UNIVERSIDADE FEDERAL DE UBERLÂNDIA

Lucas de Oliveira Ketelhut

Aplicação web para visualização interativa do método Simplex no ensino de Programação Linear

Uberlândia, Brasil

2025

UNIVERSIDADE FEDERAL DE UBERLÂNDIA

Lucas de Oliveira Ketelhut

Aplicação web para visualização interativa do método Simplex no ensino de Programação Linear

Trabalho de conclusão de curso apresentado à Faculdade de Computação da Universidade Federal de Uberlândia, como parte dos requisitos exigidos para a obtenção título de Bacharel em Sistemas de Informação.

Orientador: Paulo Henrique Ribeiro Gabriel

Universidade Federal de Uberlândia – UFU

Faculdade de Computação

Bacharelado em Sistemas de Informação

Uberlândia, Brasil

2025

Resumo

O método Simplex é um algoritmo fundamental na resolução de problemas de Programação Linear, mas seu aprendizado pode ser abstrato e desafiador. Este trabalho apresenta o desenvolvimento de uma aplicação web de código aberto, denominada *simplex-resolver*, cujo objetivo é facilitar o ensino e a compreensão do método Simplex através de uma abordagem interativa e didática. A ferramenta, desenvolvida com React, Radix UI e Tailwind CSS, permite que os usuários insiram problemas de programação linear e visualizem, passo a passo, as iterações do algoritmo Simplex apresentadas em formato tabular claro e educativo. A aplicação é hospedada na Vercel e o código-fonte é versionado no GitHub, garantindo acessibilidade e transparência. A arquitetura, totalmente executada no lado do cliente, elimina a necessidade de infraestrutura de servidor. O trabalho descreve a engenharia de software por trás do projeto, realiza uma análise comparativa com ferramentas similares, demonstrando os diferenciais da solução proposta, e apresenta estudos de caso que validam sua funcionalidade e eficácia como recurso didático.

Palavras-chave: Programação Linear, Método Simplex, Ensino, React, Aplicação Web.

Abstract

The Simplex method is a fundamental algorithm for solving Linear Programming problems, but learning it can be abstract and challenging. This monography presents the development of an open-source web application, called *simplex-resolver*, whose objective is to facilitate the teaching and understanding of the Simplex method through an interactive and educational approach. Developed with React, Radix UI, and Tailwind CSS, the tool allows users to enter linear programming problems and view, step-by-step, iterations of the Simplex algorithm presented in a clear and educational tabular format. The application is hosted on Vercel, and the source code is versioned on GitHub, ensuring accessibility and transparency. The architecture, executed entirely on the client side, eliminates the need for server infrastructure. We describe the software engineering behind the project, performs a comparative analysis with similar tools, demonstrating the differences of the proposed solution, and presents case studies that validate its functionality and effectiveness as a teaching resource. **Palavras-chave**: Linear Programming, Simplex

Method, Teaching, React, Web Application.

Lista de ilustrações

Sumário

1 Introdução

O método Simplex, desenvolvido por George Dantzig em 1947 (DANTZIG, 1963), é um dos algoritmos mais influentes para a resolução de problemas de Programação Linear, sendo amplamente aplicado em áreas como logística, economia e engenharia, onde a otimização de recursos é crucial (TAHA, 2017). Apesar de sua importância, o aprendizado do método Simplex pode ser um desafio para estudantes, dada a natureza abstrata de suas etapas iterativas (LAZARIDIS; SAMARAS, 2003). A dificuldade em visualizar a progressão do algoritmo até a solução ótima pode criar uma barreira na compreensão conceitual do processo.

Neste contexto, a utilização de ferramentas computacionais interativas surge como uma abordagem pedagógica eficaz (MILETIĆ; LEŠAJA, 2016). A capacidade de acompanhar passo a passo as iterações do algoritmo Simplex, visualizando as transformações das tabelas e a evolução da solução, pode transformar a experiência de aprendizado, tornando-a mais concreta e intuitiva (LAZARIDIS et al., 2007). Este trabalho se insere nesse cenário, propondo o desenvolvimento de uma aplicação web de código aberto, o *simplex-resolver*, focada especificamente no ensino do método Simplex por meio da apresentação interativa de suas iterações tabulares.

O objetivo principal deste trabalho é, portanto, projetar e implementar uma ferramenta educacional que permita a estudantes e professores explorar problemas de programação linear de forma interativa e didática. A aplicação permite que o usuário defina uma função objetivo e um conjunto de restrições, e então acompanhe, passo a passo, as iterações do método Simplex apresentadas em formato tabular claro e educativo. Diferentemente de calculadoras ou *solvers* genéricos, o foco do *simplex-resolver* está na clareza didática e na experiência do usuário, buscando preencher uma lacuna deixada por ferramentas que, embora poderosas, não são projetadas com um propósito primariamente pedagógico.

Para alcançar este objetivo, a aplicação foi desenvolvida utilizando tecnologias web modernas, com destaque para a biblioteca **React** para a construção da interface de usuário reativa, **Radix UI** para componentes de interface acessíveis e **Tailwind CSS** para estilização responsiva. O projeto utiliza **GitHub** para versionamento do código-fonte e **Vercel** para hospedagem em nuvem, garantindo acessibilidade e disponibilidade contínua da ferramenta educacional.

O restante desta monografia está estruturado da seguinte forma: o Capítulo 2 apresenta a base teórica sobre programação linear e as tecnologias empregadas. O Capítulo 3 analisa ferramentas similares e posiciona este trabalho no cenário atual. O Capítulo 4

detalha a arquitetura e a implementação da solução. O Capítulo 5 demonstra a aplicação através de estudos de caso práticos. Finalmente, o Capítulo 6 apresenta as conclusões, limitações e direções para trabalhos futuros.

2 Fundamentação Teórica

Este capítulo aborda os conceitos essenciais que fundamentam o desenvolvimento do projeto, incluindo os princípios da programação linear, o funcionamento do método Simplex e as tecnologias web utilizadas na implementação da ferramenta interativa.

2.1 Programação Linear

Programação Linear (PL) é uma técnica matemática utilizada para otimizar (maximizar ou minimizar) uma função linear, denominada função objetivo, sujeita a um conjunto de restrições lineares (inequações ou equações) (HILLIER; LIEBERMAN, 2015). Desenvolvida formalmente durante a década de 1940, a PL tornou-se uma ferramenta indispensável na pesquisa operacional, aplicada a uma vasta gama de problemas práticos, como planejamento de produção, logística de transporte, alocação de recursos financeiros e design de dietas (TAHA, 2017).

