numa matriz, dizemos que estamos fazendo uma combinação linear. Esta propriedade nos garante, portanto, que, ao fazermos combinações lineares entre filas de uma matriz, o valor de seu determinante não se altera.

3.5 Equação Linear

É toda equação do tipo

$$a_1x_1 + a_2x_2 + \dots + a_nx_n = k$$

em que:

- x_1, x_2, \dots, x_n são as incógnitas da equação;
- a_1, a_2, \dots, a_n são números reais denominados coeficientes da equação;
- k é um número real chamado termo independente da equação.

Sempre que o termo independente de uma equação linear é zero ($k = 0$), ela é chamada equação linear homogênea. Uma sequência $(0, 0, \dots, 0, 0)$ será sempre uma das soluções para uma equação linear homogênea. Essa solução é denominada solução trivial.

Toda sequência ordenada de números reais que verifica uma equação linear é chamada solução da equação. Assim, uma sequência $(\alpha_1, \alpha_2, \dots, \alpha_n)$ será solução da equação linear $a_1x_1 + a_2x_2 + \dots + a_nx_n = k$ se ocorrer

$$a_1\alpha_1 + a_2\alpha_2 + \dots + a_n\alpha_n = k.$$

3.6 Sistema de equações lineares

Um sistema de equações lineares $m \times n$ é um conjunto de m equações lineares, cada uma delas com n incógnitas:

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

em que:

- x_1, x_2, \dots, x_n são incógnitas;
- a_{ij} , $i = 1, \dots, m$, $j = 1, \dots, n$ são coeficientes numéricos;
- b_1, b_2, \dots, b_m são termos independentes.

Todo sistema linear do tipo:

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n = 0 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n = 0 \\ \vdots \\ a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n = 0 \end{cases}$$

é denominado sistema linear homogêneo do tipo $m \times n$.

Denominamos solução do sistema linear toda sequência ordenada de números reais que verifica, simultaneamente, todas as equações do sistema. Dessa forma, resolver um sistema significa encontrar todas as sequências ordenadas de números reais que satisfaçam as equações do sistema.

Um sistema linear pode ser classificado de acordo com o número de soluções que admitir. Considerando esse critério, existem três classes de sistemas lineares:

1. Os possíveis e determinados, para os quais existe uma única solução;
2. Os possíveis e indeterminados, que têm infinitas soluções;
3. Os sistemas impossíveis, para os quais não existe solução.

A seguir resolvemos sistemas lineares pela Regra de Cramer e por escalonamento.

3.8 Método de Gauss

A teoria das equações lineares desempenha papel importante e motivador no campo da Álgebra Linear, onde muitos problemas são equivalentes ao estudo de um sistema de equações lineares.

Considere o sistema linear $Ax = b$:

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

O método de eliminação de Gauss consiste em transformar convenientemente o sistema linear original para obter um sistema linear equivalente com a matriz dos coeficientes sendo triangular superior.

Para modificar convenientemente um sistema linear num equivalente, fazemos uso das seguintes operações elementares que não modificam a solução do sistema original:

- Trocar duas equações ou duas colunas;
- Multiplicar uma equação por uma constante não-nula;
- Adicionar um múltiplo de uma equação a outra equação.

Para triangularizar o sistema de equações lineares $Ax = b$ pelo método de Gauss seguimos o seguinte procedimento:

1. Eliminação da primeira coluna

Supondo $a_{11} \neq 0$ (pivô), eliminamos a incógnita x_1 das últimas $n - 1$ equações fazendo:

$$L_i \leftarrow L_i - \frac{a_{i1}}{a_{11}} \cdot L_1, \quad i = 2, \dots, n$$

onde L_i é a i -ésima linha do sistema de equações. Em termos dos coeficientes para cada $i = 2, \dots, n$ temos,

$$\begin{aligned} a_{ij}^2 &= a_{ij} - \frac{a_{i1}}{a_{11}} \cdot a_{1j}, & j = 2, \dots, n \\ b_i^2 &= b_i - \frac{a_{i1}}{a_{11}} \cdot b_1. \end{aligned}$$

onde o super índice 2 indica o valor atualizado de a_{ij} e b_i neste passo.

No final deste estágio, obtém-se a seguinte configuração para o sistema de equações:

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

2. Eliminação da segunda coluna

Supondo $a_{22}^2 \neq 0$ (pivô) eliminamos a incógnita x_2 das últimas $n - 2$ equações fazendo:

$$L_i \leftarrow L_i - \frac{a_{i2}^2}{a_{22}^2} \cdot L_1, \quad i = 3, \dots, n$$

Em termos dos coeficientes para cada $i = 3, \dots, n$ temos,

$$\begin{aligned} a_{ij}^3 &= a_{ij}^2 - \frac{a_{i2}^2}{a_{22}^2} \cdot a_{2j}^2, & j = 3, \dots, n \\ b_i^3 &= b_i^2 - \frac{a_{i2}^2}{a_{22}^2} \cdot b_2^2. \end{aligned}$$

onde o super índice 3 indica o valor atualizado de a_{ij} e b_i neste passo.

No final deste estágio, obtém-se a seguinte configuração para o sistema de equações:

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + \cdots + a_{1n}x_n = b_1 \\ a_{22}^2x_2 + a_{23}^2x_3 + \cdots + a_{2n}^2x_n = b_2^2 \\ a_{33}^3x_3 + \cdots + a_{3n}^3x_n = b_3^3 \\ \vdots \\ a_{n3}^n x_n + \cdots + a_{nn}^n x_n = b_n^n \end{cases}$$

3. Eliminação da k-ésima coluna

Supondo $a_{kk}^k \neq 0$ (pivô) eliminamos a incógnita x_k das últimas $n - k$ equações fazendo:

$$L_i \leftarrow L_i - \frac{a_{ik}^k}{a_{kk}^k} \cdot L_k, \quad i = k + 1, \dots, n$$

Em termos dos coeficientes para cada $i = k + 1, \dots, n$ temos,

$$\begin{aligned} a_{ij}^{k+1} &= a_{ij}^k - \frac{a_{ik}^k}{a_{kk}^k} \cdot a_{kj}^k, & j = k + 1, \dots, n \\ b_i^{k+1} &= b_i^k - \frac{a_{ik}^k}{a_{kk}^k} \cdot b_k^k. \end{aligned}$$

onde o super índice $k + 1$ indica o valor atualizado de a_{ij} e b_i neste passo.

No final deste estágio, obtém-se a seguinte configuração para o sistema de equações:

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + \cdots + a_{1n}x_n = b_1 \\ a_{22}^2x_2 + a_{23}^2x_3 + \cdots + a_{2n}^2x_n = b_2^2 \\ a_{33}^3x_3 + \cdots + a_{3n}^3x_n = b_3^3 \\ \vdots \\ a_{nn}^n x_n = b_n^n \end{cases}$$

No caso de um pivô ser nulo, deve-se obter um pivô não nulo intercambiando a posição com uma das linhas inferiores.

4. Resolvemos o sistema triangular superior por retro-substituição:

$$\begin{aligned} x_n &= \frac{b_n^n}{a_{nn}^n} \\ x_{n-1} &= \frac{b_{n-1}^{n-1} - a_{n-1,n}^{n-1}}{a_{n-1,n-1}^{n-1}} \\ &\vdots \\ x_1 &= \frac{b_1 - a_{1n}x_n - \cdots - a_{13}x_3 - a_{12}x_2}{a_{11}}. \end{aligned}$$

Exemplo: Considere o sistema de equações

$$\begin{cases} 4x_1 + 9x_2 + 10x_3 = 0 \\ 2x_1 + 5x_2 + 5x_3 = 6 \\ 7x_1 + 10x_2 = 1 \end{cases}$$

determinamos os valores de x_1 , x_2 e x_3 seguindo o método de Gauss:

1. Eliminação da primeira coluna

Considerando o pivô $a_{11} = 4$, fazemos as seguintes operações elementares,

$$\begin{aligned} L_2 &\leftarrow L_2 - \frac{a_{21}}{a_{11}}L_1 \\ &\leftarrow (2x_1 + 5x_2 + 5x_3 = 6) - \frac{2}{4}(4x_1 + 9x_2 + 10x_3 = 0) \\ &\leftarrow (0x_1 + \frac{1}{2}x_2 + 0x_3 = 6) \\ \\ L_3 &\leftarrow L_3 - \frac{a_{31}}{a_{11}}L_1 \\ &\leftarrow (7x_1 + 10x_2 + 0x_3 = 1) - \frac{7}{4}(4x_1 + 9x_2 + 10x_3 = 0) \\ &\leftarrow (0x_1 - \frac{23}{4}x_2 - \frac{35}{2}x_3 = 1) \end{aligned}$$

obtem-se o novo sistema equivalente:

$$\begin{cases} 4x_1 + 9x_2 + 10x_3 = 0 \\ \frac{1}{2}x_2 + 0x_3 = 6 \\ -\frac{23}{4}x_2 - \frac{35}{2}x_3 = 1 \end{cases}$$

2. Eliminação da segunda coluna

Considerando o pivô $a_{22} = \frac{1}{2}$, fazemos as seguintes operações elementares:

$$\begin{aligned} L_3 &\leftarrow L_3 - \frac{a_{32}}{a_{22}}L_2 \\ &\leftarrow (0x_1 - \frac{23}{4}x_2 - \frac{35}{2}x_3 = 1) - \frac{\frac{23}{4}}{\frac{1}{2}}(0x_1 + \frac{1}{2}x_2 + 0x_3 = 6) \\ &\leftarrow (0x_1 + 0x_2 - \frac{35}{2}x_3 = 70) \end{aligned}$$

Obtemos agora um sistema linear triangular superior:

$$\begin{cases} 4x_1 + 9x_2 + 10x_3 = 0 \\ \frac{1}{2}x_2 = 6 \\ -\frac{35}{4}x_3 = 70 \end{cases}$$

3. Solução por retro-substituição

Agora fica fácil a resolução. Vejam que a equação L_3 já nos fornece diretamente o valor da incógnita x_3 :

$$-\frac{35}{2}x_3 = 70 \implies x_3 = -4$$

Substituímos x_3 na equação L_2 obtendo:

$$\frac{1}{2}x_2 = 6 \implies x_2 = 12$$

Substituímos x_2 e x_3 na equação L_1 , obtendo:

$$4x_1 + 9x_2 + 10x_3 = 0 \implies x_1 = -17$$

O sistema de equações lineares tem solução única e é dado por $(-17, 12, -4)$.

3.9 Matriz inversa

Para (OLIVEIRA, 2015), encontrar a matriz inversa de uma matriz conhecida é um processo que envolve multiplicação e igualdade de matrizes. Vejamos como ocorre este processo partindo da definição de uma matriz inversa.

Seja A uma matriz quadrada de ordem n , e X uma matriz tal que $A \cdot X = I_n$ e $X \cdot A = I_n$ (onde I_n é a matriz identidade). Caso isso ocorra, denominamos a matriz X de matriz inversa

de A , tendo como notação A^{-1} .

Portanto, para encontrar a inversa de uma matriz dada, deveremos resolver a igualdade de matrizes ($A \cdot X = I_n$). No caso em que sejam dadas duas matrizes e que seja pedido para verificar se uma matriz é a inversa da outra, basta efetuar a multiplicação destas duas matrizes. Se o resultado desta operação for a matriz identidade, afirmaremos que se trata de uma matriz inversa.

Para aqueles que já sabem calcular o determinante, existe um modo prático para descobrir se uma matriz possui uma matriz inversa ou não. Basta calcular o determinante da matriz: caso o determinante dê igual a zero, não existe matriz inversa para ela.

A parte principal para matriz inversa é a parte onde se deve encontrá-la tendo como base uma matriz dada. Vejamos como proceder.

Exemplo: Seja a matriz $A = \begin{bmatrix} 2 & 3 \\ 2 & 5 \end{bmatrix}$. Sabemos que a matriz A^{-1} será uma matriz quadrada de mesma ordem,

$$A^{-1} = \begin{bmatrix} a & b \\ c & d \end{bmatrix}.$$

Logo,

$$A \cdot A^{-1} = I_n \Rightarrow \begin{bmatrix} 2 & 3 \\ 2 & 5 \end{bmatrix} \cdot \begin{bmatrix} a & b \\ c & d \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

Para tanto, deveremos compreender o processo de multiplicação de matrizes para realizarmos estes cálculos.

$$\begin{bmatrix} 2a + 3c & 2b + 3d \\ 2a + 5c & 2b + 5d \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

Através da igualdade de matrizes, obteremos 4 igualdades muito importantes para os nossos cálculos. Agrupá-las-emos de forma que as igualdades com mesmas incógnitas fiquem juntas.

$$I: \begin{cases} 2a + 3c = 1 \\ 2a + 5c = 0 \end{cases} \quad II: \begin{cases} 2b + 3d = 0 \\ 2b + 5d = 1 \end{cases}$$

Em situações como estas devemos resolver estes sistemas de equações com duas incógnitas.

Resolvemos o sistema I pelo método da adição,

$$\begin{array}{r} -2a + 3c = 1 \\ 2a + 5c = 0 \\ \hline -2c = 1 \end{array} \Rightarrow c = -\frac{1}{2}.$$

Substituindo o valor de c , obteremos o valor de a ,

$$2a + 3c = 1 \Rightarrow 2a + 3 \cdot \left(-\frac{1}{2}\right) = 1 \Rightarrow a = \frac{5}{4}.$$

Resolvendo o sistema II de forma análoga, obteremos os seguintes valores para as incógnitas:

$$b = -\frac{3}{4}, \quad d = \frac{1}{2}.$$

Como encontramos os valores para os elementos da matriz inversa, vamos esboçá-la:

$$A^{-1} = \begin{bmatrix} \frac{5}{4} & -\frac{3}{4} \\ -\frac{1}{2} & \frac{1}{2} \end{bmatrix}.$$

Neste primeiro momento verificaremos se de fato esta matriz corresponde à matriz inversa:

$$\begin{bmatrix} 2 & 3 \\ 2 & 5 \end{bmatrix} \cdot \begin{bmatrix} \frac{5}{4} & -\frac{3}{4} \\ -\frac{1}{2} & \frac{1}{2} \end{bmatrix} = \begin{bmatrix} 2 \cdot \frac{5}{4} + 3 \cdot (-\frac{1}{2}) & 2 \cdot (-\frac{3}{4}) + 3 \cdot (\frac{1}{2}) \\ 2 \cdot \frac{5}{4} + 5 \cdot (-\frac{1}{2}) & 2 \cdot (-\frac{3}{4}) + 5 \cdot (\frac{1}{2}) \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}.$$

De fato, a matriz obtida corresponde à matriz inversa, pois o produto das duas matrizes resultou na matriz identidade. Como vimos, o estudo da matriz inversa abarca diversos conceitos da matemática, desde operações básicas até a resolução de sistemas com duas incógnitas.

Capítulo 4

Aspectos numéricos

Neste capítulo apresentamos aspectos numéricos associados à implementação dos algoritmos de resolução linear e determinantes, assim como a aproximação de um número real por uma fração com o objetivo de fornecer uma melhor visualização dos resultados ao usuário.

4.1 Complexidade Computacional

A complexidade computacional ou custo computacional de um algoritmo é o número de operações necessárias para concluir o algoritmo. Neste trabalho vamos contar quantas operações fundamentais são necessárias (adição, multiplicação, subtração e divisão). Sendo que o tempo de execução do compilador para calcular uma multiplicação ou uma divisão é muito maior que o de calcular uma adição ou subtração, consideramos portanto somente na contagem o número de multiplicações e divisões.

Assim o custo computacional de um algoritmo será medido como

$$\text{Custo Computacional} = \text{número de multiplicações} + \text{número de divisões}$$

onde, uma multiplicação e uma divisão tem o mesmo custo computacional.

4.1.1 Sistema diagonal

Seja o sistema diagonal

$$\left\{ \begin{array}{l} a_{11}x_1 \\ a_{22}x_2 \\ \vdots \\ a_{nn}x_n \end{array} \right. = \begin{array}{l} b_1 \\ b_2 \\ \vdots \\ b_n \end{array}$$

onde, $a_{ii} \neq 0, \quad \forall i = 1, \dots, n.$

O algoritmo para resolver o sistema diagonal é:

Entrada: $(a_{ij}), i = 1 : n, j = 1 : n$

Entrada: $(b_i), i = 1 : n$

início

para $k = 1$ **até** $k = n - 1$ **faça**

achar $i \geq k$ **tal que** $a_{ik} \neq 0$

troque *linha* k **com** *linha* i

se $a_{ii} = 0, \forall i \geq k$ **então**

| **STOP** \triangleright A não é inversível

fim

para $i = k + 1$ **até** $i = n$ **faça**

$m \leftarrow m_{ik} = \frac{a_{ik}}{a_{kk}} \triangleright$ Multiplicador

$b_i \leftarrow b_i - m \cdot b_k$

para $j = k + 1$ **até** $j = n$ **faça**

| $a_{ij} \leftarrow a_{ij} - m \cdot a_{kj}$

fim

fim

fim

fim

Temos:

Divisões:

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

Multiplicações:

$$\sum_{k=1}^{n-1} \sum_{i=k+1}^n 1 + \sum_{k=1}^{n-1} \sum_{i=k+1}^n \sum_{j=k+1}^n 1 =$$

$$\begin{aligned}
&= \frac{n(n-1)}{2} + \sum_{k=1}^{n-1} \sum_{i=k+1}^n [n - (k+1) + 1] \\
&= \frac{n(n-1)}{2} + \sum_{k=1}^{n-1} \sum_{i=k+1}^n (n-k) \\
&= \frac{n(n-1)}{2} + \sum_{k=1}^{n-1} (n-k) \sum_{i=k+1}^n 1 \\
&= \frac{n(n-1)}{2} + \sum_{k=1}^{n-1} (n-k) [n - (k+1) + 1] \\
&= \frac{n(n-1)}{2} + \sum_{k=1}^{n-1} n^2 - 2nk + k^2 \\
&= \frac{n(n-1)}{2} + n^2 \sum_{k=1}^{n-1} 1 - 2n \sum_{k=1}^{n-1} k + \sum_{k=1}^{n-1} k^2 \\
&= \frac{n(n-1)}{2} + n^2(n-1-1+1) - 2n \frac{n(n-1)}{2} + \frac{(n-1)n[2(n-1)+1]}{6} \\
&= \frac{n(n-1)}{2} + n^2(n-1) - n^2(n-1) + \frac{(n-1)n(2n-1)}{6} \\
&= \frac{n(n-1)}{2} + \frac{(n-1)n(2n-1)}{6}
\end{aligned}$$

Logo, o custo computacional do escalonamento é:

$$\begin{aligned}
\text{Custo Computacional} &= \frac{n(n-1)}{2} + \frac{n(n-1)}{2} + \frac{n(n-1)(2n-1)}{6} \\
&= \frac{n^2}{2} + \frac{n}{2} + \frac{n^2}{2} - \frac{n}{2} + \frac{n(n-1)(2n-1)}{6} \\
&= n^2 + \frac{n(n-1)(2n-1)}{6} \\
&= \frac{3n^3 - 3n^2 + n}{6}
\end{aligned}$$

Como $n^3 > n^2$ e $n^3 > n$ temos que

$$\text{Custo Computacional} \approx n^3$$

Isto é, o custo computacional do algoritmo de eliminação de Gauss, cresce como uma potência de n^3 .

4.1.4 Determinantes por Cofatores

Se $A \in M(n)$ (matrizes quadradas de ordem n), então:

$$\det(A) = \sum_{j=1}^n a_{ij} A_{ij}$$

Para esclarecer o cálculo do custo computacional vamos exemplificar com matrizes de ordem 1, 2, 3 e 4.

1. Se $A \in M(1)$ e $A = (a_{11})$, então $\det(A) = a_{11}$

2. Se $A \in M(2)$ e $A = (a_{ij})$, então $\det(A) = a_{11}a_{22} - a_{12}a_{21}$, portanto

$$\text{Custo Computacional} \approx 2$$

3. Se $A \in M(3)$ e $A = (a_{ij})$, então $\det(A) = a_{11}A_{11} + a_{12}A_{22} + a_{13}A_{33}$, onde A_{ii} são determinantes de ordem 2, logo temos:

$$\text{Custo Computacional} \approx 3 \cdot 2 = 6$$

4. Se $A \in M(4)$ e $A = (a_{ij})$, então $\det(A) = a_{11}A_{11} + a_{12}A_{22} + a_{13}A_{33} + a_{14}A_{44}$, onde A_{ii} são determinantes de ordem 3, logo temos:

$$\text{Custo Computacional} \approx 4 \cdot 6 = 24$$

Em geral, denotemos $D_j \approx$ custo computacional de calcular o determinante de ordem j .

