UMA FERRAMENTA VISUAL PARA A MODELAGEM DE TRÁFEGO URBANO COM SISTEMAS LINEARES

A VISUAL TOOL FOR URBAN TRAFFIC MODELING WITH LINEAR SYSTEMS

Ermeson José Ribeiro Ferreira*
Ricardo Lenz César**
George Ney Almeida Moreira***

RESUMO

Disciplinas matemáticas como Álgebra Linear são fundamentais em diversos cursos acadêmicos, oferecendo suporte conceitual para múltiplas aplicações. No entanto, é comum que estudantes iniciantes enfrentem dificuldades com aprendizado delas. Para amenizar esse desafio, este trabalho propõe uma ferramenta didática que aborda um problema concreto de Álgebra Linear de maneira visual e interativa. O contexto escolhido é o tráfego urbano, representado por meio de grafos — uma estrutura amplamente utilizada em cursos de computação — que dá origem a um sistema de equações lineares. A proposta inclui elementos visuais com animações e explicações contextualizadas, permitindo ao aluno compreender como o modelo em grafo se relaciona com o sistema algébrico. Além disso, a ferramenta oferece exemplos práticos e funcionalidades de edição, promovendo maior engajamento com o conteúdo. Dessa forma, o projeto contribui para tornar o aprendizado mais acessível e significativo, ao demonstrar como conceitos matemáticos podem ser aplicados na resolução de problemas reais do cotidiano.

Palavras-chave: Matemática. Álgebra Linear. Ensino. Tecnologia. Visualização.

ABSTRACT

Mathematics subjects such as Linear Algebra play a foundational role in various academic programs, supporting a wide range of applications. Despite their importance, many first-year students often struggle with these topics. To address this issue, this work introduces an educational tool that explores a real-world Linear Algebra problem through an interactive and visual approach. The chosen context is urban traffic, modeled using graphs—a structure commonly taught in

* Graduando em Ciência da Computação, Instituto Federal de Educação, Ciência e Tecnologia do Ceará - IFCE, Aracati, Ceará, Brasil. E-mail: joseenem35@gmail.com
** Mestre em Ciência da Computação pela Universidade Federal do Ceará — UFC, Docente do Instituto Federal de Educação, Ciência e Tecnologia do Ceará - IFCE, Aracati, Ceará, Brasil. E-mail: ricardo.lenz@ifce.edu.br
*** Mestre em Matemática pela Universidade Federal de Campina Grande - UFCG, Docente do Instituto Federal de Educação, Ciência e Tecnologia do Ceará - IFCE, Aracati, Ceará, Brasil. E-mail: george.almeida@ifce.edu.br

computer science—which leads to the formulation of a linear system of equations. The tool features animations, visual aids, and contextual explanations to help students understand the connection between the graph model and the associated algebraic system. It also includes editable examples to encourage hands-on engagement. By bridging abstract concepts and practical applications, the tool enhances student interaction with the material, supporting both teaching and learning while illustrating the real-world relevance of Linear Algebra.

Keywords: Mathematics. Linear Algebra. Teaching. Technology. Visualization.

1 INTRODUÇÃO

Na formação de base em vários cursos acadêmicos estão presentes disciplinas de teor matemático, que preparam o aluno com um arcabouço intelectual para a resolução de problemas aplicáveis em diversas áreas do conhecimento (ROMEIRO; GARCIA; ROMÃO, 2021). Entretanto, é frequente a dificuldade de muitos alunos iniciantes com certas disciplinas matemáticas na base dos cursos ingressados (AGUIAR, 2022). Com dificuldades de compreender os conceitos e de aplicá-los, alguns chegam a questionar sobre a própria natureza de tais conteúdos (ROMEIRO; GARCIA; ROMÃO, 2021; ANDRADE, 2013).

Contudo, diversos problemas mais elevados de raciocínio lógico, e que exigem maior nível de abstração, não podem ser devidamente apreciados sem certa fundamentação matemática de base. A importância de tal suporte não pode ser ignorada. Conforme Moraes (2023), um dos principais desafios para a aprendizagem da matemática consiste na ausência de nexo entre os conceitos abordados em sala de aula e as aplicações dela na vida cotidiana. Assim, torna-se fundamental incentivar iniciativas que mostrem como mesmo aspectos mais teóricos podem ser uma ferramenta poderosa para compreender e resolver problemas concretos.

A Álgebra Linear, em particular, presente na base de cursos da Ciência da Computação, tem um valor potencial significativo para os alunos, com aplicações em diversas áreas, chegando mesmo a superar a aplicabilidade de outros temas matemáticos a nível de graduação (LAY; LAY; MCDONALD, 2018).

Surge, assim, uma busca por ferramentas que possibilitem uma aplicação mais concreta de conceitos de uma área da matemática como a Álgebra Linear. Por tratar especialmente sobre a resolução de sistemas lineares, uma ferramenta de cunho mais didático, voltada para demonstrar esse tipo de problema, pode ser usada para enriquecer a experiência de alunos no contato inicial com a disciplina, ajudando a visualizar como tal conhecimento pode ser relevante e atual para lidar com questões diversas na sociedade (STEWART et al., 2022).

Neste contexto, este trabalho propõe o desenvolvimento de uma ferramenta que aborda, de forma visual e interativa, um problema prático frequentemente tratado na disciplina de Álgebra Linear. O problema em questão se refere ao tráfego urbano, modelado por meio de um grafo — uma estrutura essencial e amplamente utilizada em cursos de computação —, que gera um

sistema linear de equações necessário para a resolução da situação proposta. A implementação permite uma maior interação e aproximação entre o aluno e o conteúdo explicitado, oferecendo a possibilidade de o estudante explorar o conteúdo no próprio ritmo, facilitando, assim, o entendimento e mostrando como certos conceitos e recursos matemáticos explicam e permitem subsidiar compreensões precisas sobre o mundo ao redor.

Para o desenvolvimento da ferramenta proposta, adotou-se o ecossistema *web*, tendo em vista a abrangência, a flexibilidade e a facilidade do uso e do acesso oferecidos. Por meio da *web*, os alunos podem acessar a implementação de qualquer lugar, no computador ou no celular, e usar diretamente a ferramenta sem a necessidade de instalar ou de configurar *softwares* adicionais. A ferramenta apresenta o grafo com o problema em questão e uma interface visual para trabalhar com ele, ao mesmo tempo em que o aluno visualiza o sistema de equações lineares que é gerado e pode solicitar a resolução dele.

Além disso, para tornar o conteúdo mais atrativo, a ferramenta apresenta uma animação opcional de veículos, que fluem entre os vértices do grafo. Ademais, a fim de facilitar a interpretação de como o problema se reflete no sistema linear, ao selecionar um vértice no grafo, a equação correspondente a ele é destacada na exibição do sistema linear, ajudando o aluno a inferir melhor como ocorre essa relação. Por fim, a ferramenta implementada se encontra *online*, podendo ser usada por qualquer pessoa interessada no problema [1].

O presente trabalho está organizado da seguinte maneira: a Seção 2 apresenta a fundamentação teórica; a Seção 3 contém trabalhos relacionados à temática; a Seção 4 esclarece a metodologia utilizada para o desenvolvimento da solução proposta, assim como os detalhes da implementação; a Seção 5 apresenta os resultados obtidos com o presente trabalho. Por fim, a Seção 6 dispõe das conclusões finais e das perspectivas para os trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

Para os autores (LIPSCHUTZ; LIPSON, 2013), existe uma grande ligação entre a matemática e a ciência da computação, dado que os primeiros computadores surgiram para resolver problemas matemáticos complexos, embora muitas vezes isso não seja perceptível para estudantes de computação. O trabalho de Campbell-Kelly (2009) argumenta que isso se decorre do fato de que, com a evolução da computação, as camadas abstratas da matemática foram sendo ocultadas, não sendo tão visível a presença da disciplina no processo de criação dos diversos tipos de *software*.

Além disso, disciplinas matemáticas frequentemente são vistas como de maior desafio para muitos públicos. Perguntas feitas sobre a natureza ou finalidade de tal ciência são feitas não apenas por alunos do ensino básico, mas também por aqueles alunos no nível superior (COIMBRA et al., 2008). Dentro desse contexto, o ensino da Álgebra Linear, como uma teoria fundamental de base para inúmeras áreas, pode enfrentar dificuldades em exemplificar os

[1] A implementação atual está online em: https://traffic-control-six.vercel.app.

muitos elementos dela em um universo matemático que já é abstrato (COIMBRA et al., 2008; TEIXEIRA; FONTENELE, 2017; HRYNIEWICZ et al., 2024).

Frente a dificuldades encontradas no que tange à disciplina, e em paralelo à relevância dela para a formação dos alunos, o Grupo de Estudo do Currículo de Álgebra Linear (LACSG 2.0) propõe inovações metodológicas, incluindo o uso de tecnologias, muitas das quais se tornaram mais acessíveis para públicos maiores e podem ser bastante eficazes no aprendizado. Dentre as recomendações, destacam-se a antecipação do ensino de Álgebra Linear no currículo, com uma abordagem prática focando em como os conceitos são usados hoje, além da integração de ferramentas tecnológicas como GeoGebra e MATLAB, que facilitam o entendimento por meio da experimentação e visualização (ver mais adiante na Seção 3 a respeito dessas tecnologias). Essas abordagens visam não apenas aprimorar a compreensão dos conceitos, mas também preparar os estudantes para os desafios contemporâneos, como a análise de dados e a inteligência artificial, tornando a disciplina mais relevante e aplicada ao contexto em que se vive (STEWART et al., 2022).