Um problema de programação linear é caracterizado por três componentes principais:

- **Função Objetivo:** Uma expressão linear que se deseja otimizar. Por exemplo, maximizar o lucro $Z = 3x + 5y$.
- **Restrições:** Um conjunto de inequações ou equações lineares que limitam as variáveis de decisão. Por exemplo, $2x + y \leq 100$.
- **Restrições de Não-Negatividade:** As variáveis de decisão devem ser não-negativas (e.g., $x \geq 0, y \geq 0$), condição comum em problemas do mundo real.

A solução de um problema de PL corresponde a um conjunto de valores para as variáveis de decisão que satisfaz todas as restrições e otimiza o valor da função objetivo. A região definida pelo conjunto de todas as soluções viáveis é chamada de região factível. Para problemas com duas variáveis, essa região pode ser representada graficamente como um polígono convexo no plano cartesiano.

2.2 O Método Simplex Tabular

O método Simplex é um algoritmo iterativo que sistematicamente explora os vértices da região factível em busca da solução ótima (DANTZIG, 1963). Ele parte de um

vértice inicial e move-se para um vértice adjacente que melhore o valor da função objetivo, repetindo o processo até que nenhuma melhoria seja possível, garantindo assim a descoberta da solução ótima (TAHA, 2017).

O algoritmo Simplex opera através de uma sequência de tabelas, onde cada iteração representa uma transformação matemática que conduz à solução ótima. Os passos principais são:

1. Converter o problema para a forma padrão, introduzindo variáveis de folga ou excesso.
2. Construir a tabela inicial do Simplex com as variáveis básicas e não-básicas.
3. Identificar a variável de entrada (coluna pivô) através do teste de otimalidade.
4. Determinar a variável de saída (linha pivô) através do teste da razão mínima.
5. Realizar operações de pivoteamento para obter a nova tabela.
6. Repetir o processo até que o critério de otimalidade seja satisfeito.

Esta abordagem tabular é considerada mais adequada para o ensino, pois permite acompanhar passo a passo a evolução do algoritmo, compreendendo como cada iteração aproxima a solução do ponto ótimo, o que motivou a criação da ferramenta *simplex-resolver*.

A fim de ilustrar o funcionamento do método Simplex tabular, considere o seguinte problema de maximização (TAHA, 2017):

Maximizar

$$Z = 5x_1 + 4x_2 \quad \text{(Função Objetivo)} \tag{2.1}$$

Sujeito às restrições:

$$6x_1 + 4x_2 \leq 24 \tag{2.2}$$

$$x_1 + 2x_2 \leq 6 \tag{2.3}$$

$$-x_1 + x_2 \leq 1 \tag{2.4}$$

$$x_2 \leq 2 \tag{2.5}$$

$$x_1, x_2 \geq 0 \tag{2.6}$$

Nesse caso, busca-se determinar os valores de x_1 e x_2 que maximizam a função Z, respeitando os limites estabelecidos pelas quatro restrições.

Para aplicar o Método Simplex, transformamos o problema para sua **forma padrão**, que possui as seguintes características:

- Função objetivo de maximização
- Todas as restrições são igualdades
- Todas as variáveis são não-negativas
- Lados direitos das restrições são não-negativos

Adicionamos **variáveis de folga** s_1, s_2, s_3, s_4 para transformar as desigualdades em igualdades[1]:

$$\begin{aligned} 6x_1 + 4x_2 + s_1 &= 24 \\ x_1 + 2x_2 + s_2 &= 6 \\ -x_1 + x_2 + s_3 &= 1 \\ x_2 + s_4 &= 2 \end{aligned}$$

A tabela Simplex organiza todas as informações do problema de forma sistemática:

Base	x_1	x_2	s_1	s_2	s_3	s_4	Solução
Z	-5	-4	0	0	0	0	0
s_1	6	4	1	0	0	0	24
s_2	1	2	0	1	0	0	6
s_3	-1	1	0	0	1	0	1
s_4	0	1	0	0	0	1	2

Analisando a tabela inicial, tem-se que:

- **Base inicial:** s_1, s_2, s_3, s_4 (variáveis de folga)
- **Solução inicial:** $x_1 = 0$, $x_2 = 0$, $s_1 = 24$, $s_2 = 6$, $s_3 = 1$, $s_4 = 2$
- **Valor de Z:** $Z = 0$ (não produzimos nada)
- **Custos reduzidos:** -5 para x_1 e -4 para x_2 indicam quanto Z aumenta por unidade de cada variável

Para continuar a execução do método, é necessário definir qual variável entrará na base. Para isso, escolhemos a variável com coeficiente mais negativo na linha Z, ou seja,

[1] Cada variável de folga s_i representa a quantidade não utilizada do recurso correspondente à restrição i.

aquela que possui maior potencial de melhoria. Nesse caso, a variável que entra na base é x_1, pois seu coeficiente é -5.

Para saída da base, aplicamos a **regra do mínimo quociente**:

$$\begin{aligned}
\frac{24}{6} &= 4 \quad \text{(para } s_1\text{)} \\
\frac{6}{1} &= 6 \quad \text{(para } s_2\text{)} \\
\frac{1}{-1} &= -1 \quad \text{(ignorado, pois o denominador é negativo)} \\
\frac{2}{0} &= \infty \quad \text{(ignorado, pois o denominador é zero)}
\end{aligned}$$

Como a variável s_1 possui menor quociente não negativo (4), s_1 sairá da base. Logo, o elemento pivô é 6, dado pela interseção da coluna x_1 com a linha s_1. Inicia-se, então, as operações de pivoteamento:

1. Linha do pivô: Dividimos toda a linha por 6
2. Demais linhas: Subtraímos múltiplos da nova linha pivô para zerar a coluna x_1

Assim, constrói-se a nova tabela Simplex:

Base	x_1	x_2	s_1	s_2	s_3	s_4	Solução
Z	0	$-\frac{2}{3}$	$\frac{5}{6}$	0	0	0	20
x_1	1	$\frac{2}{3}$	$\frac{1}{6}$	0	0	0	4
s_2	0	$\frac{4}{3}$	$-\frac{1}{6}$	1	0	0	2
s_3	0	$\frac{5}{3}$	$\frac{1}{6}$	0	1	0	5
s_4	0	1	0	0	0	1	2

Após a primeira iteração, temos:

- **Nova solução:** $x_1 = 4$, $x_2 = 0$, $Z = 20$
- **Melhoria:** Z aumentou de 0 para 20
- **Próxima variável candidata:** x_2 (coeficiente $-\frac{2}{3}$ na linha Z)

Repetindo o processo (segunda iteração), obtemos a tabela final do Simplex, dada por:

Base	x_1	x_2	s_1	s_2	s_3	s_4	Solução
Z	0	0	$\frac{3}{4}$	$\frac{1}{2}$	0	0	21
x_1	1	0	$\frac{1}{4}$	$-\frac{1}{2}$	0	0	3
x_2	0	1	$-\frac{1}{8}$	$\frac{3}{4}$	0	0	$\frac{3}{2}$
s_3	0	0	$\frac{3}{8}$	$-\frac{5}{4}$	1	0	$\frac{5}{2}$
s_4	0	0	$\frac{1}{8}$	$-\frac{3}{4}$	0	1	$\frac{1}{2}$

Observa-se que todos os coeficientes na linha Z são não-negativos. Logo, essa é a tabela final e, portanto, temos a seguinte solução ótima:

- $x_1 = 3$
- $x_2 = \frac{3}{2}$
- $Z = 21$

Portanto, o método Simplex convergiu para a solução ótima em duas iterações. A solução encontrada é factível e ótima, maximizando a função objetivo enquanto respeita todas as restrições do problema.

2.3 Tecnologias de Desenvolvimento Web

2.3.1 React

A interface da aplicação foi construída com **React** (FACEBOOK, 2023), uma biblioteca JavaScript de código aberto para a criação de interfaces de usuário (UI), mantida pelo Facebook. Sua principal característica é a componentização, que permite dividir a UI em partes isoladas e reutilizáveis. No contexto deste projeto, o React gerencia o estado da aplicação (os dados da função objetivo e das restrições) e atualiza a interface de forma reativa e eficiente sempre que o usuário interage com os campos de entrada, proporcionando uma experiência fluida e dinâmica.

2.3.2 Radix UI

Para os componentes de interface, foi utilizada a biblioteca **Radix UI** (Radix UI Team, 2023), uma coleção de componentes React de baixo nível, não estilizados e acessíveis. O Radix UI oferece primitivos como botões, formulários, separadores e abas, todos implementados seguindo as melhores práticas de acessibilidade web. Esta escolha foi fundamental para garantir que a aplicação seja utilizável por estudantes com diferentes necessidades, mantendo a conformidade com padrões de acessibilidade enquanto oferece flexibilidade total para customização visual.

2.3.3 Tailwind CSS

A estilização da aplicação foi implementada com **Tailwind CSS** (Tailwind Labs, 2023), um framework CSS utilitário que permite a construção rápida de interfaces personalizadas. O Tailwind CSS oferece classes utilitárias de baixo nível que podem ser compostas para criar qualquer design, proporcionando controle granular sobre a aparência sem a necessidade de escrever CSS customizado. Esta abordagem foi essencial para criar uma interface responsiva e moderna, garantindo que a ferramenta funcione adequadamente em dispositivos desktop, tablet e mobile.

2.3.4 GitHub

O versionamento do código-fonte é realizado através do **GitHub** (GitHub Inc., 2023), uma plataforma de hospedagem de código que utiliza o sistema de controle de versão Git. O GitHub permite o acompanhamento de todas as modificações no código, facilitando a colaboração, o controle de versões e a transparência do desenvolvimento. Para um projeto educacional de código aberto como o *simplex-resolver*, o GitHub garante que o código seja acessível à comunidade acadêmica, permitindo contribuições e melhorias contínuas.

2.3.5 Vercel

A hospedagem da aplicação é realizada através da **Vercel** (Vercel Inc., 2023), uma plataforma de deployment especializada em aplicações frontend. A Vercel oferece integração automática com repositórios GitHub, permitindo *deploy* contínuo sempre que o código é atualizado. Para ferramentas educacionais, a Vercel garante alta disponibilidade, desempenho otimizada e acesso global, essencial para que estudantes e professores possam utilizar a ferramenta de qualquer local e a qualquer momento.

3 Trabalhos Relacionados

Nesta seção, são analisadas outras ferramentas computacionais existentes para a resolução de problemas de programação linear, comparando suas abordagens com a solução desenvolvida neste trabalho, o *simplex-resolver*. A análise foca em aplicações web interativas para contextualizar a contribuição e os diferenciais deste trabalho.

3.1 PHPSimplex

O PHPSimplex (CABELLO, 2024) é uma das ferramentas online mais conhecidas para resolver problemas de programação linear. Ele permite que o usuário insira a função objetivo e as restrições e, em seguida, resolve o problema utilizando o método Simplex tabular. Sua principal força reside em detalhar cada tabela de iteração do algoritmo, o que é útil para estudantes que estão aprendendo o processo algébrico.

No entanto, sua abordagem é puramente numérica e tabular, apresentando apenas as iterações do algoritmo sem contexto visual adicional. Em comparação, o *simplex-resolver*, aqui proposto, foca na apresentação didática e interativa das iterações do método Simplex, permitindo que o usuário acompanhe passo a passo a evolução do algoritmo de forma clara e educativa, o que representa uma abordagem pedagógica mais acessível e intuitiva para estudantes iniciantes.

3.2 Linear Programming Grapher

O LGP é uma aplicação web (ZWEIGMEDIA, 2024) que foca na resolução gráfica de problemas de PL com duas variáveis. A ferramenta desenha as retas das restrições e permite que o usuário clique nos vértices para encontrar a solução ótima, oferecendo uma abordagem visual para a compreensão da programação linear.

Contudo, sua interface de usuário é considerada datada e a experiência de uso é menos intuitiva. Diferentemente desta abordagem gráfica, o *simplex-resolver* adota uma estratégia educacional focada no algoritmo Simplex tabular, oferecendo uma interface moderna e reativa construída com React. A solução proposta neste trabalho se diferencia ao apresentar as iterações do método de forma clara e didática, proporcionando uma experiência de aprendizado mais estruturada e algorítmica.

3.3 Análise Comparativa

Ao posicionar o *simplex-resolver* em relação às ferramentas existentes, seus principais diferenciais competitivos emergem:

- **Foco Pedagógico Didático:** Diferente do PHPSimplex, que apresenta apenas as tabelas finais, o *simplex-resolver* enfatiza a apresentação clara e educativa de cada iteração do algoritmo, tornando o processo de aprendizado mais acessível para iniciantes.
- **Interface Moderna e Reativa:** Em contraste com ferramentas mais antigas, a aplicação utiliza React para oferecer uma experiência fluida, moderna e responsiva, funcionando bem em desktops e dispositivos móveis.
- **Código Aberto e Acessibilidade:** A ferramenta é de código aberto e totalmente *client-side*, o que significa que pode ser executada em qualquer navegador moderno sem a necessidade de instalação, plugins ou conexão com um servidor, removendo barreiras de acesso para estudantes e educadores.