Se $A \in M(n)$, temos

$$\det(A) = \sum_{j=1}^n a_{ij}A_{ij}$$

logo $D_n \approx n \cdot D_{n-1}$ a qual é a equação de recorrência.

Assim

$$D_2 \approx 2$$

$$D_3 \approx 3 \cdot D_2$$

$$D_4 \approx 4 \cdot D_3$$

$$D_5 \approx 5 \cdot D_4$$

$$\vdots \quad \vdots$$

$$D_{n-1} \approx (n-1) \cdot D_{n-2}$$

$$D_n \approx n \cdot D_{n-1}$$

multiplicando, temos

$$D_n \approx n!$$

isto é, o custo de avaliar um determinante de uma matriz de ordem n é aproximadamente $n!$ multiplicações.

Usando a fórmula de Stirling temos:

$$D_n \approx n! \approx \left(\frac{n}{e}\right)^n \sqrt{2\pi n}$$

significando que o custo computacional de avaliar um determinante cresce exponencialmente com a ordem do determinante. Por exemplo, se $n = 20$ temos:

$$D_{20} \approx 20! \approx \left(\frac{20}{e}\right)^{20} \sqrt{2\pi 20}$$

Se um computador tem um processador que executa um giga flop por segundo ($1\text{G flop/s} = 10^9$ flop/s), o tempo para calcular um determinante de uma matriz de ordem 20 é:

$$t_{20} \approx \frac{D_{20}}{10^9} \approx \frac{(7,38)^{20} \times 11,21}{10^9} \text{ segundos}$$

aplicando log, temos

$$\begin{aligned} \log t_{20} &\approx 20 \log 7,38 + \log 11,21 - \log(10^9) \\ &\approx 17,361 + 1,041 - 9 \\ &\approx 9,40 \end{aligned}$$

obtendo

$$\begin{aligned} t_{20} &\approx 10^{9,40} \text{ segundos, ou} \\ t_{20} &\approx 10^{5,84} \text{ horas, ou} \\ t_{20} &\approx 10^{4,46} \text{ dias, ou} \\ t_{20} &\approx 10^{1,82} \text{ anos, ou} \\ t_{20} &\approx 72 \text{ anos} \end{aligned}$$

ou seja, este processador com a capacidade de realizar 10^9 flop/s demoraria 72 anos para calcular um determinante de ordem 20 usando o método dos cofatores.

Assim, o método dos cofatores não é um método eficiente para calcular determinantes de matrizes de ordem maior que 3.

O cálculo ineficiente do determinante pelo método dos cofatores faz ineficiente também o método de Cramer para resolver sistemas lineares, pois temos que calcular $(n + 1)$ determinantes, com um custo de :

$$\text{Custo Computacional} = (n + 1) \cdot D_n \approx (n + 1) \left(\frac{n}{e}\right)^n \sqrt{2\pi n},$$

$(n + 1)$ vezes o cálculo de um determinante pelo método dos cofatores.

o custo computacional de ordem polinomial n^3 é muito mais eficiente e prático que o método de Cramer resolvendo os determinantes por cofatores com custo computacional exponencial de $(n + 1) \left(\frac{n}{e}\right)^n \sqrt{2\pi n}$

4.1.5 Matriz Inversa

Seja a matriz $A \in M(n)$ e sua inversa $A^{-1} \in M(n)$, onde $A = (a_{ij})$ é conhecida e $A^{-1} = (b_{ij})$ é a matriz desconhecida. Verificando que:

$$A \cdot A^{-1} = I$$

Para resolver a j -ésima coluna de A^{-1} , formamos o sistema linear de n equações:

$$\sum_{k=1}^n a_{ik} b_{kj} = \delta_{ij}, \quad i = 1, 2, \dots, n$$

onde,

$$\delta = \begin{cases} 1 & \text{se } i = j \\ 0 & \text{se } i \neq j \end{cases}$$

Como temos n colunas, então resolver a matriz inversa por este método requer resolver independentemente n sistemas de equações de ordem n . Logo o custo computacional de calcular a inversa de uma matriz de ordem n é:

$$\text{Custo Computacional} \approx n \cdot n^3 = n^4,$$

Assim, o cálculo da inversa de uma matriz de ordem n tem um custo computacional n^4 .

4.2 Transformação de decimais em frações

Nesta seção apresentamos como escrever um número como uma fração com o intuito de que o resultado apresentado ao usuário seja uma fração. Mesmo que o resultado seja um número irracional, este vai ser truncado com um número determinado de casas decimais representado assim uma aproximação do número obtido.

De acordo com (MIRANDA, 2015), temos as seguintes transformações:

4.2.1 Número inteiro em fração

Se pegarmos o número 2 para representá-lo em forma de fração basta achar um número que dividido por outro número o resultado seja 2. Por exemplo, as frações $\frac{10}{5}$ ou $\frac{20}{10}$ ou $\frac{200}{100}$ representam o inteiro 2.

4.2.2 Número decimal exato em fração

Se pegarmos o número 0,5 (a leitura dele é cinco décimos), é preciso lembrar que décimo vem de dez, assim como centésimos vem de cem e milésimo vem de mil, então para transformar 0,5 em fração basta eliminar a vírgula ficando o número 5, assim o denominador será o número que representa a casa decimal, então:

$$0,5 = \frac{5}{10} = \frac{1}{2}$$

Para transformar 2,25 (sua leitura é dois inteiros e vinte e cinco centésimos), retirando a vírgula fica 225 no numerador, o denominador fica 100, pois as casas decimais estão em centésimos.

$$2,25 = \frac{225}{100} = \frac{45}{20} = \frac{9}{4}$$

Se dividirmos o numerador de cada fração acima pelo denominador correspondente, chegaremos ao valor decimal correspondente a ele.

4.2.3 Dízima periódica em fração

Dízima periódica é a parte decimal infinita, pois repete igualmente. Por exemplo: 0,6666...; 0,4545...; 5,6333... .

Esses números podem ser escritos em forma de fração, mas apesar de serem números decimais na sua transformação utilizaremos um processo diferente.

Exemplo 1:

Vamos transformar 0,6666... em fração, chamaremos a dízima de X :

$$X = 0,6666... \quad (4.1)$$

Devemos eliminar as casas decimais. Para isso andaremos com a vírgula para a direita uma casa decimal, pois apenas o 6 que repete. Isso é o mesmo que multiplicar o 0,6666... por 10.

$$10.X = 6,6666... \quad (4.2)$$

Iremos subtrair as duas equações: (4.2) – (4.1)

$$\begin{array}{r} 10X = 6,6666... \\ - \quad X = 0,6666... \\ \hline 9X = 6 \end{array}$$

$$X = \frac{6}{9} = \frac{2}{3}$$

Exemplo 2:

Temos a dízima 0,4545...

$$X = 0,4545... \quad (4.3)$$

andando com a vírgula duas casas para a direita, pois o número que repete nas casas decimais é o 45, andar duas casas para a direita é o mesmo que multiplicar por 100.

$$100 \cdot X = 45,4545... \quad (4.4)$$

Iremos subtrair as duas equações: (4.4) – (4.3)

$$\begin{array}{r} 100X = 45,4545... \\ - \quad X = 0,4545... \\ \hline 99X = 45 \end{array}$$

$$X = \frac{45}{99} = \frac{5}{11}$$

Exemplo 3:

Temos a dízima 5,6333... nessa percebemos que na parte decimal temos apenas o 3.

$$X = 5,6333...$$

Como o 6 não faz parte da dízima devemos multiplicar a equação por 10 para que o número 6 passe pro outro lado deixando nas casas decimais apenas a dízima.

$$10 \cdot X = 56,3333... \quad (4.5)$$

Agora, multiplicamos a equação (4.5) por 10 novamente para que possamos cancelar a parte decimal.

$$100 \cdot X = 563,3333... \quad (4.6)$$

Subtraindo as equações (4.6) – (4.5), teremos:

$$\begin{array}{r} 100X = 563,3333... \\ - \quad 10X = 56,3333... \\ \hline 90X = 507 \end{array}$$

$$X = \frac{507}{90} = \frac{169}{30}$$

4.3 Uma abordagem computacional para a representação de uma fração

Para uso no cotidiano os exemplos acima são bem práticos, porém para programação não é tão simples assim, apesar de ser aplicável, não é viável, pois o computador utiliza muito mais recursos para multiplicar e dividir, também teria uma grande dificuldade para simplificar frações, pensando nisso surgiu a ideia de utilizarmos o que a máquina faz muito bem, que é somar e contar, para fazer a transformação de números decimais em números fracionários.

Todo número decimal é uma fração de um número inteiro, portanto se somarmos várias vezes essa fração chegaremos a um número inteiro que será o numerador da fração, o denominador será determinado pela contagem das vezes que utilizamos o decimal na soma, baseado nisso, desenvolvemos um algoritmo que transforma os decimais em frações para serem apresentados nas matrizes.

Exemplo 1:

Vamos transformar 0,75 em fração.

$$\begin{array}{r}
 + 0,75 \Rightarrow 1 \\
 + 0,75 \Rightarrow 2 \\
 \hline
 + 1,50 \\
 + 0,75 \Rightarrow 3 \\
 \hline
 + 2,25 \\
 + 0,75 \Rightarrow 4 \\
 \hline
 3,00
 \end{array}$$

Portanto $0,75 = \frac{3}{4}$.

Exemplo 2:

Vamos transformar 7,625 em fração.

$$\begin{array}{r}
 + 7,625 \Rightarrow 1 \\
 + 7,625 \Rightarrow 2 \\
 \hline
 + 15,250 \\
 + 7,625 \Rightarrow 3 \\
 \hline
 + 22,875 \\
 + 7,625 \Rightarrow 4 \\
 \hline
 + 30,500 \\
 + 7,625 \Rightarrow 5 \\
 \hline
 + 38,125 \\
 + 7,625 \Rightarrow 6 \\
 \hline
 + 45,750 \\
 + 7,625 \Rightarrow 7 \\
 \hline
 + 53,375 \\
 + 7,625 \Rightarrow 8 \\
 \hline
 61,000
 \end{array}$$

Portanto $7,625 = \frac{61}{8}$.

Fazendo dessa forma teríamos um problema com números com a parte decimal muito longa, pois a soma poderia demorar muito, para resolver esta situação utilizamos na programação, um tipo de número decimal chamado currency, destinado a valores monetários e que tem precisão até a quarta casa decimal, ou seja quando um número for 0,99995 ele considera 1. Apesar de isso gerar um erro de arredondamento, não interfere no resultado das contas, pois as frações são apenas para melhorar o visual das matrizes e os cálculos são feitos com os números decimais originais.

Capítulo 5

Descrição do Aplicativo

O aplicativo **Matriz** tem como objetivo principal auxiliar o aluno a aprender o método de Gauss para escalonar matrizes, esse método é muito útil nos mais diversos cálculos, como por exemplo: determinante, equações lineares, matriz inversa, etc.

5.1 Tela Inicial

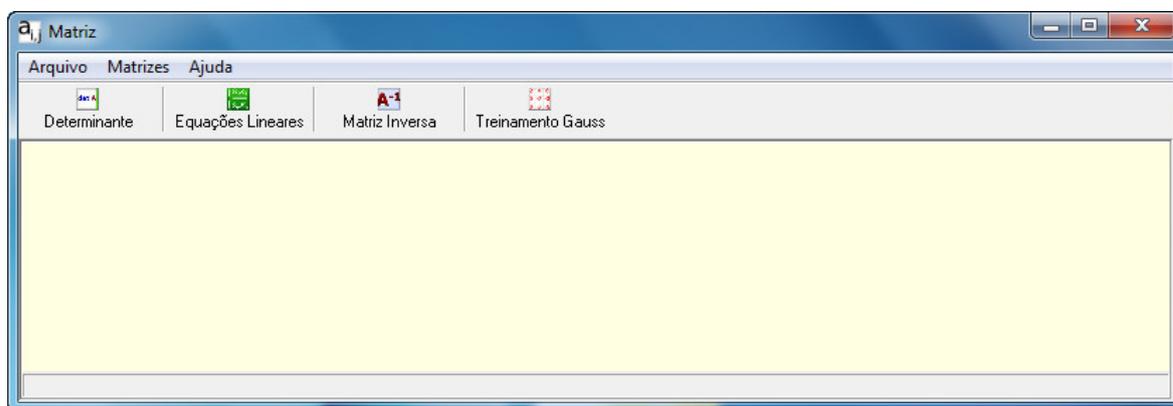


Figura 5.1: Tela Inicial

Na tela mostrada na Figura 5.1 temos os seguintes botões:

Determinante: Calcula o determinante de uma matriz pelo método de Laplace e pelo método de Gauss (escalonamento).

Equações Lineares: Calcula a solução de equações lineares utilizando o método de Gauss.

Matriz Inversa: Calcula matriz inversa utilizando o método de Gauss.

Treinamento Gauss: Treinamento do método de Gauss fazendo conferência pelo determinante da matriz.

5.2 Determinante

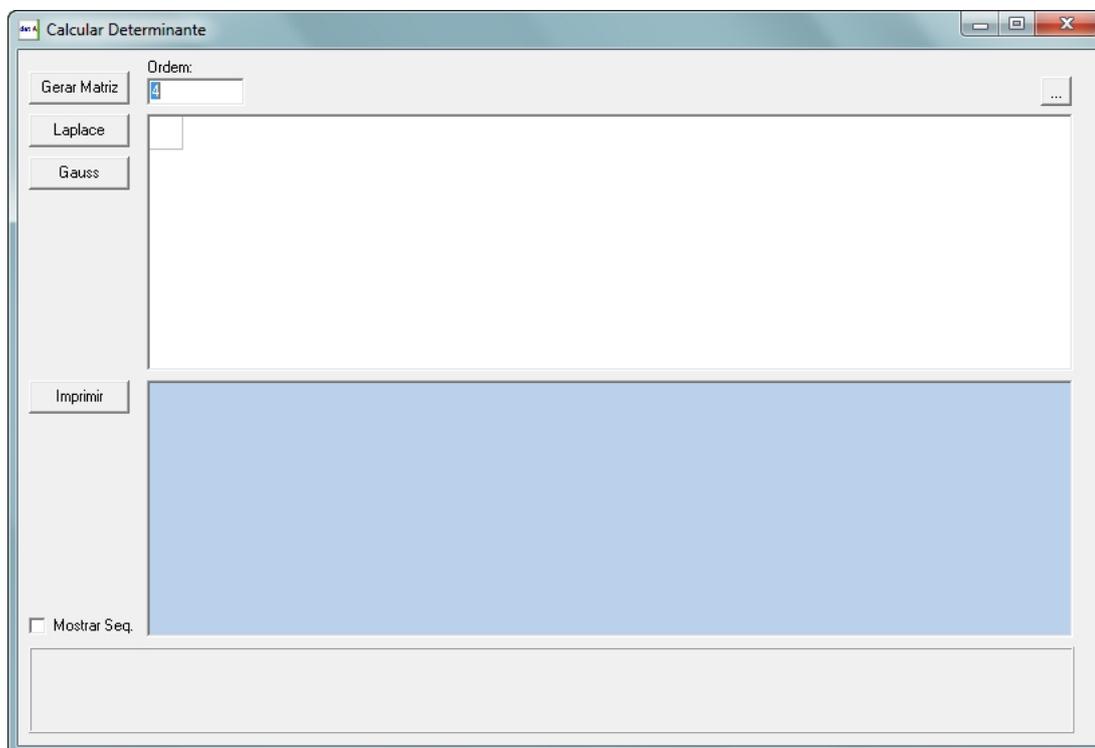


Figura 5.2: Tela Determinante

Na tela mostrada na Figura 5.2 temos as seguintes opções:

Gera Matriz: Gera a matriz de acordo com a ordem definida.

Laplace: Calcula o valor do determinante da matriz pelo método de Laplace.

Gauss: Calcula o valor do determinante da matriz pelo método de Gauss.

Imprimir: Imprimir cálculos apresentados na área azul.

Ordem: Define a ordem da matriz.

Mostra Seq.: Mostra a sequência de cálculo na área azul.

... : Gera valores aleatórios para a matriz de acordo com a ordem.

5.3 Equações Lineares

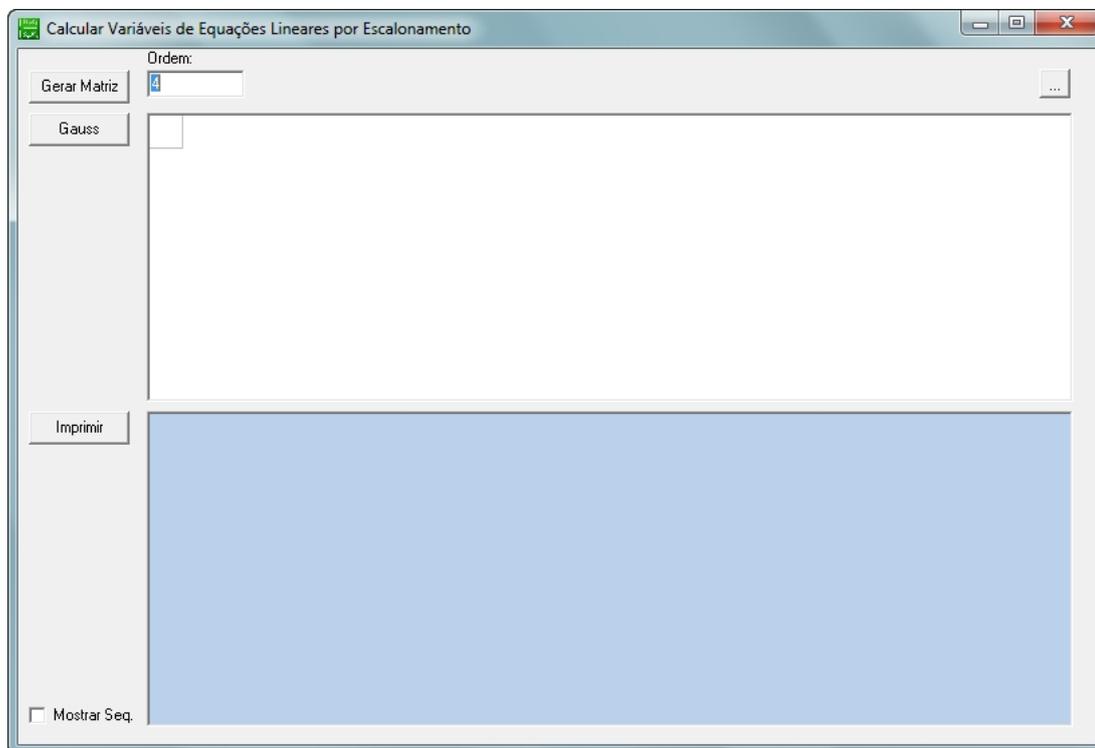


Figura 5.3: Tela Equações Lineares

Na tela mostrada na Figura 5.3 temos as seguintes opções:

Gera Matriz: Gera a matriz de acordo com a ordem definida.

Gauss: Calcula o valor das variáveis da matriz pelo método de Gauss.

Imprimir: Imprimi cálculos apresentados na área azul.

Ordem: Define a ordem da matriz.

Mostra Seq.: Mostra a sequência de cálculo na área azul.

... : Gera valores aleatórios para a matriz de acordo com a ordem.