Apesar da crescente presença de Tecnologias da Informação e Comunicação (TICs) no cotidiano dos graduandos, métodos tradicionais de ensino, como a memorização de fórmulas desconectadas da realidade, ainda prevalecem em muitas salas de aula (ROMEIRO; GARCIA; ROMÃO, 2021). Contudo, em paralelo, metodologias educacionais, especialmente as metodologias ativas, têm ganhado maior destaque. Essas abordagens buscam promover uma aprendizagem colaborativa e flexível, conectando teoria à prática por meio de ferramentas educacionais. Tais tecnologias não substituem os professores, mas os auxiliam como mediadores, oferecendo aos alunos uma oportunidade de aprendizado mais interativa, tanto dentro quanto fora da sala de aula (MORÁN, 2015).

Dado o foco do presente trabalho em uma implementação interativa de um problema de sistemas lineares para facilitar a aprendizagem na disciplina de Álgebra Linear, os conceitos envolvidos de grafos, e os detalhes específicos do problema abordado, são discutidos a seguir.

2.1 Grafos

Para esta seção serão vistos alguns fundamentos relacionados à Teoria dos Grafos com base em (CATARINO, 2025). Um grafo pode ser definido como um conjunto de vértices V e um conjunto de arestas E que conectam esses vértices. As arestas podem ou não ter orientação. Quando têm uma orientação, trata-se de um grafo direcionado (ou dígrafo), em contraste com grafos não direcionado. Em grafos direcionados, pode-se considerar que cada aresta é composta por um vértice de origem e um de destino.

Grafos podem ser ponderados ou não. Em um grafo ponderado, as arestas possuem um peso, o que é útil para definir o custo, o tempo de travessia ou o fluxo entre os vértices. No presente trabalho, esse peso será usado como fluxo.

Para representar grafos no computador, existem duas principais formas: através de listas de adjacência ou por matrizes de adjacência. Para grafos densos, a matriz de adjacência é

preferida, pois permite analisar rapidamente a conectividade entre os vértices. A matriz de adjacência é uma matriz quadrada onde cada elemento a_{ij} indica a conexão entre os vértices i e j.

Em grafos não ponderados, a matriz de adjacência tem $a_{ij} = 1$ onde há conexão (e 0 caso contrário). Em grafos ponderados, a matriz de adjacência contém os pesos das arestas:

$$\begin{bmatrix} 0 & 4 & 7 \\ 4 & 0 & 3 \\ 7 & 3 & 0 \end{bmatrix}$$

Em grafos direcionados, a matriz de adjacência não é simétrica, com o valor de a_{ij} indicando a presença de uma aresta direcionada de i para j com determinado peso:

$$\begin{bmatrix} 0 & 2 & 3 \\ 0 & 0 & 4 \\ 5 & 0 & 0 \end{bmatrix}$$

É possível adotar o princípio da conservação de fluxo em grafos direcionados; isso significa que o somatório dos pesos (ou fluxos) de arestas de entrada de um vértice deve ser igual ao somatório dos fluxos de saída. Isso é fundamental para problemas como redes de transporte ou circuitos elétricos. Assim, para cada vértice, as arestas que incidem sobre ele, e as arestas que saem dele, devem ter o mesmo somatório em relação aos pesos deles.

2.2 Modelagem do problema escolhido com Sistema Linear

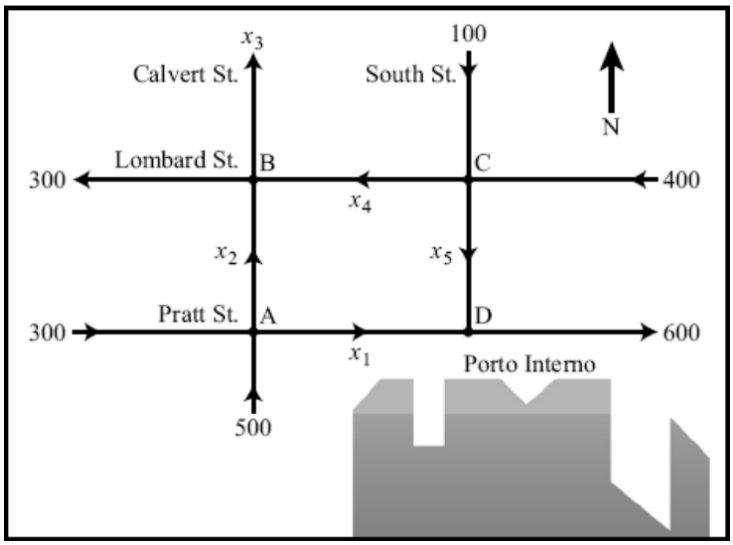
A Álgebra Linear é o ramo da matemática que estuda vetores e matrizes (STRANG, 2016), tendo como um dos problemas centrais a resolução de sistemas lineares, os quais podem ser representados por equações matriciais, ou vetoriais envolvendo combinação linear de vetores (LAY; LAY; MCDONALD, 2018).

Com a crescente importância da álgebra linear em inúmeras aplicações, há avanços que são conquistados com cada nova geração de hardware e software, buscando atender as demandas computacionais numéricas cada vez mais avançadas. Para ilustrar um problema concreto para alunos começando a ter contato com a disciplina, foi escolhido um problema de fácil entendimento e que ilustra bem como o uso de sistemas lineares pode ajudar em situações práticas.

O livro "Álgebra Linear e Suas Aplicações" de Lay, Lay e McDonald (2018) traz muitos problemas para aplicações de sistemas lineares. Um dos problemas apresentados é o de Fluxo de Tráfego. Ele consiste em determinar o padrão geral em um fluxo de rede, que representa um fluxo de tráfego de veículos por hora em diversas ruas de mão única, no centro da cidade de Baltimore durante uma tarde típica.

Para resolver o problema de determinar o fluxo de trânsito em vários locais desconhecidos, a solução consiste em escrever equações que descrevam o fluxo nos vários vértices (encontros de

Figura 1 – Ilustração do Fluxo de Rede



Fonte: (LAY; LAY; MCDONALD, 2018)

ruas), gerando um sistema de equações lineares, e, após isso, encontrar a solução geral do sistema. Na modelagem do problema, é preciso marcar as interseções das ruas e os fluxos desconhecidos, como ilustrado na figura abaixo. Além disso, em cada interseção é necessário igualar o fluxo de entrada com o de saída. Essa abordagem se ilustra pela seguinte figura:

Interseção	Fluxo de entrada		Fluxo de saída
A	$300 + 500$	=	$x_1 + x_2$
B	$x_2 + x_4$	=	$300 + x_3$
C	$100 + 400$	=	$x_4 + x_5$
D	$x_1 + x_5$	=	600

Fonte: (LAY; LAY; MCDONALD, 2018)

Com base na figura, pode-se verificar que o fluxo total que entra na rede

$$500 + 300 + 100 + 400$$

é igual ao fluxo total que sai dela

$$300 + x_3 + 600,$$

o que fornece $x_3 = 400$. Logo, torna-se necessário combinar essa equação com as quatro primeiras equações reorganizadas, a fim de obter o seguinte sistema de equações:

$$\begin{aligned} x_1 + x_2 &= 800 \\ x_2 - x_3 + x_4 &= 300 \\ x_4 + x_5 &= 500 \\ x_1 + x_5 &= 600 \\ x_3 &= 400 \end{aligned}$$

Escalonando a matriz aumentada associada, obtém-se:

$$\begin{aligned} x_1 \quad & +x_5 = 600 \\ x_2 \quad & -x_5 = 200 \\ & x_3 = 400 \\ & x_4 + x_5 = 500 \end{aligned}$$

Percebe-se que o padrão de fluxo geral para a rede é descrito por:

$$\begin{cases} x_1 = 600 - x_5 \\ x_2 = 200 + x_5 \\ x_3 = 400 \\ x_4 = 500 - x_5 \\ x_5 \text{ é livre} \end{cases}$$

Sendo assim, o autor conclui que o resultado de um fluxo negativo em um ramo da rede corresponde a um fluxo no sentido oposto do ilustrado no modelo, pois uma vez que as ruas nesse problema são de mão única, nenhuma das variáveis pode ser negativa. Além disso, percebe-se que esse fato leva a limitações nos valores possíveis das variáveis. Por exemplo, $x_5 \leq 500$, já que x_4 não pode ser negativa.

2.3 Resolução do Sistema Linear

Resolver sistemas lineares $Ax = b$ é um problema fundamental em diversas áreas da ciência e engenharia. Há uma variedade de métodos desenvolvidos para essa tarefa que reflete a diversidade de situações em que tais sistemas aparecem. Quando uma matriz A é quadrada e sem estruturas específicas conhecidas, pode-se adotar o método de eliminação de Gauss. Quando a matriz A não é quadrada, ou é muito grande e esparsa (cheia de zeros), ou exibe determinadas estruturas que podem ser exploradas de forma inteligente por algoritmos, outros métodos mais especializados são aplicáveis[2].