Trabalhos recentes, como o GILP (ROBBINS et al., 2023) também abordam a visualização interativa do método Simplex, com foco similar no algoritmo tabular e suas iterações. O *simplex-resolver* se alinha a esta tendência, mas se diferencia pela simplicidade de uso e foco específico no ensino. Em suma, o *simplex-resolver* preenche uma lacuna ao combinar uma abordagem pedagógica focada no método Simplex tabular com uma implementação tecnológica moderna e acessível, criando uma ferramenta otimizada para o ensino e aprendizado da programação linear.

4 Desenvolvimento da Aplicação

Este capítulo detalha a concepção e a implementação da aplicação web *simplex-resolver*. Abordam-se o levantamento de requisitos, a metodologia de desenvolvimento, a arquitetura do sistema, as tecnologias selecionadas, a estrutura do código-fonte e o fluxo algorítmico que possibilita a resolução gráfica dos problemas de programação linear.

4.1 Levantamento de Requisitos

O desenvolvimento da aplicação iniciou-se com a identificação e especificação dos requisitos funcionais e não-funcionais, baseados nas necessidades pedagógicas identificadas e nas limitações das ferramentas existentes analisadas no capítulo anterior.

4.1.1 Requisitos Funcionais

Os requisitos funcionais definem as funcionalidades que a aplicação deve oferecer aos usuários:

- **RF001 - Entrada de Função Objetivo:** O sistema deve permitir ao usuário inserir os coeficientes da função objetivo linear (formato $Z = ax_1 + bx_2$) e selecionar entre maximização ou minimização.
- **RF002 - Entrada de Restrições:** O sistema deve permitir a inserção de múltiplas restrições lineares no formato $ax_1 + bx_2 \leq c$, $ax_1 + bx_2 \geq c$ ou $ax_1 + bx_2 = c$.
- **RF003 - Adição e Remoção Dinâmica:** O sistema deve permitir adicionar e remover restrições dinamicamente durante a interação.
- **RF004 - Processamento do Método Simplex:** O sistema deve implementar o algoritmo Simplex para resolver o problema de programação linear inserido.
- **RF005 - Apresentação das Iterações:** O sistema deve apresentar cada iteração do método Simplex em formato tabular claro e educativo.
- **RF006 - Solução Ótima:** O sistema deve identificar e apresentar a solução ótima, exibindo os valores das variáveis e o valor da função objetivo.
- **RF007 - Atualização em Tempo Real:** O sistema deve processar e apresentar os resultados instantaneamente conforme o usuário modifica os dados de entrada.
- **RF008 - Limpeza de Dados:** O sistema deve oferecer funcionalidade para limpar todos os campos e reiniciar o problema.

4.1.2 Requisitos Não-Funcionais

Os requisitos não-funcionais especificam as qualidades e restrições do sistema:

- **RNF001 - Usabilidade:** A interface deve ser intuitiva e adequada para usuários com conhecimento básico em programação linear.
- **RNF002 - Desempenho:** O sistema deve processar e apresentar os resultados em tempo inferior a 500ms para problemas com até 10 restrições.
- **RNF003 - Compatibilidade:** A aplicação deve funcionar nos principais navegadores web (Chrome, Firefox, Safari, Edge) em suas versões atuais.
- **RNF004 - Responsividade:** A interface deve ser responsiva e funcional em dispositivos desktop, tablet e mobile.
- **RNF005 - Acessibilidade:** A aplicação deve ser executável sem instalação, plugins ou conexão com servidor.
- **RNF006 - Código Aberto:** O código-fonte deve estar disponível publicamente sob licença open source.

4.2 Metodologia de Desenvolvimento

O desenvolvimento da aplicação seguiu uma abordagem iterativa e incremental, com foco na prototipagem rápida e testes contínuos. A metodologia adotada pode ser caracterizada pelos aspectos descritos a seguir.

4.2.1 Abordagem de Desenvolvimento

Foi utilizada uma metodologia ágil simplificada, adequada para projetos individuais, com as seguintes características:

- **Desenvolvimento Iterativo:** O projeto foi dividido em iterações curtas, cada uma focada em um conjunto específico de funcionalidades.
- **Prototipagem Rápida:** Criação de protótipos funcionais para validação precoce dos conceitos e interface.
- **Testes Contínuos:** Validação constante das funcionalidades através de testes manuais com problemas conhecidos.

4.2.2 Fases de Desenvolvimento

O desenvolvimento deste trabalho foi organizado nas seguintes fases:

1. **Análise e Design:** Estudo das ferramentas existentes, definição de requisitos e esboço da interface.
2. **Implementação do Core:** Desenvolvimento do algoritmo Simplex e das funções de processamento matemático.
3. **Interface de Usuário:** Criação da interface React com formulários de entrada e área de apresentação dos resultados.
4. **Apresentação Tabular:** Implementação da apresentação das iterações do método Simplex em formato tabular educativo.
5. **Integração e Refinamento:** Conexão entre os componentes e refinamento da experiência do usuário.
6. **Testes e Validação:** Testes extensivos com diversos tipos de problemas e correção de bugs.

4.3 Arquitetura da Aplicação

A aplicação foi projetada com uma arquitetura exclusivamente *client-side* (lado do cliente). Essa decisão estratégica implica que todo o processamento de dados, os cálculos matemáticos e a apresentação das iterações do método Simplex ocorrem diretamente no navegador do usuário, sem a necessidade de comunicação com um servidor de *back-end*. As principais vantagens desta abordagem para o projeto são a simplicidade de distribuição (*deploy*), a acessibilidade, o desempenho interativo e a capacidade de uso offline.

4.4 Estrutura do Projeto e Fluxo de Dados

O código-fonte foi organizado seguindo as melhores práticas de desenvolvimento React com TypeScript, separando claramente as responsabilidades para facilitar a manutenção e escalabilidade. A estrutura principal é composta por: pasta `/src/components` contendo os componentes React organizados por funcionalidade, pasta `/src/lib` com as funções utilitárias e implementação do algoritmo Simplex, pasta `/src/hooks` para custom hooks do React, e arquivo `/src/index.css` para os estilos globais utilizando Tailwind CSS.

O fluxo de dados na aplicação segue o padrão reativo do React, de acordo com o seguinte fluxo:

1. O usuário interage com o componente `ProblemInput.tsx`, inserindo os coeficientes da função objetivo e das restrições através de formulários controlados.

2. O estado da aplicação é gerenciado pelo componente principal `App.tsx`, que atualiza automaticamente a interface a cada modificação dos dados de entrada.