5.4 Matriz Inversa

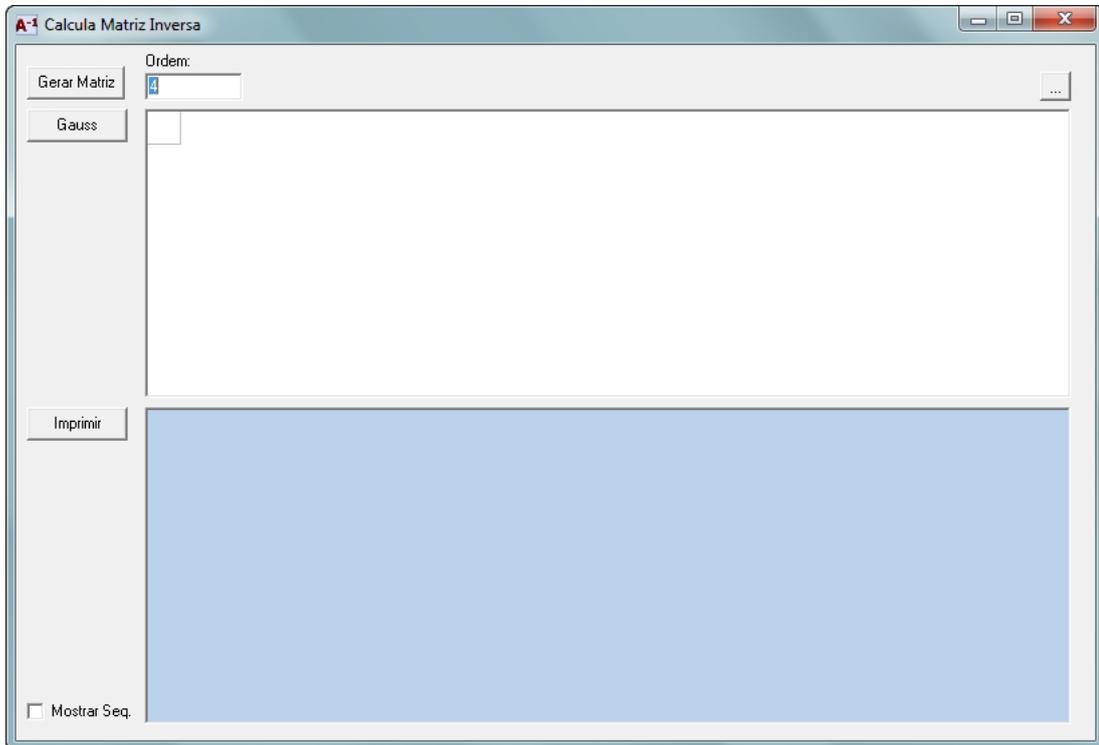


Figura 5.4: Tela Matriz Inversa

Na tela mostrada na Figura 5.4 temos as seguintes opções:

Gera Matriz: Gera a matriz de acordo com a ordem definida.

Laplace: Calcula o valor do determinante da matriz pelo método de Laplace.

Gauss: Calcula a matriz inversa pelo método de Gauss.

Imprimir: Imprimir cálculos apresentados na área azul.

Ordem: Define a ordem da matriz.

Mostra Seq.: Mostra a sequência de cálculo na área azul.

... : Gera valores aleatórios para a matriz de acordo com a ordem.

5.5 Treinamento Gauss

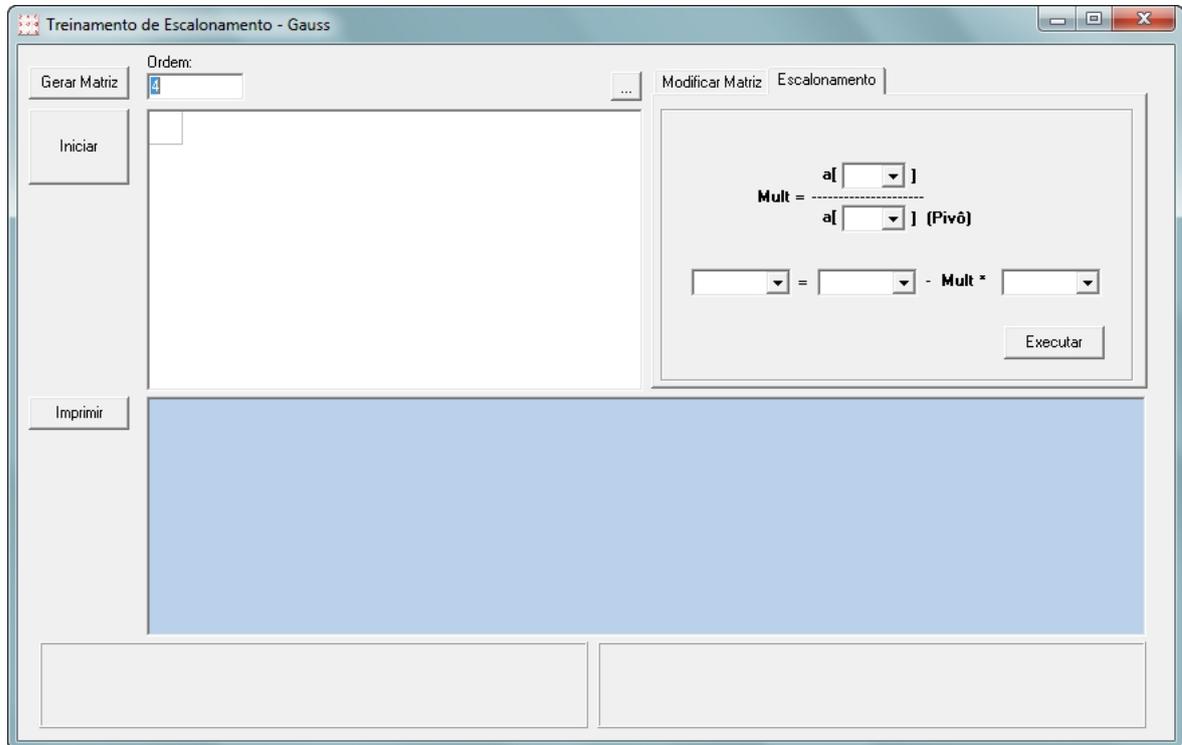


Figura 5.5: Tela Treinamento Gauss

Na tela mostrada na Figura 5.5 temos as seguintes opções:

Gera Matriz: Gera a matriz de acordo com a ordem definida.

Iniciar: Calcula o determinante da matriz e inicia o treinamento de escalonamento pelo método de Gauss.

Executar: Executa operações na última matriz apresentada na área azul.

Imprimir: Imprimi cálculos apresentados na área azul.

Ordem: Define a ordem da matriz.

Mostra Seq.: Mostra a sequência de cálculo na área azul.

... : Gera valores aleatórios para a matriz de acordo com a ordem.

Escalonamento: Operações de divisão de elementos, multiplicação de linha por número e subtração de linhas com o objetivo de escalonar a matriz. (Gauss)

Modificar Matriz: Operações de adição ou subtração de linhas e multiplicação de linha por número com o objetivo de evitar o pivô zero.

5.6 Exemplo de Cálculo de Determinante

Para calcular o determinante da matriz abaixo:

$$A = \begin{bmatrix} -2 & -1 & 4 & 1 \\ 4 & -1 & 5 & -2 \\ -2 & 5 & -1 & 3 \\ 3 & 0 & -3 & -2 \end{bmatrix}$$

1. Primeiro digite a ordem da matriz, nesse exemplo é da quarta ordem e em seguida clique no botão "Gerar Matriz".

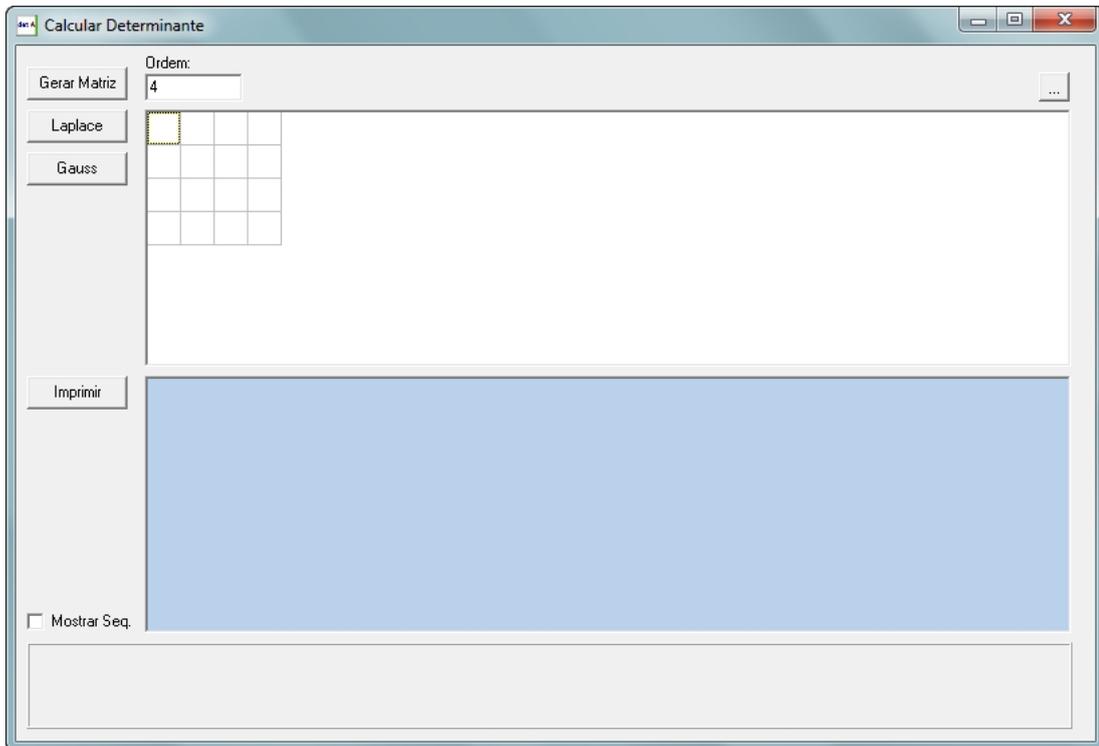


Figura 5.6: Primeiro passo - Determinante

2. Digite os elementos da matriz.

-2	-1	4	1
4	-1	5	-2
-2	5	-1	3
3	0	-3	-2

Figura 5.7: Segundo passo - Determinante

3. Marque "Mostra Seq." para que seja exibida a sequência de cálculo na área em azul, e clique em "Gauss" ou "Laplace" para processar.

Método de Gauss

-2	-1	4	1
4	-1	5	-2
-2	5	-1	3
3	0	-3	-2

$L2 = L2 - (a[2,1]/a[1,1])*L1$

$L3 = L3 - (a[3,1]/a[1,1])*L1$

$L4 = L4 - (a[4,1]/a[1,1])*L1$

-2	-1	4	1
0	-3	13	0
0	6	-5	2
0	-3/2	3	-1/2

$L3 = L3 - (a[3,2]/a[2,2])*L2$

$L4 = L4 - (a[4,2]/a[2,2])*L2$

-2	-1	4	1
0	-3	13	0
0	0	21	2
0	0	-7/2	-1/2

$L4 = L4 - (a[4,3]/a[3,3])*L3$

-2	-1	4	1
0	-3	13	0
0	0	21	2
0	0	0	-1/6

Diagonal Final =

-2	-3	21	-1/6
----	----	----	------

Determinante = -21

Figura 5.8: Terceiro passo - Determinante

4. Para imprimir o cálculo, clique em "Imprimir".

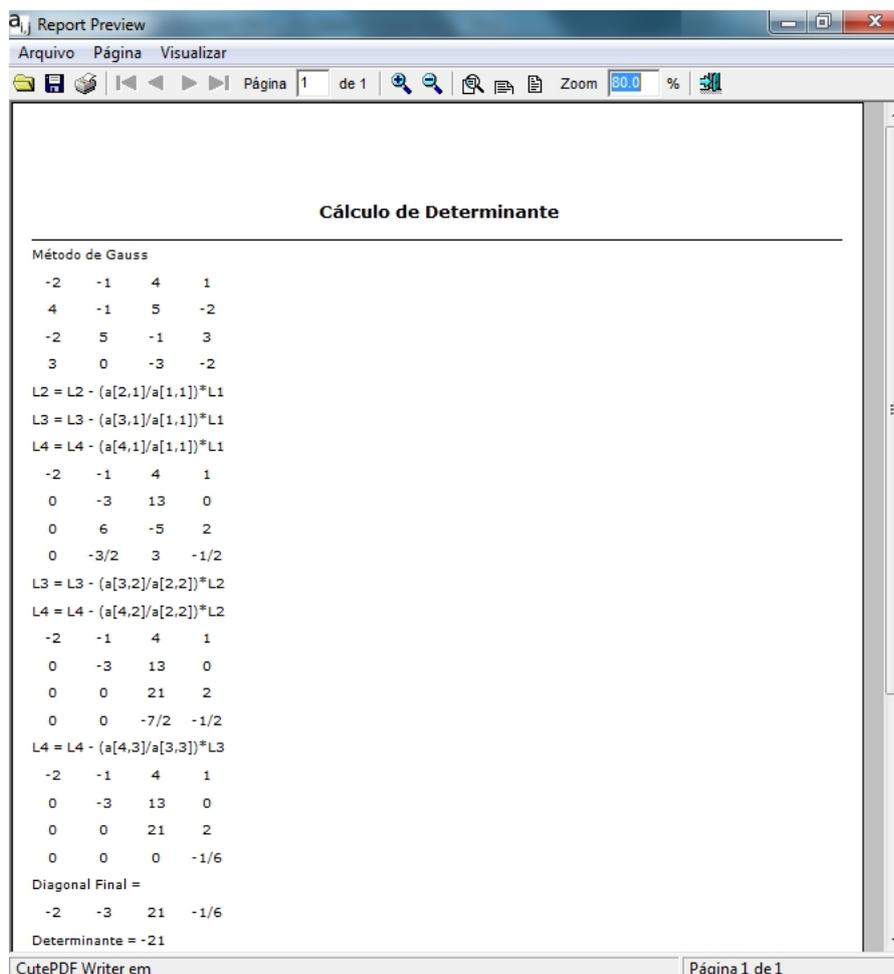


Figura 5.9: Quarto passo - Determinante

5.7 Exemplo de Cálculo de Equações Lineares

Para calcular as variáveis do sistema abaixo:

$$\begin{cases} 2x_1 - 3x_2 + x_3 - 2x_4 = 1 \\ 4x_1 + 2x_2 - x_3 - 3x_4 = -7 \\ -x_1 - 2x_2 + 3x_3 + x_4 = 10 \\ -2x_1 + x_2 - 2x_3 + 2x_4 = -1 \end{cases}$$

1. Primeiro digite a ordem da matriz, nesse exemplo é da quarta ordem e em seguida clique no botão "Gerar Matriz".

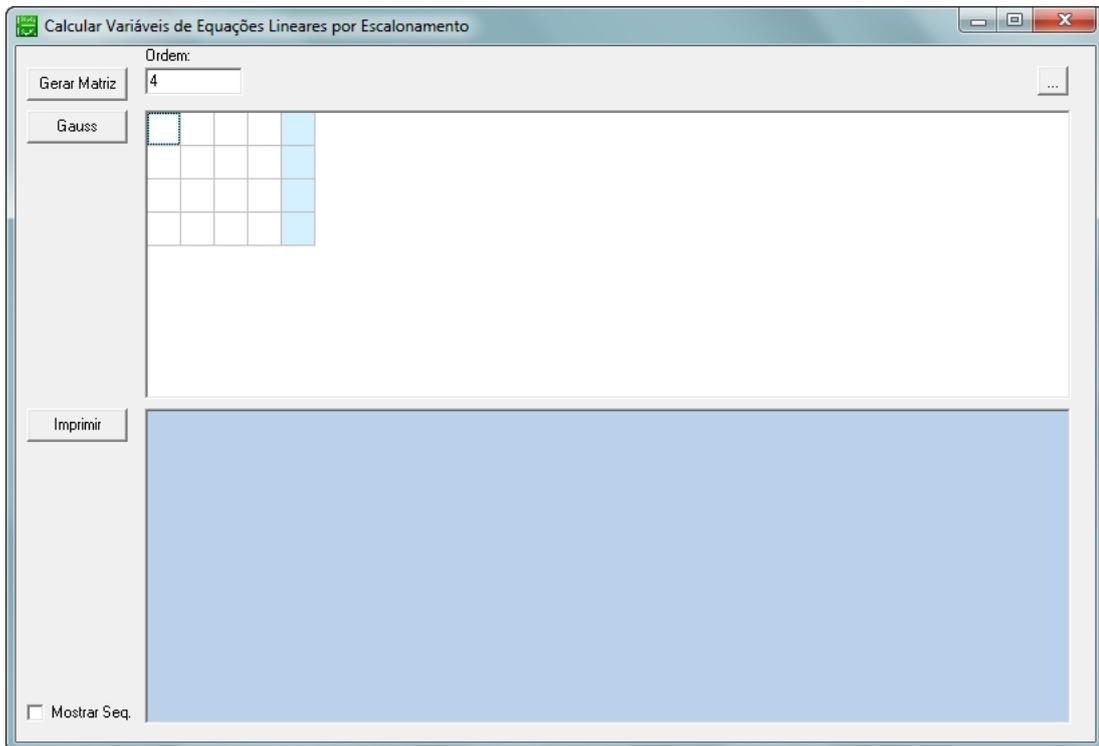


Figura 5.10: Primeiro passo - Equações Lineares

2. Digite os coeficientes das equações, na ordem que aparecem, para montar a matriz.

2	-3	1	-2	1
4	2	-1	-3	-7
-1	-2	3	1	10
-2	1	-2	2	-1

Figura 5.11: Segundo passo - Equações Lineares

3. Marque "Mostra Seq." para que seja exibida a sequência de cálculo na área em azul, e clique em "Gauss" para processar.

2	-3	1	-2	1
4	2	-1	-3	-7
-1	-2	3	1	10
-2	1	-2	2	-1

$L2 = L2 - (a[2.1]/a[1.1])*L1$
 $L3 = L3 - (a[3.1]/a[1.1])*L1$
 $L4 = L4 - (a[4.1]/a[1.1])*L1$