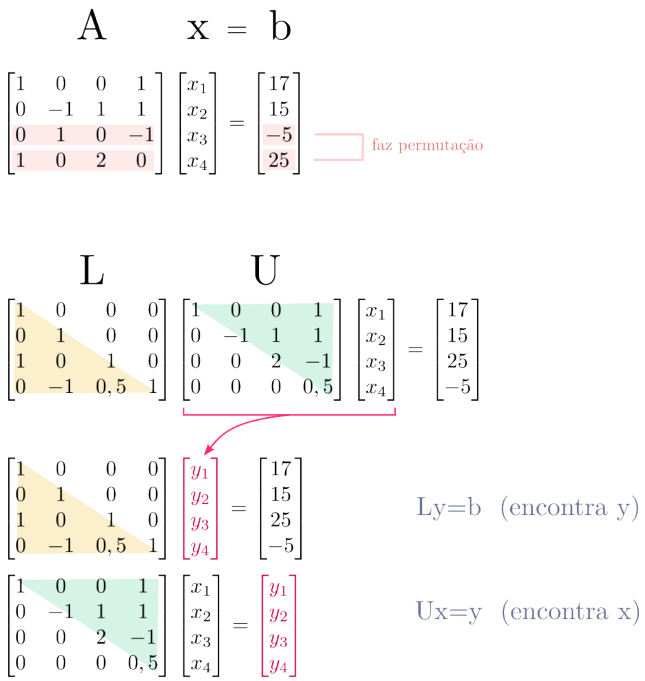
Por outro lado, resolver sistemas triangulares é uma operação relativamente direta. A ideia por trás da eliminação de Gauss é converter um sistema $Ax = b$ em outro sistema triangular equivalente (isto é, com a matriz A sendo triangular superior), por meio de etapas sucessivas que levam A a uma matriz triangular superior U. Conforme Golub e Loan (1996), a fatoração LU é uma descrição algébrica de "alto nível" da eliminação gaussiana, onde as etapas sucessivas que atuam em A são usadas também para produzir L, uma matriz triangular inferior, de modo que $A = LU$. Assim, o sistema original $Ax = b$ torna-se $LUx = b$, com L e U triangulares (ver

[2] Outros métodos incluem decomposição QR, Cholesky, ou ainda métodos iterativos como Jacobi, Gauss-Seidel, Gradientes Conjugados, etc. Ver Golub e Loan (1996) para um tratamento extenso disso.

Figura 2), sendo facilmente resolvidos: resolve-se primeiro $Ly = b$ (para achar y, usando *forward substitution*) e depois $Ux = y$ (para achar x, usando *backward substitution*).

Na prática, é comum o uso de permutações para melhorar a estabilidade numérica do algoritmo. Assim, em NumPy (cf. adiante na Seção 3 sobre essa tecnologia), por exemplo, a função[3] que resolve o sistema linear é tipicamente implementada usando a rotina gesv() do LAPACK. Essa rotina tem diversas variantes, como para trabalhar com valores reais usando float (32 bits), double (64 bits) ou ainda números complexos. O que ela faz é usar a fatoração LU com permutações e consegue, assim, resolver o sistema. Como o mais custoso é decompor A em LU, uma vez que essa fatoração tenha sido feita, é rápido resolver $LUx = b$ para o valor de b e obter x. De fato, a rotina gesv() permite receber uma matriz B de colunas b_i para resolver vários sistemas $LUx_1 = b_1$, $LUx_2 = b_2$, etc., aproveitando que já se tem L e U calculados, otimizando o processo para resolver múltiplos sistemas lineares.

Figura 2 – Fatoração LU



Fonte: Elaborada pelo autor.

2.4 Questões Tecnológicas

2.4.1 Tecnologias de base para o processamento numérico

Muitas pesquisas científicas atuais envolvem simulações numéricas em computadores, desde a astronomia à biologia; essas simulações executam problemas expressos em uma linguagem matemática que o computador entenda. Nesse processo, utiliza-se tecnologias e bibliotecas

[3] https://numpy.org/doc/stable/reference/generated/numpy.linalg.solve.html

de funções de referência, muitas das quais estão sendo amplamente testadas por muitos pesquisadores, e sendo ajustadas e refinadas com o tempo, tornando-se padrões bem estabelecidos.

De fato, a história de muitas tecnologias numéricas de base, envolvendo Álgebra Linear e outras disciplinas, está bastante entrelaçada com a própria evolução dos computadores. Diversos algoritmos fundamentais de álgebra linear, usados em software científicos, foram otimizados ao longo das últimas décadas para aproveitarem melhor os recursos do hardware, como a memória Cache, as instruções SIMD e as diversas técnicas de paralelismo.

Na verdade, as rotinas matemáticas fundamentais costumam ser algumas das primeiras instruções a serem remodeladas com novos avanços do hardware (DONGARRA, 2022). Esses avanços se dão de forma colaborativa, com a maior parte do software científico sendo de código aberto, gerando assim algoritmos de alta qualidade com testes rigorosos[4].

Entre alguns padrões bem estabelecidos em relação a tecnologias fundamentais para implementações numéricas, pode-se citar o padrão IEEE 754 de ponto flutuante, o padrão MPI (para a comunicação entre os processos em sistemas distribuídos) e o BLAS (*Basic Linear Algebra Subprograms*).

O BLAS é um conjunto de rotinas para lidar com vetores e matrizes, que se tornou um padrão seguido por muitos softwares, com uma implementação de referência feita originalmente em Fortran e várias outras implementações posteriores, otimizadas para diferentes arquiteturas. Esses padrões são muito importantes, sendo construídos com a experiência de diversas comunidades e favorecendo o reúso de conquistas anteriores (DONGARRA, 2022).

O BLAS fica em um grande repositório de softwares numéricos, em www.netlib.org. Nesse repositório é possível encontrar também uma outra conhecida tecnologia de base, o LAPACK[5], que é uma biblioteca robusta e profissional de rotinas numéricas de álgebra linear, bastante usada em vários outros softwares. Ele oferece rotinas para resolver sistemas lineares, fatorações de matriz (LU, Cholesky, QR, SVD, etc), dentre outras funções, permitindo trabalhar com números reais e complexos, com precisão simples ou dupla (*float* ou *double*).

O LAPACK é construído usando como base as rotinas fundamentais do BLAS. Assim, se uma implementação de BLAS é otimizada para uma determinada arquitetura, isso beneficia também o LAPACK. Ele foi feito originalmente em Fortran, mas há interface em C também para usá-lo. Assim como o BLAS, o LAPACK também é fruto de um esforço comunitário, tendo contribuições de pessoas de diferentes instituições[6].

A existência dessas bibliotecas e tecnologias de referência é muito importante para a área. Projetos assim, de longo prazo, de pesquisa e padrões, bibliotecas e tecnologias fundamentais, que servem para muitas pesquisas, dependem de um fluxo constante de pessoas qualificadas atuando nisso, o que exige um componente educacional robusto, além de estabilidade e continuidade na infraestrutura de pesquisa (DONGARRA, 2022). É necessário investir na qualificação, nos

[4] Muito do software científico adota também licenças que permitem a integração do software em outras aplicações open source, ou mesmo comerciais, sem restrições significativas, promovendo assim um amplo uso na pesquisa e na indústria (DONGARRA, 2022).

[5] https://www.netlib.org/lapack

[6] https://www.netlib.org/lapack/contributor-list.html

estudos, na formação sólida de novos alunos e futuras equipes interdisciplinares, que trabalhem bem os fundamentos e construam novas soluções.

2.4.2 Tecnologias web para a aplicação

A aplicação deste trabalho utiliza tecnologias *web*, ou seja, a construção dele é realizada com as ferramentas fundamentais HTML, CSS e JavaScript do ecossistema *web*. A Linguagem de Marcação de Hipertexto (*Hypertext Markup Language* - HTML) provê a estrutura geral do conteúdo de uma página ou aplicação, incluindo textos, imagens, botões, campos de entrada, links e recursos multimídia. As Folhas de Estilo em Cascata (*Cascading Style Sheets* - CSS) detalham questões de como a estrutura HTML é apresentada, como cores, fontes de texto, etc. Por fim, JavaScript é a linguagem de programação responsável por gerar a interatividade na aplicação, incluindo as respostas e os algoritmos relacionados (MILETTO; BERTAGNOLLI, 2014).

Para trabalhar com grafos visuais usando a tecnologia *web*, existe uma popular biblioteca chamada vis.js[7]. A vis.js é utilizada para fornecer a visualização de dados em formatos como redes, gráficos de tempo, hierarquias e grafos de modo geral. Ela foi desenvolvida para lidar com grandes quantidades de dados dinâmicos e possibilitar a manipulação e interação com eles. Por oferecer uma interface intuitiva e flexível, a biblioteca se tornou uma opção acessível para desenvolvedores que desejam representar estruturas complexas, como matrizes de adjacência e conexões entre nós, sem a necessidade de lidar diretamente com algumas questões de baixo nível; ela abstrai detalhes técnicos e proporciona recursos como movimentação dinâmica dos nós, escalabilidade visual e personalização de eventos, facilitando a construção de representações interativas de dados em ambientes *web* (PERRONE; UNPINGCO; LU, 2020).

Outra tecnologia importante no presente trabalho é a biblioteca math.js[8], que provê diversas funções e implementações matemáticas para JavaScript. Ela também facilita o trabalho com diversos tipos de dados, como números complexos, matrizes, etc. Para o presente trabalho, utilizou-se a math.js para a realização de cálculos algébricos necessários, especialmente na resolução de sistemas lineares, por meio da função lusolve(), que resolve usando fatoração LU (usando permutações também, como mencionado anteriormente). A implementação dessa função, em math.js, é naturalmente feita puramente em JavaScript, para a execução no próprio navegador. Ademais, o tipo básico numérico em JavaScript segue o formato *double*, sendo este o formato padrão usado nos algoritmos numéricos implementados com a linguagem.