3. Quando os dados estão completos, o algoritmo implementado em

   `/src/lib/simplexAlgorithm.ts` é executado, processando o problema de programação linear.

4. O algoritmo converte o problema para a forma padrão, introduzindo variáveis de folga quando necessário, e executa as iterações do método Simplex.

5. Cada iteração do algoritmo é armazenada em estruturas de dados apropriadas, permitindo o acompanhamento passo a passo do processo de resolução.

6. Os resultados processados (iterações tabulares, solução ótima, valores das variáveis) são passados para o componente `SimplexResults.tsx`, que apresenta os dados em formato educativo e de fácil compreensão.

7. As funções de validação em `/src/lib/validation.ts` garantem a integridade dos dados de entrada, enquanto `/src/lib/utils.js` fornece funções auxiliares para formatação e manipulação de dados.

Este fluxo arquitetural garante que a apresentação dos resultados seja um reflexo instantâneo e preciso dos dados inseridos pelo usuário, proporcionando uma experiência didática fluida e responsiva para o aprendizado do método Simplex.

5 Resultados e Estudos de Caso

Neste capítulo, a funcionalidade e a eficácia da aplicação *simplex-resolver* são demonstradas através da resolução de problemas clássicos de programação linear. Para cada problema, são apresentados o modelo matemático, uma captura de tela da aplicação mostrando as iterações do método Simplex em formato tabular e uma breve análise do resultado.

5.1 Estudo de Caso 1: Problema de Produção

Este problema, adaptado de (TAHA, 2017), envolve uma empresa que produz dois tipos de produtos e deseja maximizar seu lucro, sujeito a restrições de recursos.

5.1.1 Modelo Matemático

Maximizar $Z = 5x_1 + 4x_2$

Sujeito a:

- $6x_1 + 4x_2 \leq 24$
- $x_1 + 2x_2 \leq 6$
- $-x_1 + x_2 \leq 1$
- $x_2 \leq 2$
- $x_1, x_2 \geq 0$

5.1.2 Resultado na Aplicação

As Figuras de 1 a 4 mostram a interface do *simplex-resolver* com o problema inserido e as iterações do método Simplex apresentadas em formato tabular.

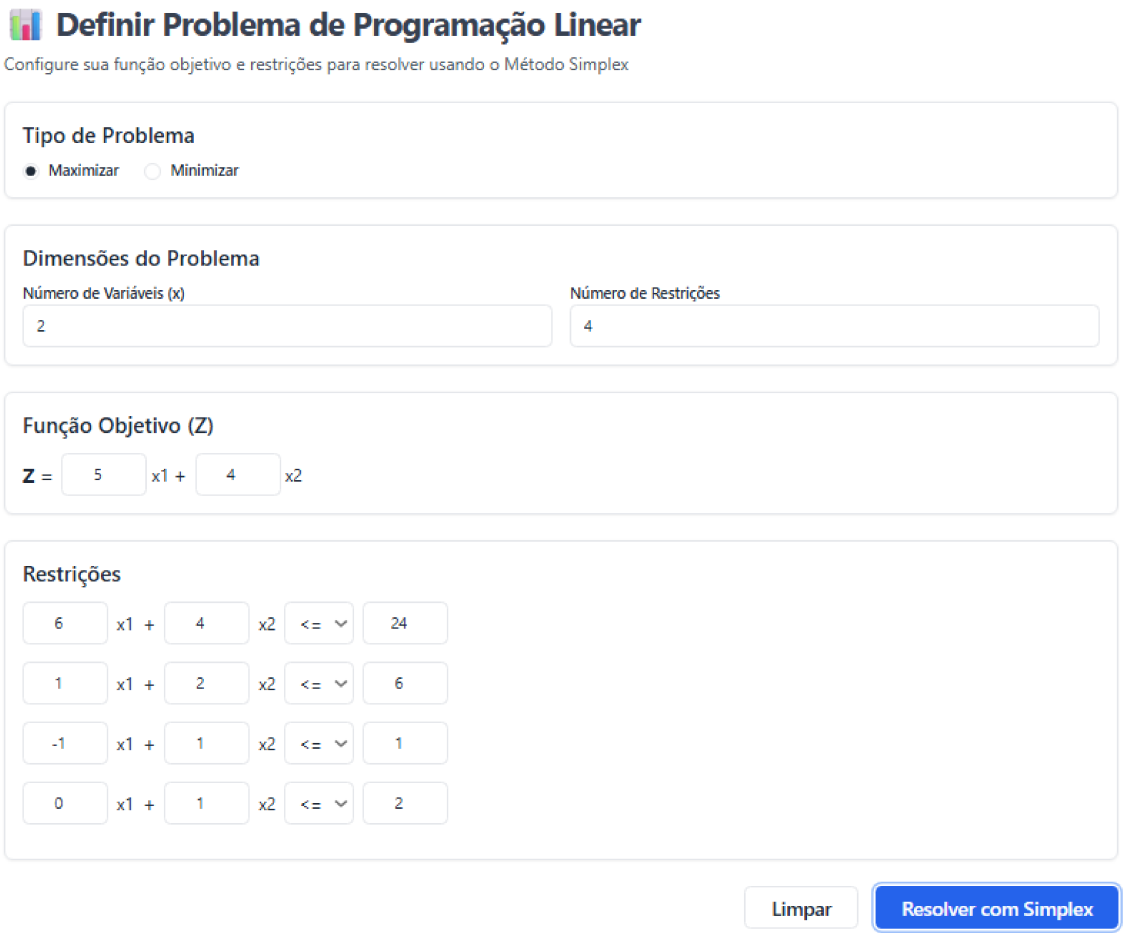
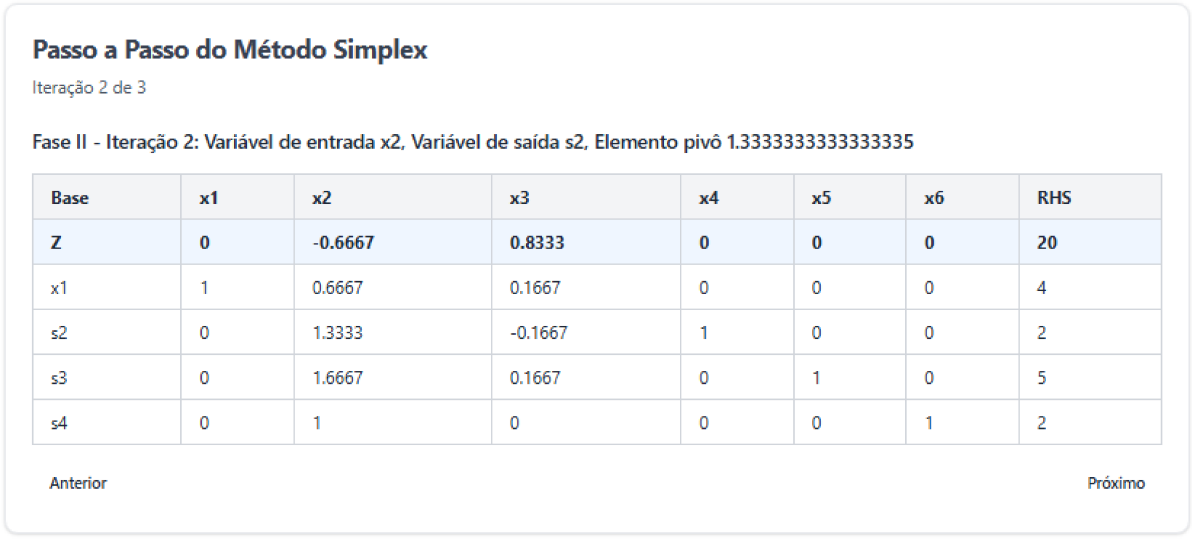