2	-3	1	-2	1
0	8	-3	1	-9
0	-7/2	7/2	0	21/2
0	-2	-1	0	0

$L3 = L3 - (a[3.2]/a[2.2])*L2$
 $L4 = L4 - (a[4.2]/a[2.2])*L2$

2	-3	1	-2	1
0	8	-3	1	-9
0	0	35/16	7/16	105/16
0	0	0	3/5	3

$X[1] = 3$
 $X[2] = -1$
 $X[3] = 2$
 $X[4] = 5$

Figura 5.12: Terceiro passo - Equações Lineares

4. Para imprimir o cálculo, clique em "Imprimir".

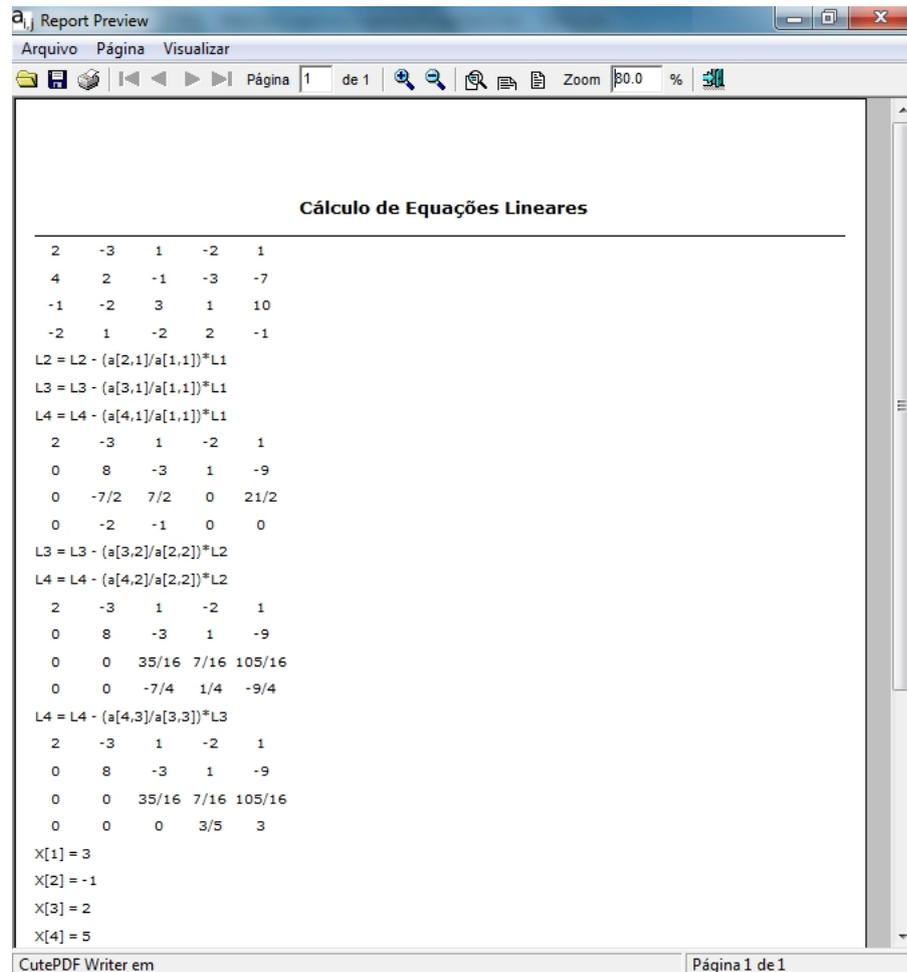


Figura 5.13: Quarto passo - Equações Lineares

5.8 Exemplo de Cálculo de Matriz Inversa

Para calcular a matriz inversa da matriz abaixo:

$$A = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$$

1. Primeiro digite a ordem da matriz, nesse exemplo é da segunda ordem e em seguida clique no botão "Gerar Matriz".

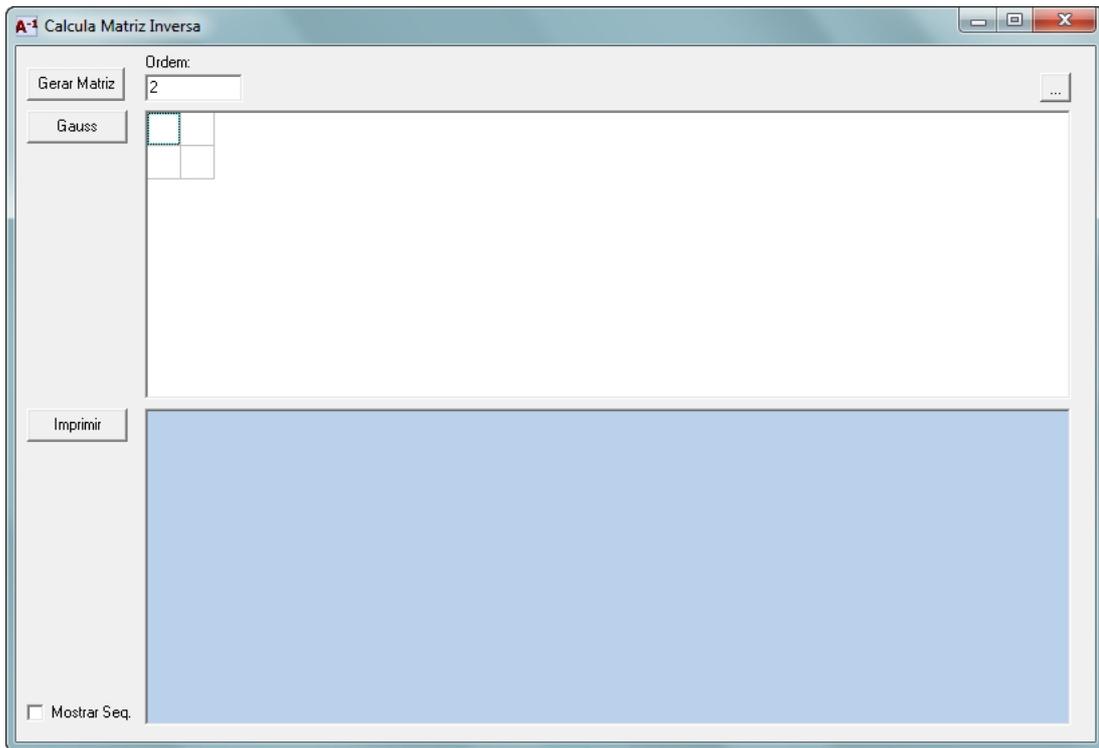


Figura 5.14: Primeiro passo - Matriz Inversa

2. Digite os elementos da matriz.

1	2
3	4

Figura 5.15: Segundo passo - Matriz Inversa

3. Marque "Mostra Seq." para que seja exibida a sequência de cálculo na área em azul, e clique em "Gauss" para processar.

1 2
3 4

1ª Matriz Transposta

1 3 1
2 4 0

$L2 = L2 - (a[2,1]/a[1,1])*L1$

1 3 1
0 -2 -2

X[1] = -2
X[2] = 1

2ª Matriz Transposta

1 3 0
2 4 1

$L2 = L2 - (a[2,1]/a[1,1])*L1$

1 3 0
0 -2 1

X[1] = 3/2
X[2] = -1/2

-2 1
3/2 -1/2

Figura 5.16: Terceiro passo - Matriz Inversa

4. Para imprimir o cálculo, clique em "Imprimir".

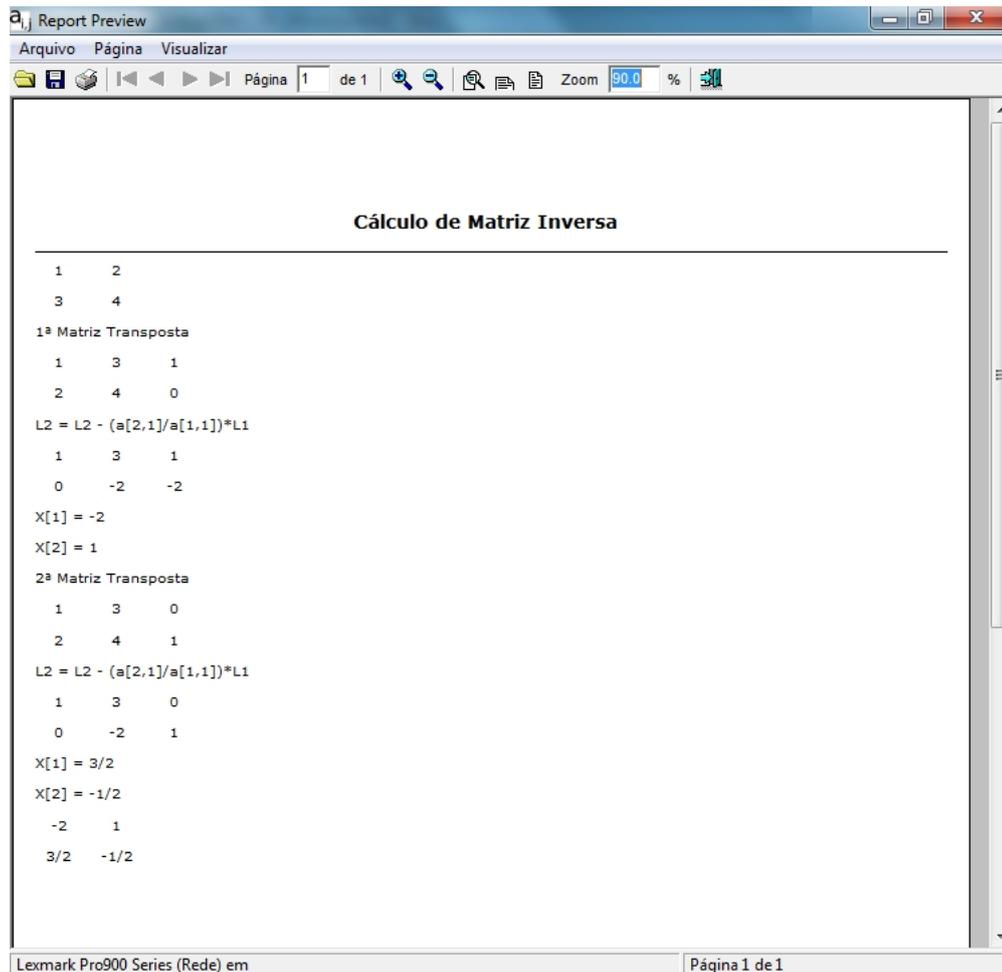


Figura 5.17: Quarto passo - Matriz Inversa

5.9 Exemplo de Treinamento Gauss

Para treinamento vamos utilizar matriz abaixo:

$$A = \begin{bmatrix} 0 & 1 & 2 \\ 4 & 2 & -1 \\ 2 & 5 & 3 \end{bmatrix} \Rightarrow \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$$

1. Primeiro digite a ordem da matriz, nesse exemplo a matriz é de terceira ordem, em seguida clique no botão "Gerar Matriz" então digite os elementos da matriz.

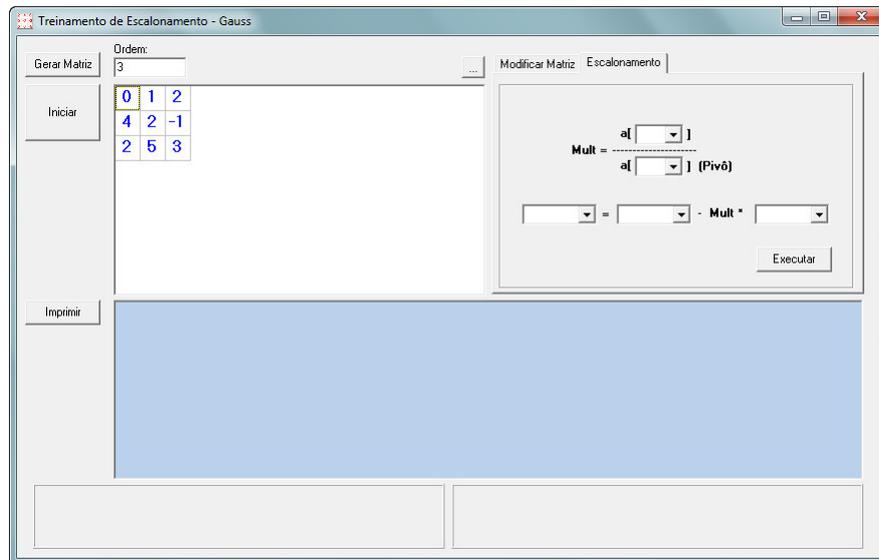


Figura 5.18: Primeiro passo - Treinamento Gauss

2. Ao clicar em "Iniciar" o sistema calcula o determinante da matriz e mostra na parte inferior esquerda da tela.

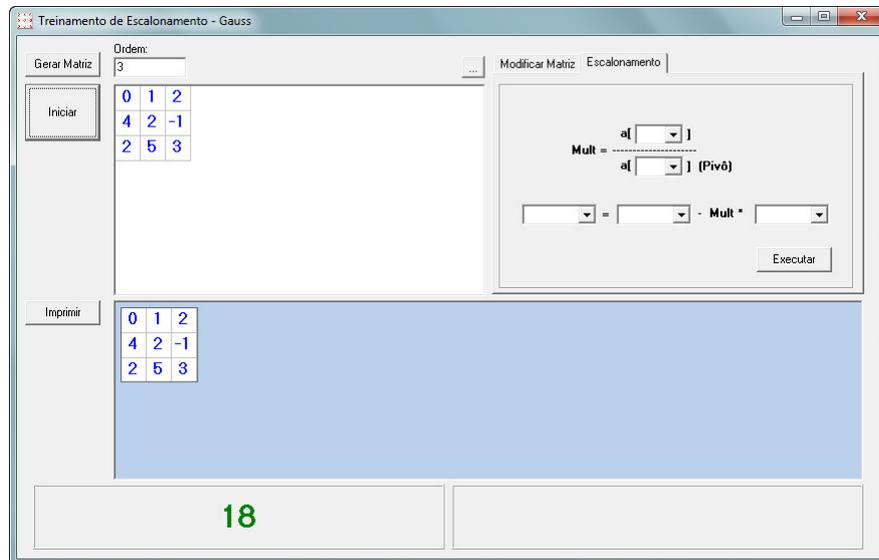


Figura 5.19: Segundo passo - Treinamento Gauss

3. Vale lembrar que cada coluna tem seu pivô e que os pivôs são os elementos da diagonal principal $(a_{11}, a_{22}, a_{33}, \dots)$ e o objetivo é zerar todos os elementos de cada coluna que estiverem abaixo de seu respectivo pivô.

Como o pivô inicial $a_{(1,1)}$ é igual a zero, devemos modificá-lo usando as propriedades das matrizes, gerando outra matriz equivalente, para isso podemos somar a $Linha_1$ com a

$Linha_3$ e substituir a $Linha_1$ pela soma, clique em "Modificar Matriz" e selecione Linha 1 / Somar / Linha 3 e clique "Executar".

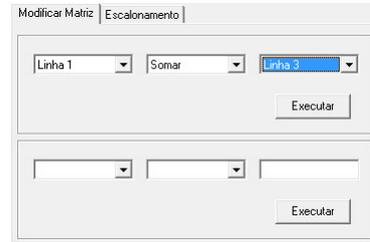


Figura 5.20: Terceiro passo - Treinamento Gauss

Todas as operações irão aparecer dentro da área azul e o determinante da nova matriz será calculado e exibido na parte inferior direita da tela.

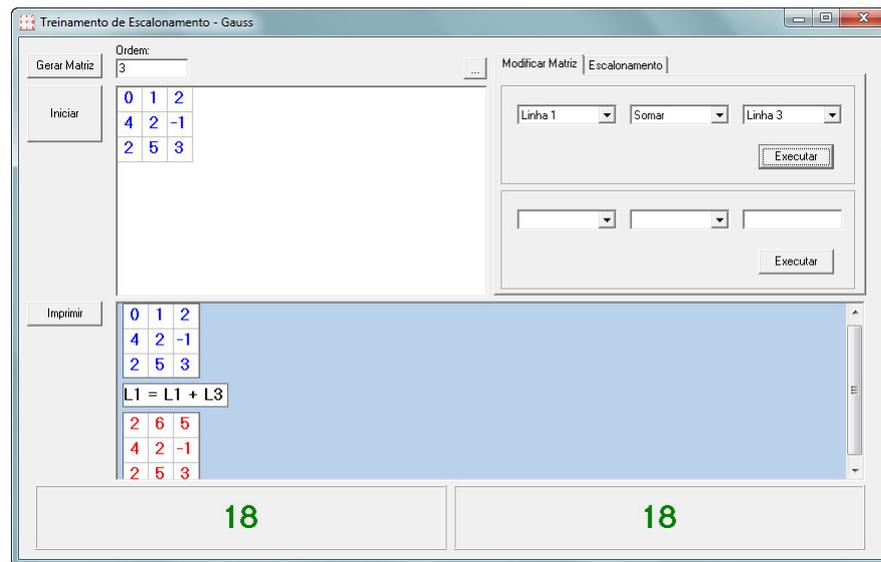


Figura 5.21: Terceiro passo - Treinamento Gauss - 2

4. Agora que o nosso primeiro pivô passou a ser $a_{(1,1)} = 2$ então iremos zerar o primeiro elemento abaixo dele que é $a_{(2,1)} = 4$.

Clique em "Escalonamento", lembrando que "Mult" é um multiplicador que tem como numerador o elemento a ser zerado e como denominador o pivô

$$Mult = \frac{a_{(2,1)}}{a_{(1,1)}} = \frac{4}{2} = 2$$

A linha a ser modificada será igual a própria linha subtraída da multiplicação de Mult pela linha do pivô.

$$Linha_2 = Linha_2 - Mult * Linha_1$$

Clique em "Executar".

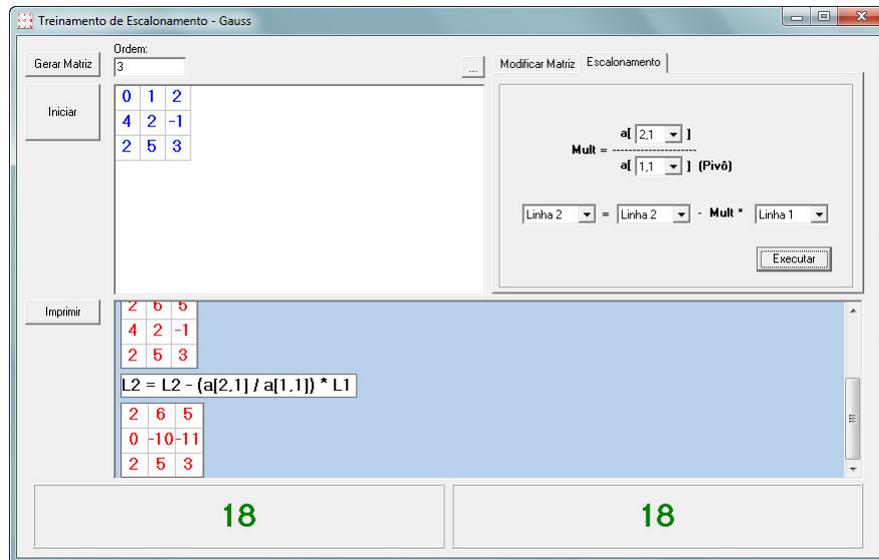


Figura 5.22: Quarto passo - Treinamento Gauss

Caso tudo esteja correto, a nova matriz terá o elemento pretendido zerado e os dois números abaixo da área azul, permaneceram iguais, pois são os determinantes da matriz inicial e da matriz modificada.

5. Vamos zerar o próximo elemento abaixo do pivô que é $a_{(3,1)} = 2$, então

$$Mult = \frac{a_{(3,1)}}{a_{(1,1)}} = \frac{2}{2} = 1$$

e a linha a ser modificada será:

$$Linha_3 = Linha_3 - Mult * Linha_1$$

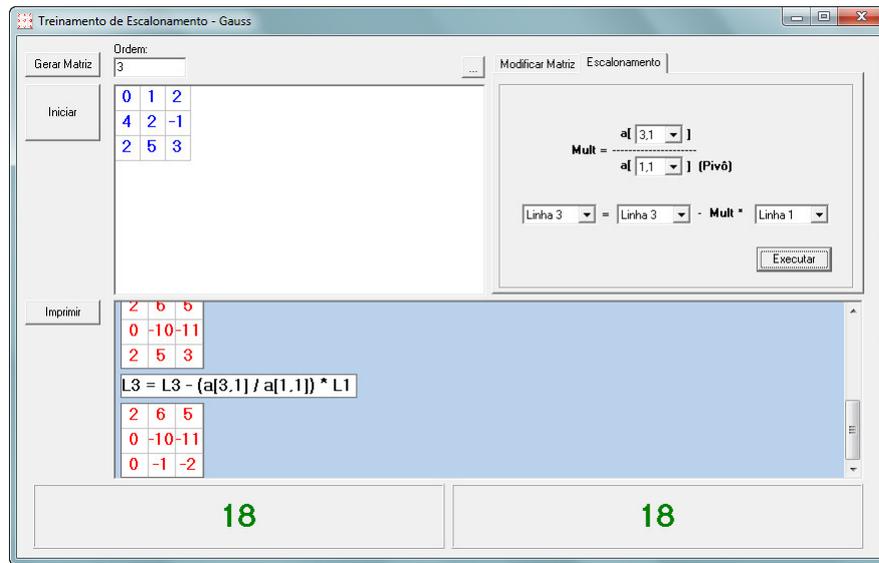


Figura 5.23: Quinto Passo - Treinamento Gauss

Vamos zerar $a_{(3,2)} = -1$, mudamos de coluna portanto mudamos de pivô que passa a ser $a_{(2,2)} = -10$ então

$$Mult = \frac{a_{(3,2)}}{a_{(2,2)}} = \frac{-1}{-10} = \frac{1}{10}$$

e a linha a ser modificada será:

$$Linha_3 = Linha_3 - Mult * Linha_2$$

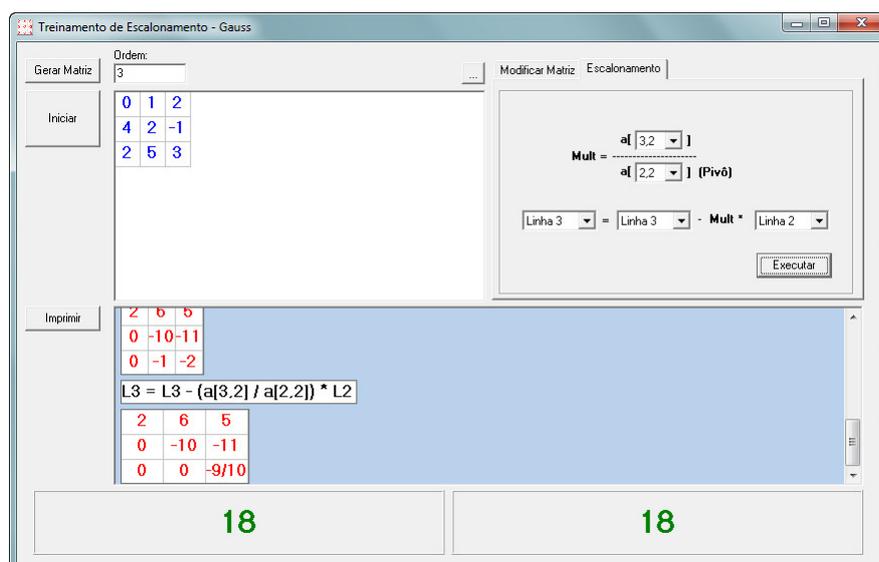


Figura 5.24: Quinto passo - Treinamento Gauss - 2

Após o escalonamento completo podemos calcular o determinante fazendo a multiplica-

ção dos pivôs resultantes.

$$\det(A) = 2 \cdot (-10) \cdot \left(-\frac{9}{10}\right) \Rightarrow \det(A) = 18$$

Considerações Finais

O material didático não deve ser esquecido, devemos lembrar que o uso do software é um auxílio no ensino-aprendizagem da matemática. O conteúdo deve vir antes da utilização do aplicativo, pois, não será suficiente somente uma abordagem computacional, devemos evitar o uso dos softwares somente para resolução de problemas, temos que questionar os resultados e comparar com os obtidos de forma manual.