3 TRABALHOS RELACIONADOS

Uma ferramenta bastante empregada no ensino em muitos contextos matemáticos é o GeoGebra[9]. O software lida com questões geométricas e algébricas, podendo ser utilizado tanto

[7] O nome oficial é escrito em letras minúsculas. Endereço: https://visjs.org.
[8] https://mathjs.org/
[9] https://www.geogebra.org

no ensino médio quanto no ensino superior, tendo um enfoque visual (permitindo trabalhar com vetores, linhas, cônicas e elementos semelhantes). Por outro lado, o GeoGebra não possui suporte específico para a manipulação formal de grafos como estruturas de dados ou objetos computacionais, o que é apresentado nesse trabalho.

Outro software na área, permitindo também trabalhar com o viés educativo com vetores, cônicas, etc. é o TBC-GAAL, desenvolvido com Java (LIMA et al., 2008). Por essa razão, não é possível acessá-lo nativamente pelo navegador, o que seria diferente se a ferramenta fosse projetada com base nas tecnologias da *web*.

Já o AlgLin é uma opção que foi desenvolvida com as tecnologias *web*, sendo voltado ao ensino médio, focando em efetuar cálculos matriciais, mostrando as etapas passo a passo (MILAGRES, 2018). Contudo, o AlgLin está desativado.

No ensino superior, pode ser mais comum alunos empregarem ferramentas mais robustas ao trabalharem com a Álgebra Linear, principalmente em contextos computacionais ou da engenharia. Segundo Chonacky e Winch (2005), alguns dos softwares comerciais mais conhecidos na área são os "3 M's": MATLAB [10], da MathWorks; Mathematica [11], da Wolfram; e Maple [12], da Maplesoft. São softwares extensos, e bastante consolidados, que permitem lidar de forma mais geral com muitos elementos matemáticos no computador, por meio de comandos de uma linguagem própria.

No caso do MATLAB, uma alternativa gratuita open source é o Octave [13], da GNU, que se espelha na linguagem do MATLAB. Há, inclusive, o site "Octave Online"[14], um serviço que implementa uma interface *web* para o Octave, sendo usado por milhares de estudantes e educadores no mundo.

No mundo do *software* livre, o SageMath (STEIN; JOYNER, 2005), [15], é um sistema open source completo que usa várias tecnologias abertas como NumPy, SciPy, R, dentre outras (sobre esses, ver mais adiante). Segundo o próprio site, a missão dele é "criar uma alternativa open source viável para o Magma, Maple, Mathematica e Matlab". Os autores comentam a importância de ter um sistema de código aberto: o código-fonte deve estar livremente disponível e legível, para que os usuários possam entender o que o sistema realmente está fazendo e até mesmo estendê-lo". Pois assim como os matemáticos adquirem uma compreensão mais profunda de um teorema ao acompanhar cuidadosamente a demonstração dele, desenvolvedores também podem ver melhor como um sistema funciona ao acompanhar cuidadosamente o código-fonte, devidamente documentado (STEIN; JOYNER, 2005).

As tecnologias mencionadas anteriormente, usadas pelo SageMath, referem-se sobretudo às populares bibliotecas NumPy [16] e SciPy [17], para a linguagem Python, além da menção a R

[10] https://www.mathworks.com/products/matlab.html
[11] https://www.wolfram.com/mathematica/
[12] https://www.maplesoft.com/products/Maple/
[13] https://octave.org/
[14] https://octave-online.net/
[15] https://www.sagemath.org
[16] https://numpy.org/
[17] https://scipy.org/

[18], uma linguagem de programação bastante aplicada em estatística, ciência de dados, dentre outras áreas, sendo implementada como *software* livre, estando disponível para Linux, Windows, macOS.

Já a linguagem Python, que apareceu na década de 1990, ganhou maior impulso na década de 2000, popularizando-se nessa década na comunidade científica com o NumPy e SciPy. Na década de 2010, houve um grande crescimento do uso de Python com ciência de dados e aprendizado de máquina, e hoje, é uma das linguagens mais populares do mundo. Com essas duas bibliotecas citadas, pode-se trabalhar em Python com matrizes e sistemas lineares. Vale frisar que Python é *software* livre, sendo o projeto desenvolvido com o comprometimento com o código aberto, de acesso livre e gratuito, e com desenvolvimento comunitário, o que permitiu contribuições de milhares de desenvolvedores e gerou um ecossistema robusto em torno da linguagem, inclusive de tecnologias como NumPy e SciPy.

Na década em que o NumPy foi lançado, surgiu também o IPython, um terminal interativo para Python com muitas conveniências, de onde sairia o IPython Notebook posteriormente, que por sua vez deu origem ao Jupyter Notebook. O Jupyter Notebook ia além de Python, sendo o nome referente às linguagens iniciais suportadas: Julia, Python e R. Julia [19] é uma linguagem mais recente, sendo também, assim como Python e R, um projeto open source, e tendo maior foco em desempenho, com uso em cálculo numérico, ciência de dados, inteligência artificial.

Na comunidade acadêmica, o Jupyter tem sido bastante utilizado. Inspirado no Mathematica, o Jupyter é uma ferramenta *web* que funciona como um ambiente para trabalhar com documentos dinâmicos que misturam texto, códigos e visualizações gráficas de dados em uma interface gráfica de *notebook*, um paradigma que também havia no Mathematica e MATLAB. Os assim chamados "notebooks" podem ajudar na exploração de dados e visualizações gráficas, podendo ser compartilhados pela internet. De fato, com muito do trabalho científico atual sendo feito de forma computacional e com o uso intensivo de dados, ter acesso a esse tipo de mecanismo para lidar com os dados e compartilhar com os outros tem sido algo muito valioso para uma prática científica mais aberta, como comentam os autores (RANDLES et al., 2017).

Há diversos repositórios de notebooks para a Álgebra Linear[20]. Vários desses repositórios online contêm arquivos .ipynb (Jupyter Notebook), tecnicamente arquivos JSON com algumas informações de controle (por exemplo, a linguagem empregada nos códigos, a versão utilizada,

[18] https://www.r-project.org

[19] https://julialang.org

[20] Ver alguns como:

- <https://github.com/weijie-chen/Linear-Algebra-With-Python/tree/master/notebooks>;
- <https://github.com/fastai/numerical-linear-algebra,https://github.com/vbartle/VMLS-Companions>;
- <https://github.com/weijie-chen/Linear-Algebra-With-Python>;
- <https://github.com/juanklopper/MIT_OCW_Linear_Algebra_18_06> (baseado em um curso de Álgebra Linear do conhecido professor Gilbert Strang);
- <https://github.com/bvanderlei/jupyter-guide-to-linearalgebra> (Introdução a Álgebra Linear com notebooks Jupyter).

dentre outras informações) e uma lista de células do tipo 'código' (contendo um código em Python, por exemplo) ou 'Markdown' (uma linguagem de marcação simples, que permite fazer textos formatados), tendo ainda a possibilidade de cada célula ter uma saída gerada (que pode ser a imagem de um gráfico salvo como .png, um texto puro simples, uma fórmula em LaTex, etc).

Esses arquivos são gerados e, então, apresentados de forma estática (como na plataforma Github.com), ou dinâmica (permitindo execução interativa), em ferramentas offline (como Jupyter Notebook ou VS Code) e plataformas online (como, por exemplo, o Google Colab [21], que permite o uso de recursos de computação como GPU's para ciência de dados, aprendizado de máquina).

Certamente, isso traz benefícios para quem quer lidar com Álgebra Linear no computador, podendo digitar códigos e executá-los em um ambiente *web*, fazer visualizações dos resultados obtidos, além de poder compartilhá-los. Contudo, isso ainda se dá no paradigma de comandos, sendo necessário conhecer os nomes das rotinas, a sintaxe, a linguagem em si para usar esse tipo de tecnologia. Além disso, conforme o artigo (PIMENTEL et al., 2019), muitos repositórios de notebooks têm problemas, como a falta de especificação completa das dependências, versões de bibliotecas usadas ou necessárias, entre outros.

Ademais, como Python não é uma linguagem nativa do navegador, a execução de código Python é tipicamente encaminhada para o lado do servidor. No Jupyter Notebook instalado localmente, é executado o *backend* na própria máquina do desenvolvedor. Em serviços online, pode ser executado por meio de alguma máquina virtual remota, executando em uma infraestrutura de nuvens.

Existem implementações de bibliotecas relacionadas para outras linguagens também. Para C++, uma conhecida biblioteca é a Eigen [22], que, segundo o site, funciona com vários compiladores, embora tenha o melhor desempenho com os compiladores GCC e Clang. Outra biblioteca, Armadillo [23], busca ter uma sintaxe de alto nível, inspirada no Matlab, e a implementação dela conta com o uso do OpenMP, que facilita um processamento paralelo otimizado com multi-thread. Desenvolvedores de C++ que usam Boost [24], uma coleção robusta de bibliotecas abertas, têm também a opção do uBLAS [25]. Por fim, podem ser citadas as bibliotecas do Commons Mathematics Library do Apache [26], para a linguagem Java, e a `ndarray_linalg` [27] ou a `nalgebra` [28], para a linguagem Rust.