Figura 1 – Interface com o problema inserido para o Estudo de Caso 1 na aplicação.

Base	x1	x2	x3	x4	x5	x6	RHS
Z	**-5**	**-4**	**0**	**0**	**0**	**0**	**0**
s1	6	4	1	0	0	0	24
s2	1	2	0	1	0	0	6
s3	-1	1	0	0	1	0	1
s4	0	1	0	0	0	1	2

Figura 2 – Iteração 1/3 do método Simplex para o Estudo de Caso 1 na aplicação.



Passo a Passo do Método Simplex

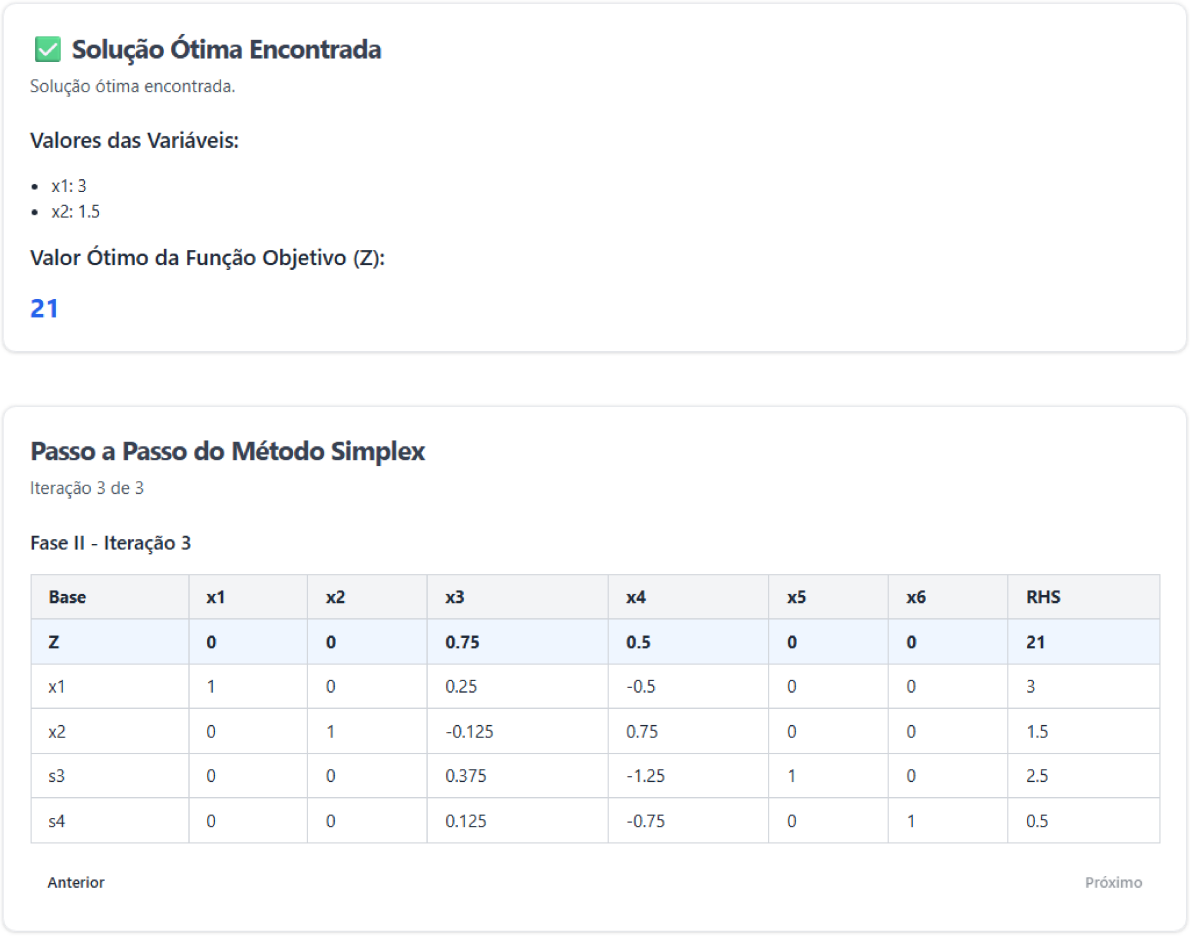
Iteração 2 de 3

Fase II - Iteração 2: Variável de entrada x2, Variável de saída s2, Elemento pivô 1.3333333333333335

Base	x1	x2	x3	x4	x5	x6	RHS
Z	**0**	**-0.6667**	**0.8333**	**0**	**0**	**0**	**20**
x1	1	0.6667	0.1667	0	0	0	4
s2	0	1.3333	-0.1667	1	0	0	2
s3	0	1.6667	0.1667	0	1	0	5
s4	0	1	0	0	0	1	2

Anterior Próximo

Figura 3 – Iteração 2/3 do método Simplex para o Estudo de Caso 1 na aplicação.



Solução Ótima Encontrada

Solução ótima encontrada.

Valores das Variáveis:

- x1: 3
- x2: 1.5

Valor Ótimo da Função Objetivo (Z):

21

Passo a Passo do Método Simplex

Iteração 3 de 3

Fase II - Iteração 3

Base	x1	x2	x3	x4	x5	x6	RHS
Z	**0**	**0**	**0.75**	**0.5**	**0**	**0**	**21**
x1	1	0	0.25	-0.5	0	0	3
x2	0	1	-0.125	0.75	0	0	1.5
s3	0	0	0.375	-1.25	1	0	2.5
s4	0	0	0.125	-0.75	0	1	0.5

Anterior Próximo

Figura 4 – Iteração 3/3 do método Simplex e solução ótima encontrada para o Estudo de Caso 1 na aplicação.