É essencial conhecer a teoria por trás dos resultados, a ferramenta deve trazer consigo motivação e interesse para voos mais longos, partindo do ponto de vista que é possível resolver questões extremamente grandes e complexas.

A utilização da informática no ensino-aprendizagem da matemática, sendo feita da forma correta, pode contribuir para o melhor desempenho do aluno, despertando sua habilidade de pensar matematicamente, de forma a ver os conceitos matemáticos ligados ao mundo real.

No apêndice "A" segue o código fonte do aplicativo, deixando disponível para outras pessoas interessadas em programação para criar outros projetos usando parte ou o todo desse código.

O próximo passo será utilizar o software nas escolas e aferir sua eficiência no que diz respeito a fixação do conteúdo e verificar se o mesmo desempenha um papel motivador nas aulas, tanto para os alunos quanto para os professores.

Referências Bibliográficas

- ALTHOEN, S. and MCLAUGHLIN, R. (1987). *GaussJordan Reduction: A Brief History*. The American Mathematical Monthly, 2 edition.
- BASHMAKOVA, I. and SMIRNOVA, G. (2000). *The beginnings and evolution of algebra*. New York: Cambridge University Press, 23 edition.
- BEMFICA, A. (2015). Um pouco de história. <http://professorandrios.blogspot.com.br/2012/03/um-pouco-de-historia-gauss-e-o-seu.html>.
- DANTE, L. R. (2014). *Contexto e Aplicações*. Editora Ática, 2 edition.
- DIEUDONNÉ, J. (1981). *History of Functional Analysis*. NorthHolland Mathematics Studies, 1 edition.
- IEZZI, G., DOLCE, O., DEGENSZAJN, D., PÉRIGO, R., and ALMEIDA, N. D. (2010). *Matemática - Ciência e Aplicações*. Editora Saraiva, 6 edition.
- KOLMOGOROV, A. N. and YUSHKEVICH, A. P. (1981). *Mathematics of the 19th Century*. Biekhäuser Verlag, 1 edition.
- MIRANDA, D. D. (2015). Transformação para números fracionários. <http://www.mundoeducacao.com/matematica/transformacao-para-numeros-fracionarios.htm>.
- MUIR, T. (1890). *The theory of determinants in the historical order of development*. London: McMillan and Co. Ltd., 1 edition.
- OLIVEIRA, G. A. D. (2015). Matriz inversa. <http://www.mundoeducacao.com/matematica/matriz-inversa.htm>.
- UFF (2015). Gauss, o príncipe da matemática. <http://www.uff.br/sintoniamatematica/curiosidadesmatematicas/curiosidadesmatematicas-html/audio-gauss-br.html>.

USP (2015). Carl fridrich gauss (1777 - 1855). <http://ecalculo.if.usp.br/historia/gauss.htm>.

Apêndice A

Programa em Delphi

```
program Matriz;

uses
  Forms,
  Matriz_Principal in 'Matriz_Principal.pas' {F_Principal},
  U_Sobre in 'U_Sobre.pas' {F_Sobre},
  U_Trn_Determinante in 'U_Trn_Determinante.pas' {F_Trn_Determinante},
  U_Inversa in 'U_Inversa.pas' {F_Inversa},
  Funcoes in 'Funcoes.pas',
  U_ELineares in 'U_ELineares.pas' {F_ELineares},
  U_Determinante in 'U_Determinante.pas' {F_Determinante},
  CustomPreview in 'CustomPreview.pas';

{$R *.res}
begin
  F_Sobre:=TF_Sobre.Create(Application);
  F_Sobre.BorderStyle:= bsNone;
  F_Sobre.Show;
  F_Sobre.Update;
  Application.Title := 'Matriz';
  Application.CreateForm(TF_Principal, F_Principal);
  Application.Initialize;
  F_Sobre.Hide;
  F_Sobre.Release;
  Application.Run;
end.

=====

unit Matriz_Principal;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls,
  Forms, Dialogs, Menus, ComCtrls, ToolWin, ImgList, ExtCtrls,
  ShellAPI, funcoes;

type
  TF_Principal = class(TForm)
    MainMenu1: TMainMenu;
    ToolBar1: TToolBar;
    Arquivol: TMenuItem;
    Fechal: TMenuItem;
    Matrices1: TMenuItem;
    Determinante: TMenuItem;
    ImageList_On: TImageList;
    ToolButton1: TToolButton;
    ToolButton2: TToolButton;
    ToolButton3: TToolButton;
    ELineares: TMenuItem;
    ToolButton4: TToolButton;
    Sobre1: TMenuItem;
```

```

Panel1: TPanel;
StatusBar1: TStatusBar;
Inversa: TToolButton;
Inversal: TMenuItem;
ToolButton7: TToolButton;
ToolButton8: TToolButton;
TreinamentoGauss: TMenuItem;
AjudadoMatriz1: TMenuItem;
N1: TMenuItem;
SobredoMatriz1: TMenuItem;
procedure DeterminanteClick(Sender: TObject);
procedure ELinearesClick(Sender: TObject);
procedure FechalClick(Sender: TObject);
procedure FormCreate(Sender: TObject);
procedure InversaClick(Sender: TObject);
procedure TreinamentoGaussClick(Sender: TObject);
procedure ChamaHelp(var Msg: TMsg; var Handled: Boolean);
procedure SobredoMatriz1Click(Sender: TObject);
procedure AjudadoMatriz1Click(Sender: TObject);
private
  { Private declarations }
public
  { Public declarations }
end;

var
  F_Principal: TF_Principal;

implementation

uses U_Sobre, U_Determinante, U_ELineares, U_Inversa,
     U_Trn_Determinante;

{$R *.dfm}

procedure TF_Principal.DeterminanteClick(Sender: TObject);
begin
  F_Determinante:= TF_Determinante.Create(Application);
  F_Determinante.ShowModal;
  F_Determinante.Release;
  F_Determinante:=Nil;
end;

procedure TF_Principal.ELinearesClick(Sender: TObject);
begin
  F_ELineares:= TF_ELineares.Create(Application);
  F_ELineares.ShowModal;
  F_ELineares.Release;
  F_ELineares:=Nil;
end;

procedure TF_Principal.FechalClick(Sender: TObject);
begin
  Close;
end;

procedure TF_Principal.FormCreate(Sender: TObject);
begin
  Application.HintColor:= clAqua;
  Application.HintHidePause:= 250;
  Application.HintHidePause:= 10000;
  Application.OnMessage := ChamaHelp;
end;

procedure TF_Principal.InversaClick(Sender: TObject);
begin
  F_Inversa:= TF_Inversa.Create(Application);
  F_Inversa.ShowModal;
  F_Inversa.Release;
  F_Inversa:=Nil;
end;

procedure TF_Principal.TreinamentoGaussClick(Sender: TObject);

```

```

begin
  F_Trn_Determinante:= TF_Trn_Determinante.Create(Application);
  F_Trn_Determinante.ShowModal;
  F_Trn_Determinante.Release;
  F_Trn_Determinante:=Nil;
end;

procedure TF_Principal.SobredoMatriz1Click(Sender: TObject);
begin
  F_Sobre:= TF_Sobre.Create(Application);
  F_Sobre.ShowModal;
  F_Sobre.Release;
  F_Sobre:=Nil;
end;

procedure TF_Principal.ChamaHelp(var Msg: TMsg; var Handled: Boolean);
begin
  if (Screen.ActiveForm.ClassName = 'TMessageForm') = false then
    if Msg.message = WM_KEYDOWN then
      if Msg.wParam = VK_F1 then
        if ProcessoExiste('hh.exe') then
          ShowMessage('Ajuda_está_aberta.')
        else
          ShellExecute(0,nil,'Ajuda.CHM' ,nil, nil, SW_SHOWMAXIMIZED);
end;

procedure TF_Principal.AjudadoMatriz1Click(Sender: TObject);
begin
  if ProcessoExiste('hh.exe') then
    ShowMessage('A_Ajudaja_está_aberta.')
  else
    ShellExecute(0,nil,'Ajuda.CHM' ,nil, nil, SW_SHOWMAXIMIZED);
end;

end.

=====

unit U_Determinante;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls,
  Forms, Dialogs, StdCtrls, Grids, ExtCtrls, Funcoes, Buttons,
  RpDefine, RpBase, RpSystem, rpDevice, CustomPreview;

type
  TF_Determinante = class(TForm)
    Grid1: TStringGrid;
    Edit1: TEdit;
    Bt_Matriz: TButton;
    Label1: TLabel;
    Bt_Laplace: TButton;
    Painel: TPanel;
    SpeedButton1: TSpeedButton;
    Bt_Gauss: TButton;
    SB: TScrollBar;
    CheckBox1: TCheckBox;
    Button2: TButton;
    RvSystem1: TRvSystem;
    Bt_cancel: TButton;
    procedure Bt_MatrizClick(Sender: TObject);
    procedure Bt_LaplaceClick(Sender: TObject);
    procedure Edit1KeyPress(Sender: TObject; var Key: Char);
    procedure Limpa;
    procedure FormCreate(Sender: TObject);
    procedure SpeedButton1Click(Sender: TObject);
    procedure Bt_GaussClick(Sender: TObject);
    procedure FormMouseDown(Sender: TObject; Shift: TShiftState;
      MousePos: TPoint; var Handled: Boolean);
    procedure FormMouseWheelUp(Sender: TObject; Shift: TShiftState;
      MousePos: TPoint; var Handled: Boolean);
  end;

```

```

    procedure CheckBox1Click(Sender: TObject);
    procedure Button2Click(Sender: TObject);
    procedure RvSystem1Print(Sender: TObject);
    procedure RvSystem1PrintHeader(Sender: TObject);
    procedure Edit1Exit(Sender: TObject);
    procedure RvSystem1OverridePreview(ReportSystem: TRvSystem;
        OverrideMode: TOverrideMode; var OverrideForm: TForm);
    procedure RvSystem1OverrideSetup(ReportSystem: TRvSystem;
        OverrideMode: TOverrideMode; var OverrideForm: TForm);
    procedure RvSystem1OverrideStatus(ReportSystem: TRvSystem;
        OverrideMode: TOverrideMode; var OverrideForm: TForm);
    procedure Edit2KeyPress(Sender: TObject; var Key: Char);
    procedure Bt_cancelClick(Sender: TObject);

private
    { Private declarations }
public
    { Public declarations }
end;

var
    F_Determinante: TF_Determinante;

implementation

{$R *.dfm}

procedure TF_Determinante.FormCreate(Sender: TObject);
begin
    Grid1.OnDrawCell:= TMenuActions.SGDrawCell;
    Grid1.OnKeyPress:= TMenuActions.GridKeyPress;
    MSeq:= false;
    EqLinear:= false;
end;

procedure TF_Determinante.Limpa;
var
    i, j : integer;
begin
    for i:= 0 to Grid1.ColCount -1 do
        for j:= 0 to Grid1.ColCount -1 do
            Grid1.Cells[i, j]:= '';
        SetLength(M, 0, 0);
        Grid1.RowCount:= 1;
        Grid1.ColCount:= 1;
    end;
end;

procedure TF_Determinante.Bt_MatrizClick(Sender: TObject);
var
    x : integer;
begin
    if Edit1.text = '' then Exit;
    Limpa;
    x:= StrToInt(Edit1.text);
    Grid1.RowCount:= x;
    Grid1.ColCount:= x;
    Grid1.SetFocus;
end;

procedure TF_Determinante.Bt_LaplaceClick(Sender: TObject);
var
    d, i, j : integer;
begin
    if GridOk(Grid1) = false then exit;
    Cancelar:= false;
    Bt_cancel.Visible:= true;
    SB.DestroyComponents;
    d:= StrToInt(Edit1.text);
    SetLength(M, d, d);
    SetLength(SG, 0);
    Col_Tp:= d;
    Tops:= 5;
    CriaGridTex(SB, 'Método_de_Laplace');
end;

```

```

(Matriz de dimensão 1)
if Grid1.ColCount = 1 then
begin
  Bt_cancel.Visible:= false;
  CriaGridTex(SB,'Determinante_=_0');
  Painel.Caption:= Grid1.Cells[0,0];
  Exit;
end;
(Matriz recebe elementos do grid)
for i := 0 to d-1 do
  for j := 0 to d-1 do
    M[i,j]:= StrToFloat(Grid1.Cells[j,i]);
  (Escreve matriz inicial)
  CriaGridBox(SB,M);
  (Verifica linha/coluna zerada)
  if LCZerada(M) then
  begin
    Bt_cancel.Visible:= false;
    CriaGridTex(SB,'Determinante_=_0');
    Painel.Caption:= '0';
    exit;
  end;
  (Calcula determinante e escreve o resultado)
  Painel.Caption:=FloatToStr(Determinante(SB,M));
  CriaGridTex(SB,'Determinante_=_'+ Painel.Caption);
  (Posiciona na última linha do SB)
  SendMessage(SB.Handle, WM_VSCROLL, SB_BOTTOM, 0);
  if Cancelar then
  begin
    SB.DestroyComponents;
    Painel.Caption:='';
  end;
  Bt_cancel.Visible:= false;
end;

procedure TF_Determinante.Edit1KeyPress(Sender: TObject;
  var Key: Char);
begin
  if not (Ord(Key) in [8,13,48..57]) then Key:= #0;
end;

procedure TF_Determinante.SpeedButton1Click(Sender: TObject);
var
  i,x,iNum: Integer;
  List_A: TStringList;
begin
  List_A:= TStringList.Create;
  x:= StrToInt(Edit1.text);
  x:= x*x -1;
  for i:=0 to x do
  begin
    Randomize;
    iNum := Random(11);
    List_A.Add(IntToStr(iNum));
  end;
  (Verifica a quantidade de elementos)
  if List_A.Count <> Grid1.RowCount*Grid1.ColCount then
  begin
    ShowMessage('A_Quantidade_de_elementos_'+IntToStr(List_A.Count)+
      '_não_preenche_todas_as_células_da_Matriz!');
    Exit;
  end;
  (Mostra valores da lista no grid)
  MostraListGrid(List_A,Grid1);
end;

procedure TF_Determinante.Bt_GaussClick(Sender: TObject);
Var
  d,i,j: Integer;
  Aux: real;
  Mtemp: Mtz;
begin
  if GridOk(Grid1) = false then exit;

```

```

Cancelar:= false;
Bt_cancel.Visible:= true;
SB.DestroyComponents;
d:= StrToInt(Edit1.text);
SetLength(M,d,d+1);
SetLength(SG,0);
Col_Tp:= d;
Tops:= 5;
CriaGridTex(SB,'Método_de_Gauss');
{Matriz de dimensão 1}
if Grid1.ColCount = 1 then
begin
  Bt_cancel.Visible:= false;
  CriaGridTex(SB,'Determinante_=_0');
  Painel.Caption:= Grid1.Cells[0,0];
  Exit;
end;
{Matriz recebe elementos do grid}
for i := 0 to d-1 do
  for j := 0 to d-1 do
    M[i,j]:= StrToFloat(Grid1.Cells[j,i]);
  {Escreve matriz inicial}
  CriaGridBox(SB,M);
  {Verifica linha/coluna zerada}
  if LCZerada(M) then
  begin
    Bt_cancel.Visible:= false;
    CriaGridTex(SB,'Determinante_=_0');
    Painel.Caption:= '0';
    exit;
  end;
  {Escalona a Matriz Gauss}
  EscalonaMatriz(SB,M);
  {Verifica cancelamento}
  if Cancelar then
  begin
    Bt_cancel.Visible:= false;
    SB.DestroyComponents;
    Painel.Caption:= '';
    Exit;
  end;
  {Calcula Variáveis}
  SetLength(MTemp,1,d);
  Aux:=1;
  for i := 0 to d-1 do
    for j := 0 to d-1 do
      if i = j then
        begin
          Aux:= Aux*M[i,j];
          Mtemp[0,j]:= M[i,j];
        end;
  CriaGridTex(SB,'Diagonal_Final_=_');
  CriaGridBox(SB,MTemp);
  CriaGridTex(SB,'Determinante_=_'+ Fracao(Aux));
  Painel.Caption:= Fracao(Aux);
  {Posiciona na última linha do SB}
  SendMessage(SB.Handle, WM_VSCROLL, SB_BOTTOM, 0);
  Bt_cancel.Visible:= false;
end;

procedure TF_Determinante.FormMouseDown(Sender: TObject;
  Shift: TShiftState; MousePos: TPoint; var Handled: Boolean);
begin
  with SB.VertScrollBar do
    begin
      Position := Position + Increment;
    end;
end;

procedure TF_Determinante.FormMouseWheelUp(Sender: TObject;
  Shift: TShiftState; MousePos: TPoint; var Handled: Boolean);
begin
  with SB.VertScrollBar do

```

```

begin
    Position := Position - Increment;
end;
end;

procedure TF_Determinante.CheckBox1Click(Sender: TObject);
begin
    if CheckBox1.Checked then Mseq:= true else Mseq:= false;
end;

procedure TF_Determinante.Button2Click(Sender: TObject);
begin
    if SB.ComponentCount = 0 then Exit;
    with RVSystem1.SystemPrinter do
    begin
        MarginRight:= 0.2;
        MarginLeft:= 0.2;
        MarginTop:= 1;
        MarginBottom:= 1;
        rpDev.Copies:= 1;
        Copies:= rpDev.Copies;
        rpDev.Orientation:= poPortrait;
        Orientation:= rpDev.Orientation;
        RVSystem1.SystemPreview.FormState:= wsMaximized;
        RVSystem1.SystemSetups:= RVSystem1.SystemSetups - [ssAllowSetup];
        rpDev.SelectPaper('A4', false);
        RVSystem1.Execute;
    end;
end;

procedure TF_Determinante.RvSystem1Print(Sender: TObject);
Var
    x,i,j: integer;
    TSG : TstringGrid;
begin
    with RVSystem1.BaseReport do
    begin
        for x:= 0 to SB.ComponentCount-1 do
        begin
            TSG:= TStringGrid(SB.Components[x]);
            for i:= 0 to TSG.RowCount - 1 do
            begin
                for j:= 0 to TSG.ColCount - 1 do
                begin
                    if TSG.ColCount = 1 then
                        PrintLeft(TSG.Cells[j,i],MarginLeft)
                    else
                        PrintCenter(TSG.Cells[j,i],MarginLeft+0.2+(j/2));
                    end;
                    NewLine;
                    NewLine;
                    if LinesLeft<1 then NewPage;
                end;
            end;
        end;
    end;

procedure TF_Determinante.RvSystem1PrintHeader(Sender: TObject);
begin
    with RVSystem1.BaseReport do
    begin
        SetFont('Verdana',12);
        Bold := True;
        PrintCenter('Cálculo_de_Determinante',PageWidth/2);
        NewLine;
        Canvas.Pen.Width := 2;
        Canvas.Pen.Color := clBlack;
        MoveTo(MarginLeft,YPos);
        LineTo(PageWidth-MarginRight,YPos);
        NewLine;
        SetFont('Verdana',8);
        Bold := False;
    end;
end;