Ter instrumentos disponíveis na *web* para lidar com sistemas lineares ou outros tópicos da Álgebra Linear é muito valioso, proporcionando acesso imediato à ferramenta praticamente de qualquer lugar, a qualquer hora e em múltiplos meios de acesso. Uma forma de aliviar a

[21] https://colab.google/
[22] https://eigen.tuxfamily.org
[23] https://arma.sourceforge.net
[24] https://www.boost.org
[25] https://www.boost.org/doc/libs/latest/libs/numeric/ublas/doc/index.html, ver também o repositório em https://github.com/boostorg/ublas.
[26] https://commons.apache.org/proper/commons-math
[27] https://docs.rs/ndarray-linalg/latest/ndarray$_{l}$*inalg*/*index.html*
[28] https://github.com/dimforge/nalgebra

pressão nos servidores e permitir a execução dos algoritmos na própria máquina do usuário, utilizando-se do próprio navegador dele, é usando a linguagem nativa do navegador, isto é, o JavaScript, ou transpilando o código para JavaScript (como TypeScript ou CoffeeScript).

Nesse contexto de desenvolvimento com JavaScript, duas bibliotecas bastante empregadas na área são Math.js [29] e ML-Matrix [30]. Segundo o site da plataforma npmjs.com, Math.js tem 1,3 milhões de downloads ao mês, enquanto ML-Matrix 680 mil [31]. O presente trabalho, optando pelo tratamento de sistemas lineares sem precisar de um servidor, faz uso de Math.js, como mencionado anteriormente.

Na *web*, foram feitos inúmeros esforços ao longo dos anos para a otimização de JavaScript e para a implantação de diversos recursos a fim de enriquecer a experiência e prover novas possibilidades às aplicações executadas no navegador. O código JavaScript pode ser convertido em código nativo durante a execução, e mesmo iniciativas como WebAssembly e WebGL fornecem opções extras ao desenvolvedor.

Um padrão ainda mais novo, WebGPU [32], traz a possibilidade de empregar a GPU de forma mais robusta e otimizada do que WebGL, inclusive para processamentos computacionais gerais à parte de gráficos através do chamado *Compute Shader*, atendendo a demandas crescentes por processamento intensivo em inúmeras aplicações, inclusive de jogos, simulações científicas e inteligência artificial, podendo executar nativamente no navegador do usuário (seja em computadores ou celulares).

Ao permitir o processamento no próprio navegador, sem uso de um servidor remoto, por exemplo, isso significa, em muitos contextos, que os dados não precisam sair do dispositivo do usuário para um serviço na nuvem, fator relevante para a privacidade (por exemplo em análise médica ou financeira). Também implica em uma latência muito melhor, pois não há a necessidade de comunicação com servidores, além de aliviar a alta sobrecarga sobre a infraestrutura de nuvens.

A busca por eficiência e otimização ocorre em função da alta demanda de processamento. A quantidade de operações de Álgebra Linear sendo processada por segundo em grandes sistemas de inteligiência artificial é imensa. Tecnologias famosas como PyTorch, TensorFlow, MATLAB, R, NumPy usam bastante rotinas fundamentais de Álgebra Linear. *Hardwares* especializados, como as TPUs da Google ou GPUs da NVidia, otimizam muitos cálculos da disciplina que são relevantes para a inteligência artificial e computação gráfica.

A Intel, por exemplo, produziu a biblioteca Intel MKL (Math Kernel Library), posteriormente renomeada para Intel oneMKL [33], que otimiza diversas rotinas fundamentais (incluindo de Álgebra Linear) para a arquitetura dos processadores, usando bem os recursos disponíveis de *hardware*, como as instruções SIMD AVX-512. Assim, quando um indivíduo usa um *software*, ou biblioteca que adota essa implementação da Intel, o *software* pode executar significativamente mais rápido.

[29] https://mathjs.org/
[30] https://github.com/mljs/matrix
[31] Ver as estatísticas em https://www.npmjs.com/package/mathjs e https://www.npmjs.com/package/ml-matrix.
[32] https://www.w3.org/TR/webgpu
[33] https://www.intel.com/content/www/us/en/developer/tools/oneapi/onemkl.html

Também foram feitas implementações de várias rotinas fundamentais para executarem com maior velocidade nas GPUs da NVidia, por meio da tecnologia dela conhecida como CUDA (Compute Unified Device Architecture). Assim, ao usar uma biblioteca ou *software* de Álgebra Linear com esse suporte, a execução poderá ser muito mais rápida. As otimizações das rotinas fundamentais permitem treinamentos mais rápidos em Inteligência Artificial, a otimização do trabalho em *datacenters* e a viabilização de novos recursos.

O enriquecimento das tecnologias *web*, no navegador, pode fornecer muitas possibilidades relevantes para a educação. Por exemplo, os autores Dan Margalit e Joseph Rabinoff produziram um livro de Álgebra Linear que funciona também como um site interativo, onde gráficos e outros recursos interativos são inseridos em meio ao texto e o leitor pode interagir com eles [34]; o código do projeto deles está disponível [35]. O site tem um breve índice de capítulos, e em cada capítulo há figuras que são interativas, isto é, o usuário pode clicar, arrastar, interagir.

Segundo o próprio site, o livro é escrito com XML, usando uma variante de MathBook XML; a renderização das fórmulas é feita com LaTeX e convertida para o formato gráfico vetorial SVG para a exibição HTML, usando o software livre Inkscape no processo, além da produção das figuras com PGF/TikZ. A interatividade do projeto é feita com JavaScript e WebGL, com o framework MathBox. Esse tipo de projeto ilustra algumas das possibilidades valiosas para a educação, com os novos avanços tecnológicos na *web*.

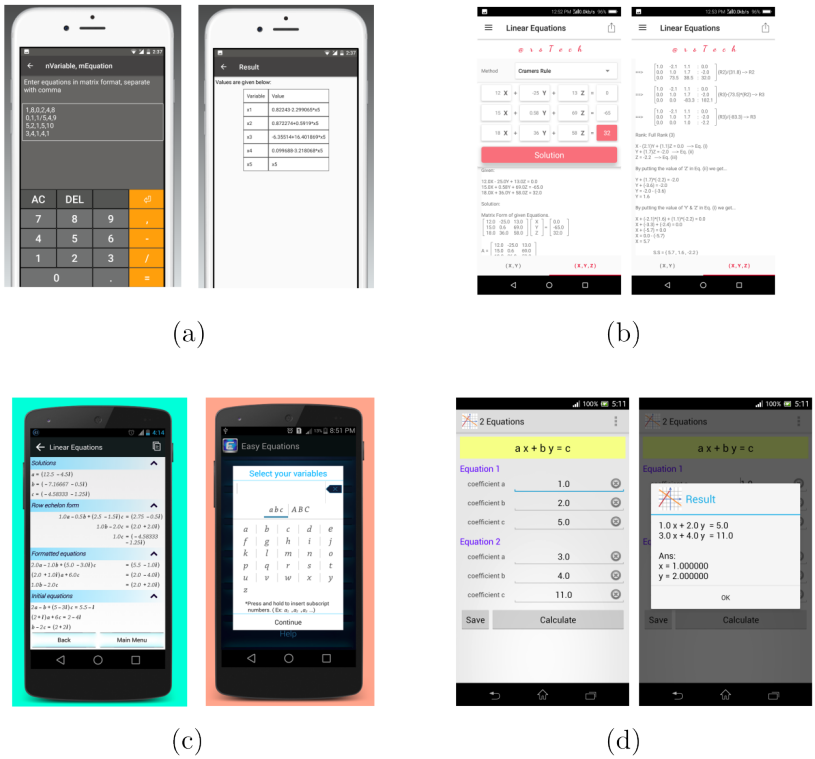
No desenvolvimento de implementações *web*, cabe não apenas a preocupação com a apresentação da interface gráfica na tela de um computador de mesa (*desktop*), como também na tela de um celular. De fato, várias implementações frequentemente buscam otimizar isso, melhorando a chamada 'responsividade' do projeto.

Existem aplicações para Android, por exemplo, lidando com a Álgebra Linear e sistemas de equações, mesmo que em uma tela pequena. Alguns aplicativos relacionados são apresentados na Figura 3, que incluem: (a) Linear Equation System Solver [36], (b) Linear Equations Solver [37], (c) Easy Equations: Linear Equation Solver [38] e (d) Linear Equations [39]. São aplicativos que apresentam os conceitos de forma geral, com a maioria contendo também anúncios.

Com essa análise, pode-se perceber que existem ferramentas bem robustas para lidar com a Álgebra Linear no ensino superior usando comandos e programação. Contudo, para quem está iniciando ou deseja focar no aprendizado dos conceitos em si, à parte de comandos e códigos, pode sentir falta de algo mais pontual, voltado a cenários mais simples e de teor ilustrativo, didático. A ferramenta proposta nesse trabalho se encaixa nessa linha, abordando um problema específico de sistema linear, podendo assim apresentá-lo de forma mais gráfica e interativa. O conteúdo gráfico dela é passível de ser apresentado tanto na tela de um computador de mesa, quanto na tela de um celular, buscando assim ter a devida responsividade. O presente trabalho

[34] disponibilizado em https://textbooks.math.gatech.edu/ila
[35] https://github.com/QBobWatson/ila
[36] https://play.google.com/store/apps/details?id=com.mathapps.linearequations
[37] https://play.google.com/store/apps/details?id=com.arslan.linearequationscs
[38] https://play.google.com/store/apps/details?id=com.kasun.easyequations
[39] https://play.google.com/store/apps/details?id=com.peterhohsy.linearequation

Figura 3 – Telas de Aplicativos Relacionados



(a) (b)

(c) (d)

Fonte: Elaborada pelo autor.

busca, assim, apresentar uma aplicação mais específica de sistemas lineares, de modo a trabalhar melhor o aspecto visual e o apelo da interatividade para o aprendizado, sem nenhuma necessidade de comandos, mas apenas focando em um problema concreto e convidativo à exploração, sem deixar, contudo, de apresentar o sistema linear e as equações dele na relação com o problema.