A aplicação executa corretamente o algoritmo Simplex, apresentando cada iteração em formato tabular claro e educativo. A solução ótima encontrada é $x_1 = 3$ e $x_2 = 1.5$, que resulta em um lucro máximo de $Z = 21$. As tabelas mostram passo a passo como o algoritmo converge para esta solução, validando a capacidade da ferramenta de resolver o problema e apresentar o processo de forma didática.

5.2 Estudo de Caso 2: Problema de Otimização de Custos

Este problema envolve uma empresa que deseja minimizar os custos operacionais de dois processos produtivos, sujeito a restrições de capacidade mínima e demanda de produção.

5.2.1 Modelo Matemático

Minimizar $Z = 65x_1 + 30x_2$

Sujeito a:

- $2x_1 + 3x_2 \geq 7$
- $3x_1 + 2x_2 \geq 9$
- $x_1 \geq 1$
- $x_1, x_2 \geq 0$

5.2.2 Resultado na Aplicação

As Figuras 5 e 6 mostram a interface do *simplex-resolver* com o problema inserido e apresentam as iterações finais do método Simplex para o problema de minimização.

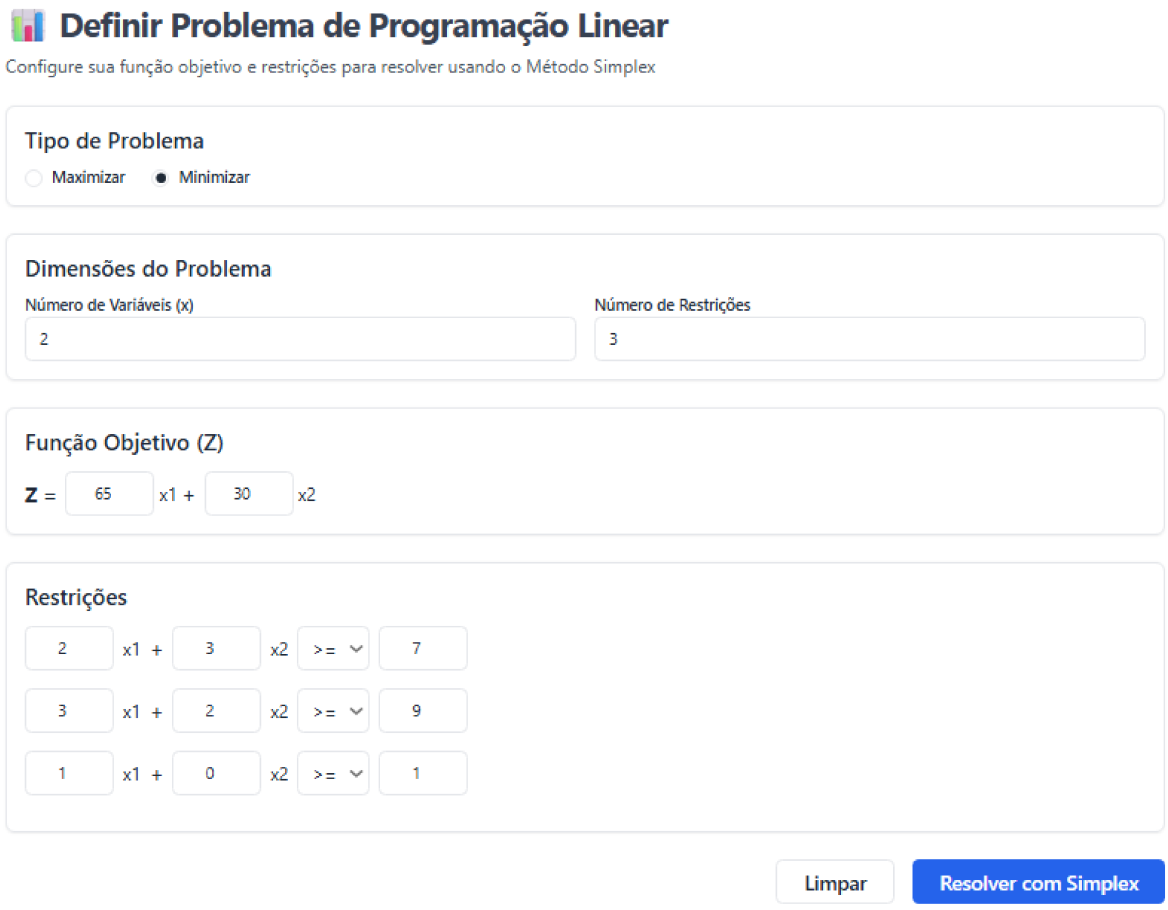


Figura 5 – Interface com o problema inserido para o Estudo de Caso 2 na aplicação.

Figura 6 – Iterações finais do método Simplex e solução ótima encontrada para o Estudo de Caso 2 na aplicação.

Neste problema de minimização, a aplicação converte adequadamente o problema para a forma padrão e executa o algoritmo Simplex. As tabelas mostram claramente cada iteração do processo, desde a tabela inicial até a convergência para a solução ótima. Este exemplo demonstra a capacidade da ferramenta de resolver problemas de minimização com restrições do tipo "maior ou igual", validando sua versatilidade para diferentes tipos de problemas de programação linear e sua eficácia como ferramenta educacional.

5.3 Testes e Validação

Para garantir a confiabilidade e precisão da aplicação, foi realizada uma bateria de testes abrangente, incluindo testes funcionais, de usabilidade e de desempenho.

5.3.1 Testes Funcionais

Os testes funcionais verificaram se todos os requisitos funcionais foram adequadamente implementados:

- **Teste de Entrada de Dados:** Verificação da correta interpretação dos coeficientes inseridos pelo usuário, incluindo valores negativos, decimais e zero.
- **Teste de Cálculo Matemático:** Validação dos algoritmos de interseção de retas e identificação de vértices através de comparação com soluções analíticas conhecidas.
- **Teste de Casos Especiais:** Verificação do comportamento da aplicação em situações como região factível vazia, região não limitada e múltiplas soluções ótimas.
- **Teste de Responsividade:** Confirmação de que a interface se adapta adequadamente a diferentes tamanhos de tela e dispositivos.