```

```

end;

procedure TF_Determinante.Edit1Exit(Sender: TObject);
begin
  Bt_MatrizClick(Bt_Matriz);
end;

procedure TF_Determinante.RvSystem1OverridePreview(ReportSystem:
  TRvSystem; OverrideMode: TOverrideMode; var OverrideForm: TForm);
begin
  ExecutePreviewForm(Self, ReportSystem, OverrideMode, OverrideForm);
end;

procedure TF_Determinante.RvSystem1OverrideSetup(ReportSystem:
  TRvSystem; OverrideMode: TOverrideMode; var OverrideForm: TForm);
begin
  ExecuteSetupForm(Self, ReportSystem, OverrideMode, OverrideForm);
end;

procedure TF_Determinante.RvSystem1OverrideStatus(ReportSystem:
  TRvSystem; OverrideMode: TOverrideMode; var OverrideForm: TForm);
begin
  ExecuteStatusForm(Self, ReportSystem, OverrideMode, OverrideForm);
end;

procedure TF_Determinante.Edit2KeyPress(Sender: TObject;
  var Key: Char);
begin
  if not (Ord(Key) in [3, 8, 13, 22, 44, 45, 48..57, 59]) then Key := #0;
end;

procedure TF_Determinante.Bt_cancelClick(Sender: TObject);
begin
  Cancelar := true;
  Bt_cancel.Visible := false;
end;

end.

=====

unit U_ELineares;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls,
  Forms, Dialogs, StdCtrls, Grids, ExtCtrls, Funcoes, Menus, Printers,
  Buttons, ShellAPI, UPrintGrid, RpDefine, RpBase, RpSystem, rpDevice,
  CustomPreview;

type
  TF_ELineares = class(TForm)
    Grid1: TStringGrid;
    Edit1: TEdit;
    Bt_Matriz: TButton;
    Label1: TLabel;
    Bt_Gauss: TButton;
    SB: TScrollBar;
    SpeedButton1: TSpeedButton;
    CheckBox1: TCheckBox;
    Button2: TButton;
    RvSystem1: TRvSystem;
    Bt_cancel: TButton;
    procedure Bt_MatrizClick(Sender: TObject);
    procedure Edit1KeyPress(Sender: TObject; var Key: Char);
    procedure Grid1KeyPress(Sender: TObject; var Key: Char);
    procedure LimpaGrid;
    procedure Bt_GaussClick(Sender: TObject);
    procedure Edit2KeyPress(Sender: TObject; var Key: Char);
    procedure FormCreate(Sender: TObject);
    procedure FormMouseWheelDown(Sender: TObject; Shift: TShiftState;
      MousePos: TPoint; var Handled: Boolean);
  end;

```

```

    procedure FormMouseWheelUp(Sender: TObject; Shift: TShiftState;
      MousePos: TPoint; var Handled: Boolean);
    procedure SpeedButton1Click(Sender: TObject);
    procedure CheckBox1Click(Sender: TObject);
    procedure Button2Click(Sender: TObject);
    procedure RvSystem1Print(Sender: TObject);
    procedure RvSystem1PrintHeader(Sender: TObject);
    procedure Edit1Exit(Sender: TObject);
    procedure RvSystem1OverridePreview(ReportSystem: TRvSystem;
      OverrideMode: TOverrideMode; var OverrideForm: TForm);
    procedure RvSystem1OverrideSetup(ReportSystem: TRvSystem;
      OverrideMode: TOverrideMode; var OverrideForm: TForm);
    procedure RvSystem1OverrideStatus(ReportSystem: TRvSystem;
      OverrideMode: TOverrideMode; var OverrideForm: TForm);
    procedure Bt_cancelClick(Sender: TObject);

private
  { Private declarations }
public
  { Public declarations }
end;

var
  F_ELineares: TF_ELineares;

implementation

{$R *.dfm}

procedure TF_ELineares.LimpaGrid;
var
  i, j : integer;
begin
  for i:= 0 to Grid1.RowCount -1 do
    for j:= 0 to Grid1.ColCount -1 do
      Grid1.Cells[i, j]:= '';
    SetLength(M, 0, 0);
    Grid1.RowCount:= 1;
    Grid1.ColCount:= 1;
  end;
end;

procedure TF_ELineares.Bt_MatrizClick(Sender: TObject);
var
  x : integer;
begin
  if Edit1.text = '' then Exit;
  LimpaGrid;
  x:= StrToInt(Edit1.text);
  Grid1.RowCount:= x; {linhas do grid}
  Grid1.ColCount:= x+1; {colunas do grid}
  Grid1.SetFocus;
  Col_Tp:= x+1;
end;

procedure TF_ELineares.Bt_GaussClick(Sender: TObject);
var
  d, i, j, x :integer;
  Resp: Mtz1;
begin
  if GridOk(Grid1) = false then exit;
  Cancelar:= false;
  Bt_cancel.Visible:= true;
  d:= StrToInt(Edit1.text);
  SetLength(M, d, d+1);
  //Col_Tp:= d+1;
  {Preenche os valores do grid na Matriz}
  for i:= 0 to d-1 do
    for j:= 0 to d do
      M[i, j]:= StrToFloat(Grid1.Cells[j, i]);
    {Limpa escalonamento anterior}
  end;
  SB.DestroyComponents;
  SetLength(SG, 0);
  Tops:=5;
end;

```

```

CriaGridBox(SB,M);
{Verifica linha coluna zerada}
if LCZerada(M) then
begin
  Bt_cancel.Visible:= false;
  CriaGridTex(SB, 'Sistema_impossivel_-_Não_tem_Solução!');
  ShowMessage('Sistema_impossivel_-_Não_tem_Solução!');
  exit;
end;
{Escalona a Matriz Gauss}
EscalonaMatriz(SB,M);
{Verifica cancelamento}
if Cancelar then
begin
  SB.DestroyComponents;
  Exit;
end;
{Verifica se existe solução}
for i := 0 to d-1 do
  for j := 0 to d-1 do
    if (i = j) and (M[i,j] = 0) then
      begin
        Bt_cancel.Visible:= false;
        if M[j,d] = 0 then
          begin
            CriaGridTex(SB,
              'Sistema_possivel_e_indeterminado_-_Infinitas_Soluções!');
            ShowMessage(
              'Sistema_possivel_e_indeterminado_-_Infinitas_Soluções!');
          end
        else
          begin
            CriaGridTex(SB, 'Sistema_impossivel_-_Não_tem_Solução!');
            ShowMessage('Sistema_impossivel_-_Não_tem_Solução!');
          end;
          Exit;
        end;
      end;
    {Calcula as Variáveis}
    Resp:= Variaveis(M,d);
    {Mostra Respostas}
    x:=1;
    for i:= d-1 downto 0 do
      begin
        CriaGridTex(SB, 'X['+IntToStr(x)+']='+Fracao(Resp[i]));
        Inc(x);
      end;
      SendMessage(SB.Handle, WM_VSCROLL, SB_BOTTOM, 0);
      Bt_cancel.Visible:= false;
    end;

procedure TF_ELineares.Edit1KeyPress(Sender: TObject; var Key: Char);
begin
  if not (Ord(Key) in [8,13,48..57]) then Key:= #0;
end;

procedure TF_ELineares.Grid1KeyPress(Sender: TObject; var Key: Char);
begin
  if not (Ord(Key) in [8,13,44,45,48..57]) then Key:= #0;
end;

procedure TF_ELineares.Edit2KeyPress(Sender: TObject; var Key: Char);
begin
  if not (Ord(Key) in [3,8,13,22,44,45,48..57,59]) then Key:= #0;
end;

procedure TF_ELineares.FormCreate(Sender: TObject);
begin
  Grid1.OnDrawCell:= TMenuActions.SGDrawCell;
  Grid1.OnKeyPress:= TMenuActions.GridKeyPress;
  MSeq:= false;
  EqLinear:= true;
end;

```

```

procedure TF_ELineares.FormMouseWheelDown(Sender: TObject;
  Shift: TShiftState; MousePos: TPoint; var Handled: Boolean);
begin
  with SB.VertScrollBar do
    begin
      Position := Position + Increment;
    end;
  end;
end;

procedure TF_ELineares.FormMouseWheelUp(Sender: TObject;
  Shift: TShiftState; MousePos: TPoint; var Handled: Boolean);
begin
  with SB.VertScrollBar do
    begin
      Position := Position - Increment;
    end;
  end;
end;

procedure TF_ELineares.SpeedButton1Click(Sender: TObject);
var
  i,x,iNum: Integer;
  List_A: TStringList;
begin
  List_A:= TStringList.Create;
  x:= StrToInt(Edit1.text);
  x:= x*(x+1) -1;
  for i:=0 to x do
    begin
      Randomize;
      iNum := Random(11);
      List_A.Add(IntToStr(iNum));
    end;
  {Verifica a quantidade de elementos}
  if List_A.Count <> Grid1.RowCount*Grid1.ColCount then
    begin
      ShowMessage('A_Quantidade_de_elementos_' +IntToStr(List_A.Count)
        +'_nãopreenche_todas_as_células_da_Matriz!');
      Exit;
    end;
  {Mostra valores da lista no grid}
  MostraListGrid(List_A,Grid1);
end;

procedure TF_ELineares.CheckBox1Click(Sender: TObject);
begin
  if CheckBox1.Checked then Mseq:= true else Mseq:= false;
end;

procedure TF_ELineares.Button2Click(Sender: TObject);
begin
  if SB.ComponentCount = 0 then Exit;
  with RVSystem1.SystemPrinter do
    begin
      MarginRight:= 0.2;
      MarginLeft:= 0.2;
      MarginTop:= 1;
      MarginBottom:= 1;
      rpDev.Copies:= 1;
      Copies:= rpDev.Copies;
      rpDev.Orientation:= poPortrait;
      Orientation:= rpDev.Orientation;
      RVSystem1.SystemPreview.FormState:= wsMaximized;
      RVSystem1.SystemSetups:= RVSystem1.SystemSetups - [ssAllowSetup];
      rpDev.SelectPaper('A4',false);
      RVSystem1.Execute;
    end;
  end;

procedure TF_ELineares.RvSystem1Print(Sender: TObject);
Var
  x,i,j: integer;
  TSG : TstringGrid;
begin

```

```

with RVSystem1.BaseReport do
begin
  for x:= 0 to SB.ComponentCount-1 do
  begin
    TSG:= TStringGrid(SB.Components[x]);
    for i:= 0 to TSG.RowCount - 1 do
    begin
      for j:= 0 to TSG.ColCount - 1 do
      begin
        if TSG.ColCount = 1 then
          PrintLeft (TSG.Cells[j,i],MarginLeft)
        else
          PrintCenter(TSG.Cells[j,i],MarginLeft+0.2+(j/2));
        end;
        NewLine;
        NewLine;
        if LinesLeft<1 then NewPage;
      end;
    end;
  end;
end;

procedure TF_ELineares.RvSystem1PrintHeader(Sender: TObject);
begin
  with RVSystem1.BaseReport do
  begin
    SetFont('Verdana',12);
    Bold := True;
    PrintCenter('Cálculo_de_Equações_Lineares',PageWidth/2);
    NewLine;
    Canvas.Pen.Width := 2;
    Canvas.Pen.Color := clBlack;
    MoveTo(MarginLeft,YPos);
    LineTo(PageWidth-MarginRight,YPos);
    NewLine;
    SetFont('Verdana',8);
    Bold := False;
  end;
end;

procedure TF_ELineares.Edit1Exit(Sender: TObject);
begin
  Bt_MatrizClick(Bt_Matriz);
end;

procedure TF_ELineares.RvSystem1OverridePreview(ReportSystem:
TRvSystem; OverrideMode: TOverrideMode; var OverrideForm: TForm);
begin
  ExecutePreviewForm(Self,ReportSystem,OverrideMode,OverrideForm);
end;

procedure TF_ELineares.RvSystem1OverrideSetup(ReportSystem:
TRvSystem; OverrideMode: TOverrideMode; var OverrideForm: TForm);
begin
  ExecuteSetupForm(Self,ReportSystem,OverrideMode,OverrideForm);
end;

procedure TF_ELineares.RvSystem1OverrideStatus(ReportSystem:
TRvSystem; OverrideMode: TOverrideMode; var OverrideForm: TForm);
begin
  ExecuteStatusForm(Self,ReportSystem,OverrideMode,OverrideForm);
end;

procedure TF_ELineares.Bt_cancelClick(Sender: TObject);
begin
  Cancelar:= true;
  Bt_cancel.Visible:= false;
end;

end.
=====

```

```

unit U_Inversa;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls,
  Forms, Dialogs, StdCtrls, Grids, ExtCtrls, Funcoes, Buttons, RpDefine,
  RpBase, RpSystem, rpDevice, CustomPreview;

type
  TF_Inversa = class(TForm)
    Edit1: TEdit;
    Bt_Matriz: TButton;
    Label1: TLabel;
    Grid1: TStringGrid;
    Bt_Gauss: TButton;
    SB: TScrollBar;
    SpeedButton1: TSpeedButton;
    CheckBox1: TCheckBox;
    Button2: TButton;
    RvSystem1: TRvSystem;
    Bt_cancel: TButton;
    procedure Bt_MatrizClick(Sender: TObject);
    procedure Edit1KeyPress(Sender: TObject; var Key: Char);
    procedure Grid1KeyPress(Sender: TObject; var Key: Char);
    procedure Grid1KeyDown(Sender: TObject; var Key: Word;
      Shift: TShiftState);
    procedure Edit1KeyDown(Sender: TObject; var Key: Word;
      Shift: TShiftState);
    procedure LimpaGrid;
    procedure Edit2KeyPress(Sender: TObject; var Key: Char);
    procedure Bt_GaussClick(Sender: TObject);
    procedure FormCreate(Sender: TObject);
    procedure FormMouseWheelDown(Sender: TObject; Shift: TShiftState;
      MousePos: TPoint; var Handled: Boolean);
    procedure FormMouseWheelUp(Sender: TObject; Shift: TShiftState;
      MousePos: TPoint; var Handled: Boolean);
    procedure SpeedButton1Click(Sender: TObject);
    procedure SBMouseWheelDown(Sender: TObject; Shift: TShiftState;
      MousePos: TPoint; var Handled: Boolean);
    procedure SBMouseWheelUp(Sender: TObject; Shift: TShiftState;
      MousePos: TPoint; var Handled: Boolean);
    procedure CheckBox1Click(Sender: TObject);
    procedure Button2Click(Sender: TObject);
    procedure RvSystem1Print(Sender: TObject);
    procedure RvSystem1PrintHeader(Sender: TObject);
    procedure Edit1Exit(Sender: TObject);
    procedure RvSystem1OverridePreview(ReportSystem: TRvSystem;
      OverrideMode: TOverrideMode; var OverrideForm: TForm);
    procedure RvSystem1OverrideSetup(ReportSystem: TRvSystem;
      OverrideMode: TOverrideMode; var OverrideForm: TForm);
    procedure RvSystem1OverrideStatus(ReportSystem: TRvSystem;
      OverrideMode: TOverrideMode; var OverrideForm: TForm);
    procedure Bt_cancelClick(Sender: TObject);

  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  F_Inversa: TF_Inversa;

implementation

{$R *.dfm}

procedure TF_Inversa.LimpaGrid;
var
  i, j : integer;
begin
  for i:= 0 to Grid1.RowCount -1 do

```

```

    for j:= 0 to Grid1.ColCount -1 do
    begin
        Grid1.Cells[i,j]:='';
    end;
    SetLength(M,0,0);
    Grid1.RowCount:= 1;
    Grid1.ColCount:= 1;
end;

procedure TF_Inversa.Bt_MatrizClick(Sender: TObject);
var
    x : integer;
begin
    if Edit1.text = '' then Exit;
    LimpaGrid;
    x:= StrToInt(Edit1.text);
    Grid1.RowCount:= x; {linhas do grid}
    Grid1.ColCount:= x; {colunas do grid}
    Grid1.SetFocus;
end;

procedure TF_Inversa.Edit1KeyPress(Sender: TObject; var Key: Char);
begin
    if not (Ord(key) in [8,13,48..57]) then Key:= #0;
end;

procedure TF_Inversa.Grid1KeyPress(Sender: TObject; var Key: Char);
begin
    if not (Ord(key) in [8,13,44,45,48..57]) then Key:= #0;
end;

procedure TF_Inversa.Grid1KeyDown(Sender: TObject; var Key: Word;
    Shift: TShiftState);
begin
    if Key = 13 then Key:=9;
end;

procedure TF_Inversa.Edit1KeyDown(Sender: TObject; var Key: Word;
    Shift: TShiftState);
begin
    if Key = 13 then Perform(Wm_NextDlgCtl,0,0);
end;

procedure TF_Inversa.Edit2KeyPress(Sender: TObject; var Key: Char);
begin
    if not (Ord(key) in [3,8,13,22,44,45,48..57,59]) then Key:= #0;
    {3 CTR+C e 22 CTR+V}
end;

procedure TF_Inversa.Bt_GaussClick(Sender: TObject);
var
    i,j,k,d : Integer;
    Resp: Mtz1;
    MtzInv, Mtemp: Mtz;
    W: real;
begin
    if GridOk(Grid1) = false then exit;
    {Limpa escalonamento anterior}
    Cancelar:= false;
    Bt_cancel.Visible:= true;
    d:= StrToInt(Edit1.text);
    {Cria Matriz resposta e temporaria}
    SetLength(MtzInv,d,d);
    SetLength(Mtemp,d,d+1);
    {Preenche os valores do grid nas Matrizes}
    for i:= 0 to d-1 do
        for j:= 0 to d-1 do
            begin
                MtzInv[i,j]:= StrToFloat(Grid1.Cells[j,i]);
                Mtemp[i,j]:= StrToFloat(Grid1.Cells[j,i]);
            end;
    {Verifica o determinante da matriz temporaria linha/coluna zerada}
    if LCZerada(Mtemp) then

```

```

begin
  Bt_cancel.Visible:= false;
  CriaGridTex(SB,'Determinante_0_Matriz_não_Inversível. ');
  ShowMessage('Determinante_0_Matriz_não_Inversível. ');
  exit;
end;
{Escalona a Matriz Gauss}
EscalonaMatriz(SB,Mtemp);
{Verifica cancelamento}
if Cancelar then
begin
  SB.DestroyComponents;
  Exit;
end;
{Calcula determinante}
W:=1;
for i := 0 to d-1 do
  for j := 0 to d-1 do
    if i = j then
      W:= W*Mtemp[i,j];
  if W = 0 then
begin
  Bt_cancel.Visible:= false;
  CriaGridTex(SB,'Determinante_0_Matriz_não_Inversível. ');
  ShowMessage('Determinante_0_Matriz_não_Inversível. ');
  exit;
end;
{Escreve matriz inicial}
SB.DestroyComponents;
SetLength(SG,0);
Tops:=5;
Col_Tp:= d;
CriaGridBox(SB,MtzInv);
{Calcula Matriz Inversa}
for k:= 1 to d do
begin
  {Cria Matriz}
  SetLength(M,d,d+1);
  Col_Tp:= d+1;
  for i := 0 to d-1 do
  for j:= 0 to d do
  begin
    if j = d then
    begin
      if i = k-1 then M[i,j]:= 1
      else M[i,j]:= 0;
    end
    else M[i,j]:= StrToFloat(Grid1.Cells[i,j]);
  end;
  if Mseq then CriaGridTex(SB,IntToStr(k)+'ª_Matriz_Transposta');
  if Mseq then CriaGridBox(SB,M);
  {Escalona a Matriz}
  EscalonaMatriz(SB,M);
  {Verifica cancelamento}
  if Cancelar then
  begin
    SB.DestroyComponents;
    Exit;
  end;
  {Calcula as Variáveis}
  Resp:= Variaveis(M,d);
  {Inclui dados Matriz resposta}
  j:=0;
  for i := d-1 downto 0 do
  begin
    MtzInv[k-1,j]:=Resp[i];
    if Mseq then CriaGridTex(SB,'X['+IntToStr(j+1)+']_='
      +Fracao(Resp[i]));
    Inc(j);
  end;
end;
{Mostra a Matriz resposta}
Col_Tp:= d;