4 METODOLOGIA

A fim de se alcançar o objetivo proposto, adotou-se a seguinte metodologia: pesquisa aplicada em busca de um problema potencial; exploração e escolha de tecnologias, bibliotecas e ferramentas a serem empregadas; implementação; e, por fim, a aplicação de um questionário para validar a proposta apresentada.

Com base na pesquisa realizada em livros de Álgebra Linear, como Álgebra Linear e Suas Aplicações de (LAY; LAY; MCDONALD, 2018) e *Linear Algebra and Its Applications* de (STRANG, 2016), foi escolhido o problema a ser abordado e apresentada a respectiva solução. Esta seleção levou em consideração um problema relacionado a conteúdos comumente apresentados em livros da área, mas frequentemente vistos como desafiantes ou distantes de aplicações práticas por muitos alunos.

Em um segundo momento, optou-se pelo uso de tecnologias web, e, para o desenvolvimento, foi escolhida a ferramenta do *Visual Studio Code*, gratuita, relativamente leve, bastante utilizada na área e que oferece suporte para várias linguagens. O código para a solução foi

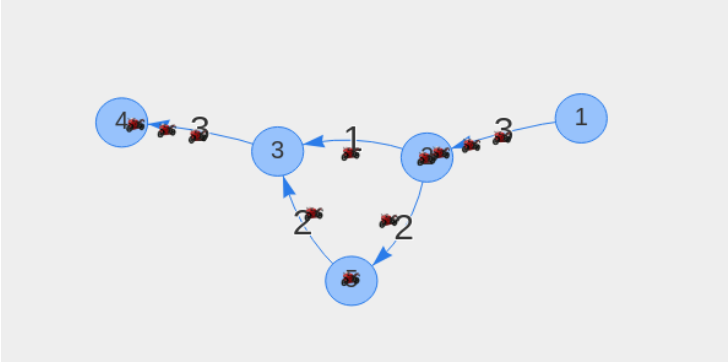
armazenado no Github.com, sendo ela disponibilizada *on-line* por meio do serviço de nuvem da *Vercel*.

4.1 Problema e Implementação

No que diz respeito à representação do Fluxo de Tráfego Urbano, propõe-se a modelagem e a exibição de um grafo ponderado e direcionado, o qual representa o fluxo de tráfego urbano. Como visto na seção da fundamentação teórica, as maneiras mais comuns de se representar um grafo são por meio de matriz de adjacência ou lista de adjacência. Para este trabalho, utilizou-se, de modo direto, a matriz de adjacência pois ela permite, matematicamente, uma visualização mais intuitiva da composição do grafo. Além disso, ela lida melhor com grafos mais densos, sendo mais adequada para analisar rapidamente a conectividade entre vértices (CATARINO, 2025), propriedade relevante para esta solução.

Para a presente implementação, os vértices indicam os cruzamentos, as arestas simbolizam as vias e os pesos das arestas correspondem à quantidade de veículos que entram ou saem de um determinado cruzamento (Figura 4).

Figura 4 – Modelagem do Grafo



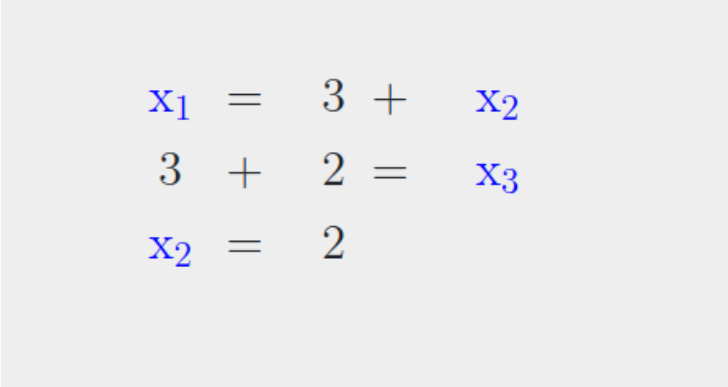
Fonte: Elaborada pelo autor.

É possível que a quantidade de automóveis próxima a um cruzamento específico pode ser desconhecida. Por isso, as arestas do grafo podem ter pesos representados por incógnitas. Para determinar esses valores desconhecidos é que o sistema de equações é modelado e então resolvido. Subsequentemente, os valores obtidos são substituídos nas respectivas incógnitas do grafo. Para a resolução do sistema, adotou-se a função lusolve(), da biblioteca math.js, que emprega a fatoração LU.

Para a modelagem das equações, percorre-se o grafo e identificam-se as incógnitas, que são então coletadas. Além disso, constroem-se, em cada vértice, expressões que representam o somatório dos pesos das arestas que chegam a ele, bem como expressões da soma dos valores que saem do vértice. Assim, para garantir a conservação do fluxo de tráfego, igualam-se as expressões de entrada e de saída, que são utilizadas na resolução do sistema linear. Por fim, para a aplicação da função lusolve(), foram coletados os coeficientes das equações e os resultados delas, possibilitando a obtenção dos valores desconhecidos.

Além da visualização do grafo que representa o fluxo de tráfico urbano, exibe-se, quando há incógnitas inseridas no grafo, o sistema de equações relacionado à matriz geradora do grafo. Nele há integrações de eventos de escuta, que garantem que as alterações no grafo sejam refletidas de forma simultânea. Logo, ao se adicionar um vértice, editar o peso de uma aresta ou resolver o sistema linear, tal modificação será refletida na visualização do sistema de equações. Quando se utiliza a funcionalidade de resolução de sistemas, retorna-se para a mensagem de quando não existe incógnitas, avisando: "Insira pelo menos uma incógnita para gerar as equações". Além da visualização do grafo, também é fornecida a exibição do sistema linear correspondente (Figura 5).

Figura 5 – Sistema linear correspondente



Fonte: Elaborada pelo autor.

Como recurso adicional, implementou-se uma animação para simular o tráfego de veículos. Na animação, os veículos se movem ao longo das arestas do grafo, representando o fluxo de entrada e saída em cada cruzamento. A quantidade do fluxo respeita o peso da aresta de cada vértice, produzindo um efeito mais realístico. Por exemplo, se uma aresta tem peso 3, então 3 veículos estarão renderizados nela; se tem peso 5, a animação será feita com 5 veículos.

A lógica empregada para a animação segue os seguintes princípios: usa-se a função requestAnimationFrame(), que solicita ao navegador a intenção de realizar uma animação por meio de uma função específica, que atualiza a posição dos veículos em cada quadro renderizado pelo navegador, continuamente. Essa repetição permite a ideia de movimento suave ao longo do tempo. Para cada aresta, associa-se ainda um valor de progresso, que representa a fração da distância percorrida entre o cruzamento de origem e o de destino, variando de 0.0 a 1.0; assim, a posição do veículo é recalculada a cada quadro com base no progresso. Quando o progresso ultrapassa o valor um, o veículo é reiniciado para o início da aresta, onde o progresso será zero, criando um movimento contínuo.

Para garantir que cada veículo saia do vértice de origem e chegue ao de destino, a posição em tela referente a ele é calculada por uma interpolação linear, segundo a expressão a seguir, responsável por calcular as coordenadas de x e y para cada vértice:

$$x = \text{fromNode}.x + (\text{toNode}.x - \text{fromNode}.x) \cdot \text{progress} \quad (1)$$

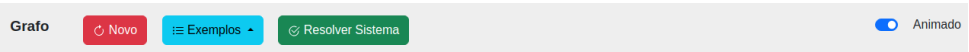
$$y = \text{fromNode}.y + (\text{toNode}.y - \text{fromNode}.y) \cdot \text{progress} \quad (2)$$

No que diz respeito ao desenho do respectivo veículo, que pode ser um dentre os três disponíveis, escolhido aleatoriamente, utiliza-se a função ctx.drawImage(), da API do Canvas 2D para o desenho da imagem na posição correta. Por fim, chama-se a função afterDrawing(), da biblioteca vis.js, para sobrepor os veículos sobre o grafo previamente desenhado.

Um ponto a se destacar é que a biblioteca vis.js, nativamente, utiliza-se de uma estrutura de listas (arrays) para a geração de grafos, esperando receber, portanto, em forma de array, os dois componentes principais, que são os *nodes* e as *edges*, isto é, os nós e as arestas. Por essa razão, para se tornar possível a renderização do grafo por meio dela, é efetuada a conversão da matriz de adjacência em arrays.