```

```

    CriaGridBox(SB,MtzInv);
    SendMessage(SB.Handle, WM_VSCROLL, SB_BOTTOM, 0);
    Bt_cancel.Visible:= false;
end;

procedure TF_Inversa.FormCreate(Sender: TObject);
begin
    Grid1.OnDrawCell:= TMenuActions.SGDrawCell;
    Grid1.OnKeyPress:= TMenuActions.GridKeyPress;
    MSeq:= false;
    EqLinear:= false;
end;

procedure TF_Inversa.FormMouseWheelDown(Sender: TObject;
    Shift: TShiftState; MousePos: TPoint; var Handled: Boolean);
begin
    with SB.VertScrollBar do
    begin
        Position := Position + Increment;
    end;
end;

procedure TF_Inversa.FormMouseWheelUp(Sender: TObject;
    Shift: TShiftState; MousePos: TPoint; var Handled: Boolean);
begin
    with SB.VertScrollBar do
    begin
        Position := Position - Increment;
    end;
end;

procedure TF_Inversa.SpeedButton1Click(Sender: TObject);
var
    i,x,iNum: Integer;
    List_A: TStringList;
begin
    List_A:= TStringList.Create;
    x:= StrToInt(Edit1.text);
    x:= x*x -1;
    for i:=0 to x do
    begin
        Randomize;
        iNum := Random(11);
        List_A.Add(IntToStr(iNum));
    end;
    {Verifica a quantidade de elementos}
    if List_A.Count <> Grid1.RowCount*Grid1.ColCount then
    begin
        ShowMessage('A_Quantidade_de_elementos_' +IntToStr(List_A.Count)
            +'_não_preenche_todas_as_células_da_Matriz!');
        Exit;
    end;
    {Mostra valores da lista no grid}
    MostraListGrid(List_A,Grid1);
end;

procedure TF_Inversa.SBMouseWheelDown(Sender: TObject; Shift: TShiftState;
    MousePos: TPoint; var Handled: Boolean);
begin
    with SB.VertScrollBar do
    begin
        Position := Position + Increment;
    end;
end;

procedure TF_Inversa.SBMouseWheelUp(Sender: TObject; Shift: TShiftState;
    MousePos: TPoint; var Handled: Boolean);
begin
    with SB.VertScrollBar do
    begin
        Position := Position - Increment;
    end;
end;

```

```
procedure TF_Inversa.CheckBox1Click(Sender: TObject);
begin
  if CheckBox1.Checked then Mseq:= true else Mseq:= false;
end;

procedure TF_Inversa.Button2Click(Sender: TObject);
begin
  if SB.ComponentCount = 0 then Exit;
  with RVSystem1.SystemPrinter do
  begin
    MarginRight:= 0.2;
    MarginLeft:= 0.2;
    MarginTop:= 1;
    MarginBottom:= 1;
    rpDev.Copies:= 1;
    Copies:= rpDev.Copies;
    rpDev.Orientation:= poPortrait;
    Orientation:= rpDev.Orientation;
    RVSystem1.SystemPreview.FormState:= wsMaximized;
    RVSystem1.SystemSetups:= RVSystem1.SystemSetups - [ssAllowSetup];
    rpDev.SelectPaper('A4', false);
    RVSystem1.Execute;
  end;
end;

procedure TF_Inversa.RvSystem1Print(Sender: TObject);
Var
  x,i,j: integer;
  TSG : TstringGrid;
begin
  with RVSystem1.BaseReport do
  begin
    for x:= 0 to SB.ComponentCount-1 do
    begin
      TSG:= TStringGrid(SB.Components[x]);
      for i:= 0 to TSG.RowCount - 1 do
      begin
        for j:= 0 to TSG.colCount - 1 do
        begin
          if TSG.colCount = 1 then
            PrintLeft(TSG.Cells[j,i],MarginLeft)
          else
            PrintCenter(TSG.Cells[j,i],MarginLeft+0.2+(j/2));
          end;
          NewLine;
          NewLine;
          if LinesLeft<1 then NewPage;
        end;
      end;
    end;
  end;

procedure TF_Inversa.RvSystem1PrintHeader(Sender: TObject);
begin
  with RVSystem1.BaseReport do
  begin
    SetFont('Verdana',12);
    Bold := True;
    PrintCenter('Cálculo_de_Matriz_Inversa',PageWidth/2);
    NewLine;
    Canvas.Pen.Width := 2;
    Canvas.Pen.Color := clBlack;
    MoveTo(MarginLeft,YPos);
    LineTo(PageWidth-MarginRight,YPos);
    NewLine;
    SetFont('Verdana',8);
    Bold := False;
  end;
end;

procedure TF_Inversa.Edit1Exit(Sender: TObject);
begin
```

```

    Bt_MatrizClick(Bt_Matriz);
end;

procedure TF_Inversa.RvSystem1OverridePreview(ReportSystem: TRvSystem;
    OverrideMode: TOverrideMode; var OverrideForm: TForm);
begin
    ExecutePreviewForm(Self, ReportSystem, OverrideMode, OverrideForm);
end;

procedure TF_Inversa.RvSystem1OverrideSetup(ReportSystem: TRvSystem;
    OverrideMode: TOverrideMode; var OverrideForm: TForm);
begin
    ExecuteSetupForm(Self, ReportSystem, OverrideMode, OverrideForm);
end;

procedure TF_Inversa.RvSystem1OverrideStatus(ReportSystem: TRvSystem;
    OverrideMode: TOverrideMode; var OverrideForm: TForm);
begin
    ExecuteStatusForm(Self, ReportSystem, OverrideMode, OverrideForm);
end;

procedure TF_Inversa.Bt_cancelClick(Sender: TObject);
begin
    Cancelar:= true;
    Bt_cancel.Visible:= false;
end;

end.

=====

unit U_Trn_Determinante;

interface

uses
    Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls,
    Forms, Dialogs, StdCtrls, Grids, ExtCtrls, Funcoes, Buttons,
    ComCtrls, RpDefine, RpBase, RpSystem, rpDevice, CustomPreview;

type
    TF_Trn_Determinante = class(TForm)
        Grid1: TStringGrid;
        Edit1: TEdit;
        Bt_Matriz: TButton;
        Label1: TLabel;
        Paine1: TPanel;
        SpeedButton1: TSpeedButton;
        Bt_Gauss: TButton;
        SB: TScrollBar;
        PageControl1: TPageControl;
        TabSheet1: TTabSheet;
        TabSheet2: TTabSheet;
        GroupBox1: TGroupBox;
        CBox1: TComboBox;
        CBox2: TComboBox;
        CBox3: TComboBox;
        Bt_Soma: TButton;
        GroupBox2: TGroupBox;
        CBox4: TComboBox;
        CBox5: TComboBox;
        Bt_Mult: TButton;
        Edit3: TEdit;
        Paine1: TPanel;
        GroupBox3: TGroupBox;
        Label4: TLabel;
        Pivot1: TComboBox;
        Label5: TLabel;
        Pivot2: TComboBox;
        Li: TComboBox;
        Label2: TLabel;
        Li2: TComboBox;
        Label3: TLabel;
    end;

```

```

Lf: TComboBox;
Button1: TButton;
Label6: TLabel;
Button2: TButton;
RvSystem1: TRvSystem;
Label7: TLabel;
Label8: TLabel;
Label9: TLabel;
Bt_cancel: TButton;
procedure Bt_MatrizClick(Sender: TObject);
procedure Edit1KeyPress(Sender: TObject; var Key: Char);
procedure Grid1KeyPress(Sender: TObject; var Key: Char);
procedure LimpaGrid;
procedure Edit2KeyPress(Sender: TObject; var Key: Char);
procedure FormCreate(Sender: TObject);
procedure SpeedButton1Click(Sender: TObject);
procedure Bt_GaussClick(Sender: TObject);
procedure FormMouseDown(Sender: TObject; Shift: TShiftState;
  MousePos: TPoint; var Handled: Boolean);
procedure FormMouseWheelUp(Sender: TObject; Shift: TShiftState;
  MousePos: TPoint; var Handled: Boolean);
procedure Bt_SomaClick(Sender: TObject);
procedure Bt_MultClick(Sender: TObject);
procedure Edit3KeyPress(Sender: TObject; var Key: Char);
procedure Button1Click(Sender: TObject);
procedure Button2Click(Sender: TObject);
procedure RvSystem1Print(Sender: TObject);
procedure RvSystem1PrintHeader(Sender: TObject);
procedure Edit1Exit(Sender: TObject);
procedure RvSystem1OverridePreview(ReportSystem: TRvSystem;
  OverrideMode: TOverrideMode; var OverrideForm: TForm);
procedure RvSystem1OverrideSetup(ReportSystem: TRvSystem;
  OverrideMode: TOverrideMode; var OverrideForm: TForm);
procedure RvSystem1OverrideStatus(ReportSystem: TRvSystem;
  OverrideMode: TOverrideMode; var OverrideForm: TForm);
procedure Bt_cancelClick(Sender: TObject);

private
  { Private declarations }
public
  { Public declarations }
end;

var
  F_Trn_Determinante: TF_Trn_Determinante;
  Mt : Mtz;

implementation

{$R *.dfm}

procedure TF_Trn_Determinante.LimpaGrid;
var
  i, j : integer;
begin
  for i:= 0 to Grid1.ColCount -1 do
    for j:= 0 to Grid1.ColCount -1 do
      Grid1.Cells[i, j]:='';
    SetLength(M,0,0);
    Grid1.RowCount:= 1; //linhas do grid
    Grid1.ColCount:= 1; //colunas do grid
  end;

procedure TF_Trn_Determinante.Bt_MatrizClick(Sender: TObject);
var
  x : integer;
begin
  if Edit1.text = '' then Exit;
  LimpaGrid;
  x:= StrToInt(Edit1.text);
  Grid1.RowCount:= x; {linhas do grid}
  Grid1.ColCount:= x; {colunas do grid}
  Grid1.SetFocus;

```

```

end;

procedure TF_Trn_Determinante.Edit1KeyPress(Sender: TObject;
  var Key: Char);
begin
  if not (Ord(key) in [8,13,48..57]) then Key:= #0;
end;

procedure TF_Trn_Determinante.Grid1KeyPress(Sender: TObject;
  var Key: Char);
begin
  if not (Ord(key) in [8,13,44,45,48..57]) then Key:= #0;
end;

procedure TF_Trn_Determinante.Edit2KeyPress(Sender: TObject;
  var Key: Char);
begin
  if not (Ord(key) in [3,8,13,22,44,45,48..57,59]) then Key:= #0;
  {3 CTR+C e 22 CTR+V}
end;

procedure TF_Trn_Determinante.FormCreate(Sender: TObject);
begin
  Grid1.OnDrawCell:= TMenuActions.SGDrawCell;
  Grid1.OnKeyPress:= TMenuActions.GridKeyPress;
  MSeq:= false;
  EqLinear:= false;
end;

procedure TF_Trn_Determinante.SpeedButton1Click(Sender: TObject);
var
  i,x,iNum: Integer;
  List_A: TStringList;
begin
  List_A:= TStringList.Create;
  x:= StrToInt(Edit1.text);
  x:= x*x -1;
  for i:=0 to x do
  begin
    Randomize;
    iNum := Random(11);
    List_A.Add(IntToStr(iNum));
  end;
  {Verifica a quantidade de elementos}
  if List_A.Count <> Grid1.RowCount*Grid1.ColCount then
  begin
    ShowMessage('A_Quantidade_de_elementos_' +IntToStr(List_A.Count)
      +'_não_preenche_todas_as_células_da_Matriz!');
    Exit;
  end;
  {Mostra valores da lista no grid}
  MostraListGrid(List_A,Grid1);
end;

procedure TF_Trn_Determinante.Bt_GaussClick(Sender: TObject);
Var
  d,i,j: Integer;
  Aux: real;
begin
  if GridOk(Grid1) = false then exit;
  Cancelar:= false;
  Bt_cancel.Visible:= true;
  {Matriz de dimensão 1}
  if Grid1.ColCount = 1 then
  begin
    Pannel.Caption:= Grid1.Cells[0,0];
    Exit;
  end;
  d:= StrToInt(Edit1.text);
  {Adiciona linhas aos combos}
  CBox1.Items.Clear;
  CBox3.Items.Clear;
  CBox4.Items.Clear;

```

```

Li.Items.Clear;
Li2.Items.Clear;
Lf.Items.Clear;
Pivot1.Items.Clear;
Pivot2.Items.Clear;
for i:= 1 to d do
begin
  {Verifica cancelamento}
  Application.ProcessMessages;
  if Cancelar then
  begin
    SB.DestroyComponents;
    Pannel.Caption:='';
    Exit;
  end;
  CBox1.Items.Add('Linha_' + IntToStr(i));
  CBox3.Items.Add('Linha_' + IntToStr(i));
  CBox4.Items.Add('Linha_' + IntToStr(i));
  Li.Items.Add('Linha_' + IntToStr(i));
  Li2.Items.Add('Linha_' + IntToStr(i));
  Lf.Items.Add('Linha_' + IntToStr(i));
end;
{Preenche os valores do grid nas Matrizes}
Col_Tp:= d;
SetLength(M,d,d+1);
SetLength(Mt,d,d+1);
for i:= 0 to d-1 do
  for j:= 0 to d-1 do
  begin
    {Verifica cancelamento}
    Application.ProcessMessages;
    if Cancelar then
    begin
      SB.DestroyComponents;
      Pannel.Caption:='';
      Exit;
    end;
    M[i,j]:= StrToFloat(Grid1.Cells[j,i]);
    Mt[i,j]:= StrToFloat(Grid1.Cells[j,i]);
    Pivot1.Items.Add(IntToStr(j+1) + ',' + IntToStr(i+1));
    Pivot2.Items.Add(IntToStr(j+1) + ',' + IntToStr(i+1));
  end;
  {Verifica linha coluna zerada}
  if LCZerada(M) then
  begin
    Pannel.Caption:= '0';
    exit;
  end;
  SB.DestroyComponents;
  SetLength(SG,0);
  Tops:= 5;
  CriaGridBox(SB,M);
  {Escalona a Matriz Gauss}
  EscalonaMatriz(SB,M);
  {Verifica cancelamento}
  Application.ProcessMessages;
  if Cancelar then
  begin
    SB.DestroyComponents;
    Pannel.Caption:='';
    Exit;
  end;
  {Calcula Variáveis}
  Aux:=1;
  for i := 0 to length(M)-1 do
    for j := 0 to length(M)-1 do
      if i = j then
        Aux:= Aux*M[i,j];
    Pannel.Caption:= Fracao(Aux);
    Pannel1.Caption:= '';
    Bt_cancel.Visible:= false;
  end;
end;

```

```

procedure TF_Trn_Determinante.FormMouseWheelDown(Sender: TObject;
  Shift: TShiftState; MousePos: TPoint; var Handled: Boolean);
begin
  with SB.VertScrollBar do
    begin
      Position := Position + Increment;
    end;
  end;
end;

procedure TF_Trn_Determinante.FormMouseWheelUp(Sender: TObject;
  Shift: TShiftState; MousePos: TPoint; var Handled: Boolean);
begin
  with SB.VertScrollBar do
    begin
      Position := Position - Increment;
    end;
  end;
end;

procedure TF_Trn_Determinante.Bt_SomaClick(Sender: TObject);
Var
  i, j, d, L1, L2 : integer;
  S : String;
  Aux: real;
begin
  if (CBox1.ItemIndex = -1) or (CBox2.ItemIndex = -1) or
    (CBox3.ItemIndex = -1) then
    Exit;
  SendMessage(SB.Handle, WM_VSCROLL, SB_TOP, 0);
  d:= StrToInt(Edit1.text);
  L1:= CBox1.ItemIndex;
  L2:= CBox3.ItemIndex;
  for i:= 0 to d-1 do
    if CBox2.ItemIndex = 0 then
      begin
        Mt[L1, i] := Mt[L1, i] + Mt[L2, i];
        S := '⬇+⬇';
      end
    else
      begin
        Mt[L1, i] := Mt[L1, i] - Mt[L2, i];
        S := '⬇-⬇';
      end;
  end;
  CriaGridTex(SB, 'L'+IntToStr(CBox1.ItemIndex+1)+'⬇=⬇L'
    +IntToStr(CBox1.ItemIndex+1)+
    S+'L'+IntToStr(CBox3.ItemIndex+1));
  CriaGridBox(SB, Mt);
  {M recebe valores de Mt}
  for i:= 0 to d-1 do
    for j:= 0 to d-1 do
      M[i, j] := Mt[i, j];
  {Escalona a Matriz Gauss}
  EscalonaMatriz(SB, M);
  {Calcula Variáveis}
  Aux:=1;
  for i := 0 to length(M)-1 do
    for j := 0 to length(M)-1 do
      if i = j then
        Aux:= Aux*M[i, j];
  Panell.Caption:= Fracao(Aux);
  if Panell.Caption = Painel.Caption then
    Panell.Font.Color:= clGreen
  else
    Panell.Font.Color:= clRed;
  SendMessage(SB.Handle, WM_VSCROLL, SB_BOTTOM, 0);
end;

procedure TF_Trn_Determinante.Bt_MultClick(Sender: TObject);
Var
  i, j, d, L1 : integer;
  S : String;
  Aux: real;
begin
  if (CBox4.ItemIndex = -1) or (CBox5.ItemIndex = -1) or
    (trim(Edit3.Text) = '') then

```

```

Exit;
SendMessage(SB.Handle, WM_VSCROLL, SB_TOP, 0);
d:= StrToInt(Edit1.text);
L1:= CBox4.ItemIndex;
for i:= 0 to d-1 do
  if CBox5.ItemIndex = 0 then
  begin
    Mt[L1,i]:= Mt[L1,i]*StrToFloat(Edit3.Text);
    S:= ' * ' ;
  end
  else
  begin
    Mt[L1,i]:= Mt[L1,i]/StrToFloat(Edit3.Text);
    S:= ' / ' ;
  end;
CriaGridTex(SB, 'L'+IntToStr(CBox4.ItemIndex+1)+' = L'
+IntToStr(CBox4.ItemIndex+1)+
S+Edit3.Text);
CriaGridBox(SB,Mt);
{M recebe valores de Mt}
for i:= 0 to d-1 do
  for j:= 0 to d-1 do
    M[i,j]:= Mt[i,j];
  {Escalona a Matriz Gauss}
EscalonaMatriz(SB,M);
{Calcula Variáveis}
Aux:=1;
for i := 0 to length(M)-1 do
  for j := 0 to length(M)-1 do
    if i = j then
      Aux:= Aux*M[i,j];
  Panel1.Caption:= Fracao(Aux);
  if Panel1.Caption = Panel.Caption then
    Panel1.Font.Color:= clGreen
  else
    Panel1.Font.Color:= clRed;
  SendMessage(SB.Handle, WM_VSCROLL, SB_BOTTOM, 0);
end;

procedure TF_Trn_Determinante.Edit3KeyPress(Sender: TObject;
  var Key: Char);
begin
  if (key = '-') and (length(TEdit(Sender).Text) > 1) then
  begin
    key:=#0;
    exit;
  end;
  if (key = ',') and (pos(',',TEdit(Sender).Text) > 0) then
  begin
    key:=#0;
    exit;
  end;
  if not (Ord(key) in [8,13,44,45,48..57]) then Key:= #0;
end;

procedure TF_Trn_Determinante.Button1Click(Sender: TObject);
Var
  i,j,d,L1,L2,L3,i1,i2,j1,j2 : integer;
  Pivot, Aux: real;
  lista : TStringList;
begin
  if (Pivot1.ItemIndex = -1) or (Pivot1.ItemIndex = -1)
  or (Li.ItemIndex = -1) or (Li2.ItemIndex = -1)
  or (Lf.ItemIndex = -1) then
    Exit;
  SendMessage(SB.Handle, WM_VSCROLL, SB_TOP, 0);
  d:= StrToInt(Edit1.text);
  L1:= Li.ItemIndex;
  L2:= Li2.ItemIndex;
  L3:= Lf.ItemIndex;
  lista:= ArraydeString(Pivot1.Text,',');
  i1:= StrToInt(lista[0])-1;
  j1:= StrToInt(lista[1])-1;

```

```

lista:= ArraydeString(Pivot2.Text,',');
i2:= StrToInt(lista[0])-1;
j2:= StrToInt(lista[1])-1;
Pivot:= Mt[i1,j1]/Mt[i2,j2];
CriaGridTex(SB,'L'+IntToStr(L1+1)+'_L'+IntToStr(L2+1)+
'_a'+Pivot1.Text+'_a'+Pivot2.Text+'')*_L'+IntToStr(L3+1));
for j:= 0 to d-1 do
  Mt[L1,j]:= Mt[L2,j]-(Pivot * Mt[L3,j]);
CriaGridBox(SB,Mt);
{M recebe valores de Mt}
for i:= 0 to d-1 do
  for j:= 0 to d-1 do
    M[i,j]:= Mt[i,j];
{Escalona a Matriz Gauss}
EscalonaMatriz(SB,M);
{Calcula Variáveis}
Aux:=1;
for i := 0 to length(M)-1 do
  for j := 0 to length(M)-1 do
    if i = j then
      Aux:= Aux*M[i,j];
Panel1.Caption:= Fracao(Aux);
if Panel1.Caption = Panel.Caption then
  Panel1.Font.Color:= clGreen
else
  Panel1.Font.Color:= clRed;
SendMessage(SB.Handle, WM_VSCROLL, SB_BOTTOM, 0);
end;

procedure TF_Trn_Determinante.Button2Click(Sender: TObject);
begin
if SB.ComponentCount = 0 then Exit;
with RVSystem1.SystemPrinter do
begin
  MarginRight:= 0.2;
  MarginLeft:= 0.2;
  MarginTop:= 1;
  MarginBottom:= 1;
  rpDev.Copies:= 1;
  Copies:= rpDev.Copies;
  rpDev.Orientation:= poPortrait;
  Orientation:= rpDev.Orientation;
  RVSystem1.SystemPreview.FormState:= wsMaximized;
  RVSystem1.SystemSetups:= RVSystem1.SystemSetups - [ssAllowSetup];
  rpDev.SelectPaper('A4',false);
  RVSystem1.Execute;
end;
end;

procedure TF_Trn_Determinante.RvSystem1Print(Sender: TObject);
Var
  x,i,j: integer;
  TSG : TstringGrid;
begin
with RVSystem1.BaseReport do
begin
  for x:= 0 to SB.ComponentCount-1 do
  begin
    TSG:= TStringGrid(SB.Components[x]);
    for i:= 0 to TSG.RowCount - 1 do
    begin
      for j:= 0 to TSG.ColCount - 1 do
      begin
        if TSG.ColCount = 1 then
          PrintLeft(TSG.Cells[j,i],MarginLeft)
        else
          PrintCenter(TSG.Cells[j,i],MarginLeft+0.2+(j/2));
        end;
        NewLine;
        NewLine;
        if LinesLeft<1 then NewPage;
      end;
    end;
  end;
end;
end;

```

```

    end;
end;

procedure TF_Trn_Determinante.RvSystem1PrintHeader(Sender: TObject);
begin
    with RVSystem1.BaseReport do
    begin
        SetFont('Verdana',12);
        Bold := True;
        PrintCenter('Treinamento_Escalonamento_-_Gauss',PageWidth/2);
        NewLine;
        Canvas.Pen.Width := 2;
        Canvas.Pen.Color := clBlack;
        MoveTo(MarginLeft,YPos);
        LineTo(PageWidth-MarginRight,YPos);
        NewLine;
        SetFont('Verdana',8);
        Bold := False;
    end;
end;

procedure TF_Trn_Determinante.Edit1Exit(Sender: TObject);
begin
    Bt_MatrizClick(Bt_Matriz);
end;

procedure TF_Trn_Determinante.RvSystem1OverridePreview(ReportSystem:
TRvSystem; OverrideMode: TOverrideMode; var OverrideForm: TForm);
begin
    ExecutePreviewForm(Self,ReportSystem,OverrideMode,OverrideForm);
end;

procedure TF_Trn_Determinante.RvSystem1OverrideSetup(ReportSystem:
TRvSystem; OverrideMode: TOverrideMode; var OverrideForm: TForm);
begin
    ExecuteSetupForm(Self,ReportSystem,OverrideMode,OverrideForm);
end;

procedure TF_Trn_Determinante.RvSystem1OverrideStatus(ReportSystem:
TRvSystem; OverrideMode: TOverrideMode; var OverrideForm: TForm);
begin
    ExecuteStatusForm(Self,ReportSystem,OverrideMode,OverrideForm);
end;

procedure TF_Trn_Determinante.Bt_cancelClick(Sender: TObject);
begin
    Cancelar:= true;
    Bt_cancel.Visible:= false;
end;

end.

=====

unit Funcoes;

interface

uses
    Messages, Classes, SysUtils, Grids, Forms, StdCtrls, Graphics,
    Windows, Math, Controls, Dialogs, Tlhelp32;

type
    TMenuActions = class
    public
        {01a}class procedure SGDDrawCell(Sender: TObject; ACol, ARow: Integer;
            Rect: TRect; State: TGridDrawState);
        {01b}class procedure GridKeyPress(Sender: TObject; var Key: Char);
    end;

type Mtz = array of array of Real; {Matriz sem dimensão pré-definida}
    Mtz1 = array of Real; {Matriz de uma dimensão}
    {02}function ArrayDeString(s:string; separador: char): TStringList;

```

```

{03}function MatrizMenor(M:Mtz; Linha,Coluna,Dim:integer):Mtz;
{04}function Determinante(SB: TScrollBar; M:Mtz):Real;
{05}function Variaveis(M:Mtz; Dim:integer):Mtz1;
{06}function Fracao(Valor: real):string;
{07}function GridOk(Grid: TStringGrid):Boolean;
{08}function LCZerada(M:Mtz): Boolean;
{09}procedure PivoZ(SB: TScrollBar; M:Mtz; Ni:integer);
{10}procedure EscalonaMatriz(SB: TScrollBar; M:Mtz);
{11}procedure MostraMatrizGrid(M:Mtz; Grid:TStringGrid);
{12}procedure MostraListGrid(List:TStringList; Grid:TStringGrid);
{13}procedure CriaGridBox(SB: TScrollBar; M:Mtz);
{14}procedure CriaGridTex(SB: TScrollBar; Tx: String);
{15}function ProcessoExiste(ExeFileName: string): boolean;

var
  M : Mtz; {matrizes temporarias}
  SG : array of TStringGrid; {grids da area azul}
  Tops : Integer; {topo da area azul}
  Mseq : Boolean; {escrever sequencia}
  Col_Tp : Integer; {colunas do grid}
  Cancelar : Boolean; {cancelar operações}
  EqLinear : Boolean; {equações lineares}

implementation

{01a Altera largura das celulas e do grid e centraliza texto}
class procedure TMenuActions.SGDDrawCell(Sender: TObject; ACol,
  ARow: Integer; Rect: TRect; State: TGridDrawState);
begin
  with Sender as TStringGrid do
  begin
    {Altera a cor do foco}
    Canvas.Brush.Color := Color;
    Canvas.Font.Color:= Font.Color;
    {Modificar coluna nas eq lineares}
    if EqLinear and (Col_Tp <> 0) and (ACOL = Col_Tp-1) then
    begin
      canvas.Brush.color := $00FFF1D5;// $00C08000;
      canvas.Font.Color := clBLACK;
      canvas.Font.Name := 'TAHOMA';
      canvas.FillRect(Rect);
      canvas.TextOut(RECT.LEFT + 3,RECT.TOP,Cells[ACOL,AROW]);
    end;
    {Aumenta o tamanho da celula}
    if Canvas.TextWidth(Cells[ACol,ARow]) > DefaultColWidth then
      DefaultColWidth:= Canvas.TextWidth(Cells[ACol,ARow]);
    {Centraliza o conteudo da celula}
    Canvas.FillRect(Rect);
    DrawText(Canvas.Handle,PChar(Cells[ACol,ARow]),-1,Rect,DT_CENTER);
    {Aumenta a Largura do grid}
    if (Name <> 'Grid1') and (Name <> 'Grid2') then
      Width:= ColCount*DefaultColWidth+5;
    end;
  end;
end;

{01b}
class procedure TMenuActions.GridKeyPress(Sender: TObject; var Key: Char);
begin
  if Key = #13 then
  begin
    Key := #0;
    TStringGrid(Sender).Perform(WM_KeyDown,VK_Tab,0);
  end;
  if not (Ord(Key) in [3,8,13,22,44,45,48..57,59]) then Key:= #0;
end;

{02 Função para pegar os valores que estao entre separadores}
function ArrayDeString(s: string; separador: char): TStringList;
var
  k: integer; //posição do separador
  VS: TStringList; //vetor de string (resultado)
  Aux: string; //substring (um valor extraido da string)
begin

```

```

VS:= TStringList.Create;
k:= pos(separador, s);
while (k <> 0) do           //pega posição do separador enquanto a
                           //cadeia "s" for diferente de nulo
begin
  Aux := copy(s, 1, k-1); //sAux recebe os valores copiados de "s"
  delete(s, 1, k);       //apaga o valor extraído da string "s"
  VS.Add(Aux);           // adiciona os valores extraídos no vetor Vr
  k:= pos(separador, s); //pega posição do separador
end;
VS.Add(s);
result := VS;
end;

{03 Função para pegar a matriz com uma dimensão a menos}
function MatrizMenor(M:Mtz; Linha,Coluna,Dim:integer):Mtz;
var
  i,j,k :integer;
  M1     :Mtz;
begin
  k:=1;
  SetLength(M1,Dim-1,Dim-1); //cria uma matriz com uma dimensão menor
  Col_Tp:= dim-1;
  for i:=0 to Dim-1 do
    for j:=0 to Dim-1 do
      if ((i<>Linha) and (j<>Coluna)) then //se for diferente da linha/coluna
        begin
          if k mod Dim = 0 then
            k:=k+1;
            M1[(k div Dim), (k mod Dim)-1]:=M[i, j];
            k:=k+1;
          end;
        end;
      result:=M1;
    end;
  end;

{04 Função para calcular o determinante laplace}
function Determinante(SB: TScrollBar; M:Mtz):Real;
var
  M1: Mtz;
  i,d: integer;
  temp: Real;
begin
  d:= length(M);
  {determinante de matriz 2x2}
  if d=2 then
    result:=(M[0,0]*M[1,1])-(M[0,1]*M[1,0])
  else
    {dimensão maior que 2}
    begin
      temp:=0;
      for i:=0 to d-1 do
        begin
          Application.ProcessMessages;
          if Cancelar then Exit;
          M1:=MatrizMenor(M,i,0,d);
          if Mseq then CriarGridTex(SB, '(-1)^( '+IntToStr(i+1)+' +1) *a[' +
            IntToStr(i+1)+' ,1]*');
          if Mseq then CriarGridBox(SB,M1);
          if (i mod 2 = 0) then
            temp:= temp + M[i,0]*Determinante(SB,M1)
          else
            temp:= temp - M[i,0]*Determinante(SB,M1);
          end;
        end;
      result:= temp;
    end;
  end;

{05 Função para calcular variáveis de um sistema linear}
function Variaveis(M:Mtz; Dim:integer):Mtz1;
var
  x,i,j,k :integer;
  Soma: Real;
  Resp: Mtz1;

```

```

begin
  SetLength(Resp, Dim);
  k:= 0;
  x:= Dim;
  Resp[k]:= M[x-1][x]/M[x-1][x-1];
  for i := x-2 downto 0 do
  begin
    Soma:= 0;
    for j:= k+1 downto 1 do
      Soma:= Soma + M[i][x-j]*Resp[j-1];
    Inc(k);
    Resp[k] := (M[i][x] - Soma)/M[i][i];
    Result:= Resp;
  end;
end;

{06 Transforma decimal em fração com precisão de 4 casas decimais}
function Fracao(Valor: real):string;
var
  i: integer;
  x: real;
  k: Currency;
begin
  x:= Valor;
  if frac(x) = 0 then
  begin
    result:= FloatToStr(x);
    exit;
  end;
  i:= 1;
  while 1 < 2 do
  begin
    k:= x;
    if frac(k) = 0 then
    begin
      if i = 1 then
        result:= FloatToStr(k)
      else
        result:= FloatToStr(k)+'/'+IntToStr(i);
      exit;
    end;
    x:= x+Valor;
    inc(i);
  end;
end;

{07 Verifica o grid antes de continuar}
function GridOK(Grid: TStringGrid):boolean;
var
  i, j, n : integer;
  S : String;
begin
  result:= true;
  for i:= 0 to Grid.RowCount -1 do
    for j:= 0 to Grid.ColCount -1 do
      begin
        {Celulas em branco}
        if Grid.Cells[j,i] = '' then
          begin
            ShowMessage('A_Matriz_não_pode_celulas_em_branco!');
            result:= false;
            exit;
          end;
        {Celulas com mais de um sinal -}
        n:=0;
        S:= Grid.Cells[j,i];
        while Pos('-', S) > 0 do
          begin
            S[Pos('-', S)]:= '0';
            n:= n+1;
          end;
        if n > 1 then
          begin

```

```

    ShowMessage(
      'A_Matriz_não_pode_ter_elementos_com_mais_de_um_sinal_negativo!');
    result:= false;
    exit;
  end;
  S:= Grid.Cells[j,i];
  if (n = 1) and (S[1] <> '-') then
  begin
    ShowMessage(
      'A_Matriz_não_pode_ter_elementos_com_um_sinal_negativo_deslocado!');
    result:= false;
    exit;
  end;
end;
end;

{08 Verifica linha ou coluna zerada}
function LCZerada(M:Mtz): boolean;
var
  i,j,d :integer;
  LCz : Boolean;
begin
  d:= Length(M);
  {linha}
  for i:= 0 to d-1 do
  begin
    LCz:=true;
    for j:= 0 to d-1 do
      if M[i,j] <> 0 then LCz:= false;
    if LCz = true then
    begin
      result:= true;
      exit;
    end;
  end;
  end;
  {coluna}
  for j:= 0 to d-1 do
  begin
    LCz:=true;
    for i:= 0 to d-1 do
      if M[i,j] <> 0 then LCz:= false;
    if LCz = true then
    begin
      result:= true;
      exit;
    end;
  end;
  end;
  result:= false;
end;

{09 Verifica PIVô zerado}
procedure PivoZ(SB: TScrollBar; M:Mtz; Ni:integer);
var
  i,d,k :integer;
  cont : Boolean;
begin
  d:= Length(M);
  k:= Ni;
  cont:= true;
  while cont do
  begin
    if M[Ni,Ni]=0 then {se pivo for zero}
    begin
      Inc(k); {proxima linha}
      if k = Ni then Inc(k); {caso mesma linha vai p/ proxima linha}
      if k = d then k:=0; {fim da matriz vai p/ primeira linha}
      for i:= 0 to d-1 do
        M[Ni,i]:= M[Ni,i]+M[k,i]; {soma linhas}
      if Mseq then CriaGridTex(SB,'L'+IntToStr(Ni+1)+'_L'+IntToStr(Ni+1)
        +'_L'+IntToStr(k+1));
    end
  else
    cont:= false;
  end;
end;

```

```

    end;
end;

{10 Função para escalonar a matriz e mostra}
procedure EscalonaMatriz(SB: TScrollBar; M: Mtz);
var
    i, j, k, d : integer;
    Mult: Real;
begin
    d:= length(M);
    for k:= 0 to d-2 do
    begin
        for i:= 1 to d-1-k do
        begin
            {Verifica cancelamento}
            Application.ProcessMessages;
            if Cancelar then Exit;
            {Pivo zerado}
            if M[k,k] = 0 then
            begin
                PivoZ(SB,M,k);
                if Mseq then CriaGridBox(SB,M);
            end;
            Mult:= M[i+k,k];
            if Mseq then CriaGridTex(SB, 'L'+IntToStr(i+k+1)+'_L'+IntToStr(i+k+1)
                +'_L(a['+IntToStr(i+k+1)'+','+IntToStr(k+1)
                +']]/a['+IntToStr(k+1)'+','+IntToStr(k+1)'+']) *L'+IntToStr(k+1));
            for j := 0 to d-k do
                M[i+k,j+k] := ((M[i+k,j+k]*M[k,k])-(M[k,j+k]*Mult))/M[k,k];
            end;
            if Mseq then CriaGridBox(SB,M);
        end;
    end;
end;

{11 Mostra matriz em grid}
procedure MostraMatrizGrid(M: Mtz; Grid: TStringGrid);
var
    i, j : integer;
begin
    for i := 0 to Grid.RowCount - 1 do
        for j:= 0 to Grid.ColCount - 1 do
            if Grid.Name = 'Grid1' then
                Grid.Cells[i, j] := FloatToStr(M[j, i])
            else
                Grid.Cells[i, j] := Fracao(M[j, i]);
        end;
    end;
end;

{12 Mostra lista em grid}
procedure MostraListGrid(List: TStringList; Grid: TStringGrid);
var
    i, j, k : integer;
begin
    k:= 0;
    for i:= 0 to Grid.RowCount - 1 do
        for j:= 0 to Grid.ColCount - 1 do
            begin
                if Grid.Name = 'Grid1' then
                    Grid.Cells[j, i] := List.Strings[k]
                else
                    Grid.Cells[j, i] := Fracao(StrToFloat(List.Strings[k]));
                Inc(k);
            end;
        end;
    end;
end;

{13 Cria varios grids com a sequencia da resolução}
procedure CriaGridBox(SB: TScrollBar; M: Mtz);
var
    i, j, s : integer;
begin
    s:= SB.ComponentCount;
    SetLength(SG, Length(SG)+1);
    if SG[s] = nil then SG[s]:= TStringGrid.Create(SB);
    SG[s].Parent := SB;
end;

```

```

SG[s].OnDrawCell:= TMenuActions.SGDrawCell;
if s = 0 then
  SG[s].Font.Color:= clBlue
else
  SG[s].Font.Color:= clRed;
SG[s].Font.Height:= -16;
SG[s].Font.Name:= 'MS_Sans_Serif';
SG[s].Font.Style:= [fsBold];
SG[s].FixedCols:=0;
SG[s].FixedRows:=0;
SG[s].DefaultColWidth:= 24;
SG[s].DefaultRowHeight:=24;
SG[s].Ctl3D:= false;
SG[s].ScrollBars:= ssNone;
SG[s].Enabled:= false;
SG[s].Left:= 5;
SG[s].RowCount:= Length(M);
SG[s].ColCount:= Col_Tp; //Length(M[0]); //colunas
SG[s].Height:= SG[s].DefaultRowHeight*SG[s].RowCount+Length(M);
SG[s].Top:= Tops;
for i:= 0 to SG[s].colCount - 1 do
  for j:= 0 to SG[s].rowCount - 1 do
    SG[s].Cells[i,j]:= Fracao(M[j,i]);
Tops:= Tops+SG[s].Height+5;
SG[s].Show;
end;

{14 Cria varios grids com texto}
procedure CriaGridTex(SB: TScrollBar; Tx: String);
var
  s :integer;
begin
  s:= SB.ComponentCount;
  SetLength(SG,Length(SG)+1);
  if SG[s] = nil then SG[s]:= TStringGrid.Create(SB);
  SG[s].Parent := SB;
  SG[s].OnDrawCell:= TMenuActions.SGDrawCell;
  SG[s].Font.Color:= clBlack;
  SG[s].Font.Height:= -16;
  SG[s].Font.Name:= 'MS_Sans_Serif';
  SG[s].Font.Style:= [fsBold];
  SG[s].FixedCols:=0;
  SG[s].FixedRows:=0;
  SG[s].GridLineWidth:=0;
  SG[s].Ctl3D:= false;
  SG[s].ScrollBars:= ssNone;
  SG[s].Enabled:= false;
  SG[s].Left:= 5;
  SG[s].RowCount:= 1;
  SG[s].ColCount:= 1;
  SG[s].Top:= Tops;
  SG[s].Height:= SG[s].DefaultRowHeight*SG[s].RowCount;
  SG[s].Cells[0,0]:= Tx;
  Tops:= Tops+SG[s].Height+5;
  SG[s].Show;
end;

{15 Verifica se o processo está ativo}
function ProcessoExiste(ExeFileName: string): boolean;
const
  PROCESS_TERMINATE=$0001;
var
  ContinueLoop: BOOL;
  FSnapshotHandle: THandle;
  FProcessEntry32: TProcessEntry32{declarar Uses Tlhelp32};
begin
  result := false;
  FSnapshotHandle := CreateToolhelp32Snapshot(TH32CS_SNAPPROCESS, 0);
  FProcessEntry32.dwSize := Sizeof(FProcessEntry32);
  ContinueLoop := Process32First(FSnapshotHandle,FProcessEntry32);
  while integer(ContinueLoop) <> 0 do
  begin
    if ((UpperCase(ExtractFileName(FProcessEntry32.szExeFile)) =

```

```
    UpperCase(ExeFileName)
or (UpperCase(FProcessEntry32.szExeFile) = UpperCase(ExeFileName)) then
begin
    Result := true;
    exit;
end;
ContinueLoop := Process32Next(FSnapshotHandle, FProcessEntry32);
end;
CloseHandle(FSnapshotHandle);
end;
end.
```

Link's para download do aplicativo

- <http://www.mediafire.com/download/f6415rwm257p1lq/Matriz.rar>
- <https://drive.google.com/file/d/0B8d4RKfZIT3BaGpVNFAwWUwxSXM/view?pref=2&pli=1>