Além da visualização do grafo e do sistema de equações referente à matriz de adjacência geradora do grafo, foram criados dois conjuntos de funcionalidades para interagirem com toda a aplicação ou para serem aplicadas especificamente na edição do grafo e dos sistemas lineares. O primeiro conjunto apresenta as funcionalidades para reiniciar a página, para visualizar exemplos de grafos pré-definidos, resolver sistemas e ativar ou desativar a animação dos veículos, respectivamente (Figura 6). Assim, permite-se que o usuário retorne ao estado inicial da aplicação por meio do botão "Novo", aplique os modelos de grafos contidos no botão "Exemplos", encontre os valores das incógnitas por meio do botão "Resolver Sistema"e ative, ou desative, a animação com o controle correspondente.

Figura 6 – Botões de Manipulação do Site



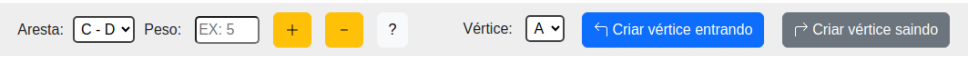
Fonte: Elaborada pelo autor.

O segundo conjunto, por sua vez, contém recursos que permitem selecionar a aresta que se deseja modificar, inserir o peso referente a ela, ou ao vértice que se deseja adicionar ao grafo, adicionar ou reduzir em uma unidade o valor do peso das arestas, ou do vértice que se deseja acrescentar e inserir incógnitas às arestas ou aos vértices. Além disso, permite-se selecionar o vértice com o qual se deseja interagir, adicionando um outro vértice a ele, seja um vértice de entrada ou de saída.

Para a edição dos pesos da aresta, é necessário selecionar a aresta desejada. Ao fazer isso, será atualizado no campo o valor do peso correspondente a ela, podendo o usuário modificá-lo com os botões representados por “+” ou “-”, ou inserir o valor do peso que se deseja no campo de entrada e pressionar Enter. Caso se queira adicionar uma incógnita ao peso da aresta, basta que o usuário apague o valor do campo peso e pressione Enter ou clique no botão representado por uma interrogação.

De forma semelhante, para se adicionar um vértice ao grafo, é necessário selecionar o vértice no qual ele será adicionado. Supondo que esteja selecionado o vértice "A", se o usuário deseja que o novo vértice esteja entrando em "A", ele deve clicar no botão "Criar vértice entrando"; por outro lado, se ele deseja que o novo vértice esteja saindo de "A", basta que ele clique no botão "Criar vértice saindo". Se for inserido um peso no campo de entrada, o novo vértice será vinculado ao vértice selecionado com tal peso. Caso contrário, é inserida uma incógnita a ele (Figura 7).

Figura 7 – Botões de Manipulação



Fonte: Elaborada pelo autor.

Para a estilização dos conjuntos de funcionalidades citados, foram utilizados recursos do *Bootstrap*[40], como classes e ícones, que auxiliam e enriquecem a interface. O *Bootstrap* é um *toolkit front-end open source* bastante utilizado e difundido em diversos sites e implementações *web*, tendo propriedades como responsividade e variedade de estilos e ícones.

Para facilitar a observação, a compreensão e a rastreabilidade das ações realizadas na ferramenta, foi criada uma seção de registros (Log). Esta, por sua vez, registra a ação e o momento em que ela foi realizada. Quando o sistema de equações é resolvido, por exemplo, o valor da incógnita é substituído no grafo, e nele ela já não existe mais. No entanto, na seção de registros aparecerá uma explicação, como "Sistema resolvido: X1 = 3 + 2. Logo, X1 = 5.", o que facilita a compreensão do usuário que desejar acompanhar o processo mais pausadamente, com atenção aos detalhes. Também é possível excluir os registros ao clicar em um botão representado por uma lixeira (Figura 8).

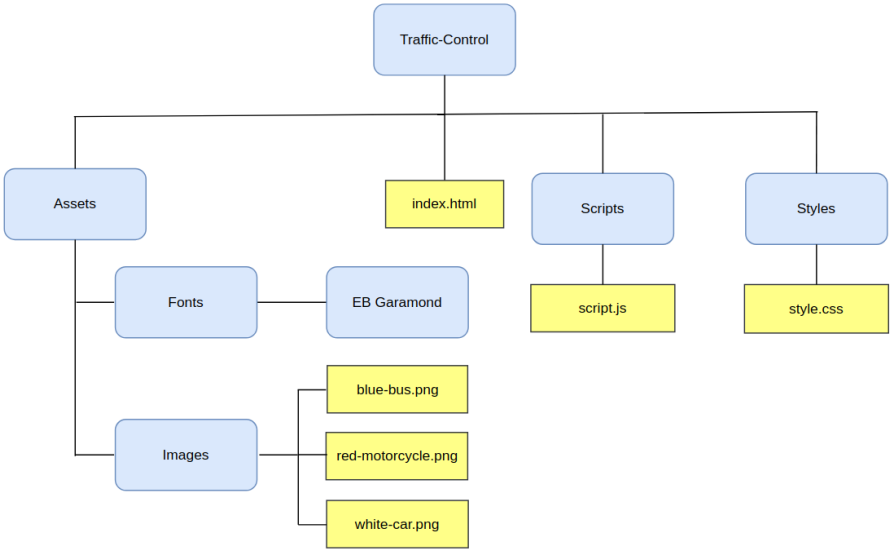
Figura 8 – Seção de Registros

Fonte: Elaborada pelo autor.

Para organizar a solução de forma eficiente, foram criadas pastas para segmentar os arquivos de acordo com as funções. Dessa forma, há a pasta ***assets***, responsável por armazenar os arquivos estáticos; a pasta de ***scripts***, destinada aos arquivos que contém a lógica da aplicação; e a pasta ***styles***, que armazena os arquivos de estilização. Na raiz do projeto, encontra-se o código principal (***index***), conforme mostra a Figura 9).

[40] https://getbootstrap.com/

Figura 9 – Organização do Projeto



Fonte: Elaborada pelo autor.

4.2 Aplicação de um Questionário

Para validar a adequação da ferramenta como instrumento de apoio ao aprendizado para Álgebra Linear, foi aplicado o seguinte questionário:

1. O sistema de equações e o grafo estão claros e bem legíveis?
2. Os botões da interface estão claros em relação ao que fazem?
3. Você conseguiu entender como o sistema de equações reflete o que se faz no grafo?
4. Você acredita que a ferramenta pode ajudar no engajamento de alunos aprendendo problemas de sistemas lineares?
5. Você teria alguma sugestão de melhoria?

As quatro primeiras questões do questionário eram de caráter objetivo, de resposta obrigatória e avaliadas em uma escala de 1 a 5, sendo 1 a menor pontuação possível e 5 a pontuação máxima. A quinta questão, por sua vez, era subjetiva e de resposta opcional, permitindo que os participantes expressassem comentários ou sugestões de forma livre.

Os dados coletados são apresentados na seção seguinte.

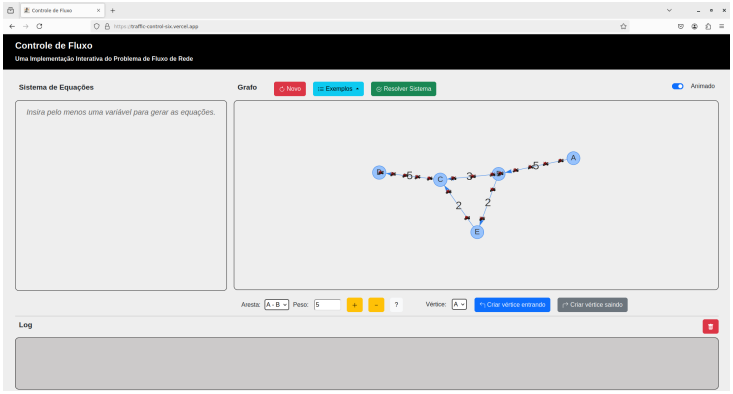
5 RESULTADOS

A Figura 10 aborda o resultado da implementação da proposta. Nessa tela, o usuário tem as opções de visualizar o sistema de equações referente à matriz geradora (à esquerda), do grafo (à direita), assim como interagir com o grafo e perceber que as modificações refletem no sistema

de equações. É possível adicionar vértices (nós) ao grafo, editar os pesos das arestas, adicionar incógnitas e também resolvê-las, observando em cada momento as alterações geradas no grafo.

As incógnitas aparecem automaticamente caso um vértice seja criado sem a informação do peso, ou se este não for informado durante a edição da aresta; mas também aparecem se o usuário selecionar a aresta e acionar o botão simbolizado por um ponto de interrogação. Além disso, a ferramenta possui uma seção que registra o histórico das operações realizadas.

Figura 10 – Visão geral da solução.



Fonte: Elaborada pelo autor.

A interface gráfica da ferramenta foi projetada de forma responsiva, garantindo o acesso por meio de dispositivos móveis. A Figura 11 apresenta a interface para um dispositivo com tela de dimensões menores, como de 320x568 pixels.

Depois da implementação ter sido realizada, um questionário foi aplicado com 28 alunos, sendo eles do curso de Ciência da Computação, que acessaram e interagiram com a ferramenta brevemente para coletar informações sobre a experiência deles. Esse questionário teve como objetivo principal entender a usabilidade da plataforma, a eficácia da temática apresentada e a satisfação geral dos usuários.

A pesquisa foi conduzida de forma anônima, sem a coleta de dados pessoais identificáveis, como nome ou e-mail, respeitando os princípios da Lei Geral de Proteção de Dados (LGPD), garantindo a privacidade e a segurança das informações dos participantes. Com base nas respostas obtidas, foi possível realizar melhorias e ajustes na interface gráfica da ferramenta, a fim de atender às demandas. Além disso, o questionário também permitiu avaliar a clareza das instruções e a facilidade de navegação. A Figura 12 apresenta os resultados do questionário.

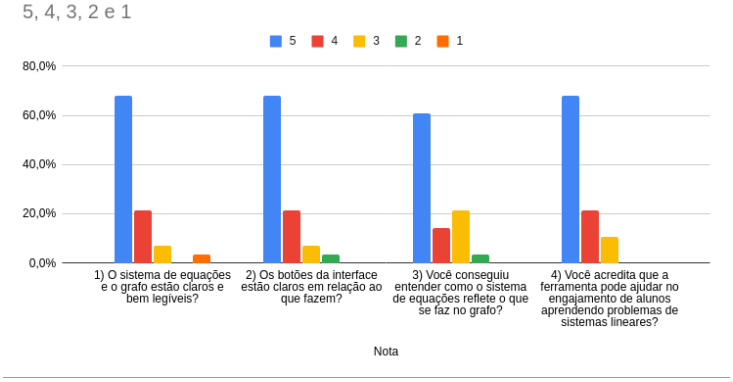
As respostas obtidas por meio da aplicação do formulário evidenciam, por exemplo, que a exibição dos principais recursos da ferramenta, como o grafo, o sistema de equações e os botões de modificação, foi clara e legível para os participantes. Isso se reflete nas altas taxas de satisfação dos usuários, com 67,9% atribuindo a nota máxima (5) e 21,4% atribuindo a nota 4, totalizando 89,3% de aprovação.

Por outro lado, observou-se uma percepção ligeiramente inferior sobre a forma como o sistema de equações se reflete no grafo. Nesse caso, 60,7% dos usuários forneceram a nota 5, enquanto 14,3% a nota 4, resultando em uma satisfação de 75%. Por fim, a pesquisa indicou que

Figura 11 – Visão geral da solução em um Smartphone.

Fonte: Elaborada pelo autor.

Figura 12 – Resultado do Questionário de Validação.



Fonte: Elaborada pelo autor.

a ferramenta tem o potencial de aumentar o engajamento dos alunos no aprendizado de sistemas lineares, com 67,9% dos usuários atribuindo a nota máxima (5) e 21,4% a nota 4, totalizando 89,3% de satisfação.

6 CONCLUSÃO

Este trabalho apresentou uma ferramenta *web* visual e interativa voltada para ajudar no aprendizado de sistemas lineares por meio de um exemplo prático, o problema de Fluxo de Tráfego (LAY; LAY; MCDONALD, 2018). O problema envolve um grafo que pode ser editado pelo aluno, com as modificações dele afetando diretamente o sistema linear que é exibido pela ferramenta.

Para tornar a ferramenta mais atrativa, foram criadas animações de veículos fluindo no grafo, simulando a natureza do problema; e, para ajudar no entendimento do aluno, ao clicar nos vértices do grafo a equação correspondente no sistema linear é destacada. Além disso, cada número editado no grafo reflete diretamente no sistema exibido, ajudando no aprendizado.

Durante a implementação, foram tratados elementos como matriz de adjacências, sistema linear, manipulações de grafo e animação de imagens. O uso de bibliotecas da linguagem JavaScript como math.js e vis.js foi fundamental, permitindo o desenvolvimento focar na lógica de nível mais alto da aplicação em si e suas abstrações e menos em detalhes de nível mais baixo. Além disso, a ferramenta também optou por um projeto com responsividade para permitir um uso apropriado tanto em computadores de mesa quanto em celulares. O uso da ferramenta por alunos, assim como o resultado da pesquisa realizada com eles, demonstrou que a prática com sistemas lineares pode, de fato, ser enriquecida por uma atividade dinâmica no computador, envolvendo um problema de natureza mais prática e visual.

Para trabalhos futuros, pode-se examinar o uso de WebGPU para a implementação do algoritmo relacionado de resolução de sistemas lineares, explorando como o uso de um *hardware* moderno, por meio do navegador, poderia ser feito. Uma versão gráfica mais intensa com *ThreeJS* também poderia ser considerada. Ademais, a adição de sons poderia ser feita para enriquecer a experiência do aluno. Por fim, opções como salvar ou abrir grafos editados também podem ser buscadas.

REFERÊNCIAS

AGUIAR, I. A. Maslab: um software interativo de simulação para apoio no ensino de modelagem e análise de sistemas lineares. 2022.

ANDRADE, C. C. d. O ensino da matemática para o cotidiano. Universidade Tecnológica Federal do Paraná, 2013.

CAMPBELL-KELLY, M. Origin of computing. **Scientific American**, JSTOR, v. 301, n. 3, p. 62–69, 2009.

CATARINO, M. H. **Teoria dos Grafos**. 1. ed. Rio de Janeiro: Freitas Bastos, 2025. E-book. Disponível em: <https://plataforma.bvirtual.com.br>. Acesso em: 25 abr. 2025.

CHONACKY, N.; WINCH, D. 3ms for instruction: Reviews of maple, mathematica, and matlab. **Computing in Science & Engineering**, IEEE, v. 7, n. 3, p. 7–13, 2005.

COIMBRA, J. L. et al. Alguns aspectos problemáticos relacionados ao ensino-aprendizagem da álgebra linear. Universidade Federal do Pará, 2008.

DONGARRA, J. J. The evolution of mathematical software. **Commun. ACM**, Association for Computing Machinery, New York, NY, USA, v. 65, n. 12, p. 66–72, nov. 2022. ISSN 0001-0782. Disponível em: <https://doi.org/10.1145/3554977>.

GOLUB, G. H.; LOAN, C. F. V. **Matrix computations (3rd ed.)**. USA: Johns Hopkins University Press, 1996. ISBN 0801854148.

HRYNIEWICZ, B. et al. Biblioteca para exemplificação no ensino de álgebra linear. In: SBC. **Simpósio Brasileiro de Educação em Computação (EDUCOMP)**. [S.l.], 2024. p. 336–345.

LAY, D. C.; LAY, S. R.; MCDONALD, J. J. **Álgebra Linear e Suas Aplicações**. 5. ed. São Paulo: LTC, 2018.

LIMA, I. R. et al. Aprendizado de geometria analítica e álgebra linear utilizando um software gráfico via internet. In: SBC. **Simpósio Brasileiro de Sistemas de Informação (SBSI)**. [S.l.], 2008. p. 94–105.

LIPSCHUTZ, S.; LIPSON, M. **Matemática discreta-: Coleção schaum**. [S.l.]: Bookman Editora, 2013.

MILAGRES, D. C. Alglin: uma ferramenta para mediar o processo de ensino-aprendizagem em álgebra linear. **Anais do Seminário Sul-Mato-Grossense de Pesquisa em Educação Matemática**, v. 12, n. 1, 2018.

MILETTO, E. M.; BERTAGNOLLI, S. de C. **Desenvolvimento de Software II: Introdução ao Desenvolvimento Web com HTML, CSS, JavaScript e PHP-Eixo: Informação e Comunicação-Série Tekne**. [S.l.]: Bookman Editora, 2014.

MORAES, G. R. C. PrÁtica pedagÓgica direcionada a possibilidatar a aprendizagem significativa dos alunos nas aulas de matemÁtica. **Revista Tópicos**, v. 1, n. 3, 2023. Disponível em: <https://doi.org/10.5281/zenodo.10348568>.

MORÁN, J. Mudando a educação com metodologias ativas. **Coleção mídias contemporâneas. Convergências midiáticas, educação e cidadania: aproximações jovens**, v. 2, n. 1, p. 15–33, 2015.

PERRONE, G.; UNPINGCO, J.; LU, H.-m. Network visualizations with pyvis and visjs. **arXiv preprint arXiv:2006.04951**, 2020. Disponível em: <https://arxiv.org/abs/2006.04951>.

PIMENTEL, J. F. et al. A large-scale study about quality and reproducibility of jupyter notebooks. In: IEEE. **2019 IEEE/ACM 16th international conference on mining software repositories (MSR)**. [S.l.], 2019. p. 507–517.

RANDLES, B. M. et al. Using the jupyter notebook as a tool for open science: An empirical study. In: IEEE. **2017 ACM/IEEE Joint Conference on Digital Libraries (JCDL)**. [S.l.], 2017. p. 1–2.

ROMEIRO, R. A. G.; GARCIA, R. V.; ROMÃO, E. C. O ensino de funções e a educação tecnológica: o simulador phet e o software winplot como facilitadores da aprendizagem. **Caminhos da Educação Matemática em Revista (Online)**, v. 11, n. 2, p. 111–131, 2021.

STEIN, W.; JOYNER, D. Sage: System for algebra and geometry experimentation. **Acm Sigsam Bulletin**, ACM New York, NY, USA, v. 39, n. 2, p. 61–64, 2005.

STEWART, S. et al. The linear algebra curriculum study group (lacsg 2.0) recommendations. **Notices of the American Mathematical Society**, v. 69, n. 5, 2022.

STRANG, G. **Linear algebra and its applications**. [S.l.]: Cengage Learning, 2016.

TEIXEIRA, K. C. B.; FONTENELE, F. C. F. Metodologia peer instruction no ensino de matrizes: um relato de experiência na disciplina de álgebra linear. **Educação Matemática em Revista-RS**, v. 1, n. 18, 2017.