5.3.2 Testes de Usabilidade

A usabilidade da ferramenta foi avaliada considerando o público-alvo (estudantes e professores de programação linear):

- **Intuitividade da Interface:** A disposição dos elementos e a sequência de interação foram projetadas para serem autoexplicativas.
- **Feedback Visual:** O sistema fornece feedback imediato através da atualização das tabelas em tempo real, permitindo ao usuário compreender o impacto de cada modificação nos dados de entrada.
- **Tratamento de Erros:** A aplicação trata adequadamente entradas inválidas, exibindo mensagens claras quando necessário.

6 Conclusão

Este trabalho teve como objetivo principal o desenvolvimento de uma aplicação web interativa, o *simplex-resolver*, para auxiliar no ensino e aprendizado do método Simplex através da apresentação didática de suas iterações tabulares. A ferramenta desenvolvida cumpre com sucesso os objetivos propostos, oferecendo uma plataforma de código aberto, acessível e focada na experiência didática do usuário.

Ao longo do desenvolvimento, foi realizada uma revisão da literatura e de ferramentas existentes, o que permitiu identificar uma lacuna no que tange a soluções que combinam uma abordagem pedagógica didática com tecnologias web modernas. O *simplex-resolver* se posiciona como uma contribuição relevante ao preencher essa lacuna, fornecendo uma interface reativa e intuitiva onde os estudantes podem acompanhar passo a passo as iterações do método Simplex e compreender a evolução do algoritmo em tempo real.

Os principais resultados alcançados incluem:

- **Implementação Completa:** Todos os requisitos funcionais foram atendidos, resultando em uma aplicação funcional e robusta.
- **Validação Técnica:** Os algoritmos implementados demonstraram precisão matemática através dos estudos de caso realizados.
- **Interface Moderna:** A utilização de React, Radix UI e Tailwind CSS resultou em uma experiência de usuário superior às ferramentas existentes.
- **Acessibilidade:** A arquitetura *client-side* garante que a ferramenta seja acessível sem barreiras técnicas.

Em resumo, as principais contribuições deste trabalho para a área de ensino de Programação Linear são:

- Desenvolvimento de uma aplicação que combina apresentação didática do método Simplex com tecnologias web modernas.
- Implementação de uma metodologia de ensino que prioriza a compreensão algorítmica do método Simplex através de iterações tabulares claras.
- Disponibilização da ferramenta como software livre, permitindo adaptações e melhorias pela comunidade acadêmica.

- Produção de documentação detalhada sobre requisitos, arquitetura e implementação, servindo como referência para trabalhos similares.

Apesar dos resultados positivos, algumas limitações foram identificadas:

- **Ausência de Validação Empírica:** Não foram realizados estudos controlados com estudantes para medir quantitativamente o impacto pedagógico.
- **Funcionalidades Avançadas:** A ferramenta não inclui recursos como análise de sensibilidade, geração automática de exercícios e construção de gráficos.

Com base nos resultados obtidos e nas limitações identificadas, propõem-se as seguintes direções para trabalhos futuros:

Extensões Técnicas

- **Gerar gráfico:** Adição do recurso de gráfico visual para melhor entendimento do usuário.
- **Análise de Sensibilidade:** Adição de funcionalidades para análise de sensibilidade dos parâmetros do problema.
- **Exportação de Resultados:** Implementação de recursos para exportar tabelas e relatórios em formatos PDF e PNG.
- **Modo Passo-a-Passo:** Desenvolvimento de um modo tutorial que guia o usuário através das etapas do método Simplex tabular.

Validação Pedagógica

- **Estudos Empíricos:** Realização de experimentos controlados com estudantes para avaliar a eficácia pedagógica da ferramenta.
- **Integração Curricular:** Desenvolvimento de material didático complementar para integração da ferramenta em disciplinas de pesquisa operacional.
- **Feedback de Usuários:** Coleta sistemática de feedback de professores e estudantes para melhorias contínuas.

Expansão Funcional

- **Biblioteca de Exercícios:** Criação de uma biblioteca de problemas pré-definidos com diferentes níveis de dificuldade.

- **Sistema de Gamificação:** Implementação de elementos de gamificação para aumentar o engajamento dos estudantes.
- **Suporte Multilíngue:** Adição de suporte para múltiplos idiomas, expandindo o alcance da ferramenta.

Acesso

- **Repositório público:** O repositório do projeto desenvolvido neste trabalho pode ser acessado em <https://github.com/LucasKetelhut/simplex-resolver>.
- **Acesso a aplicação:** A aplicação desenvolvida neste trabalho pode ser acessada em <https://simplex-resolver.vercel.app>.

Referências

CABELLO, J. J. **PHPSimplex: Online tool for solving linear programming problems**. 2024. <https://www.phpsimplex.com/en/>. Acessado em: 28/11/2025. Citado na página 15.

DANTZIG, G. B. **Linear programming and extensions**. Princeton, NJ: Princeton University Press, 1963. Citado 2 vezes nas páginas 7 e 9.

FACEBOOK. **React: A JavaScript library for building user interfaces**. 2023. <https://react.dev/>. Acessado em: 28/11/2025. Citado na página 13.

GitHub Inc. **GitHub: Where the world builds software**. 2023. <https://github.com/>. Acessado em: 28/11/2025. Citado na página 14.

HILLIER, F. S.; LIEBERMAN, G. J. **Introduction to operations research**. 10. ed. New York: McGraw-Hill Education, 2015. Citado na página 9.

LAZARIDIS, V.; PAPARRIZOS, K.; SAMARAS, N.; STEPHANIDES, G. Visual linprog: A web-based educational software for linear programming. **Computer Applications in Engineering Education**, Wiley Online Library, v. 15, n. 1, p. 1–14, 2007. Citado na página 7.

LAZARIDIS, V.; SAMARAS, N. Visualization and teaching simplex algorithm. In: IEEE. **Proceedings 3rd IEEE International Conference on Advanced Learning Technologies**. [S.l.], 2003. p. 134–138. Citado na página 7.

MILETIĆ, L.; LEŠAJA, G. Research and evaluation of the effectiveness of e-learning in the case of linear programming. **Croatian Operational Research Review**, Croatian Operational Research Society, v. 7, n. 2, p. 213–229, 2016. Citado na página 7.

Radix UI Team. **Radix UI: Low-level UI primitives with a focus on accessibility**. 2023. <https://www.radix-ui.com/>. Acessado em: 28/11/2025. Citado na página 13.

ROBBINS, H. W.; GUTEKUNST, S. C.; SHMOYS, D. B.; WILLIAMSON, D. P. Gilp: An interactive tool for visualizing the simplex algorithm. In: ACM. **Proceedings of the 54th ACM Technical Symposium on Computer Science Education V. 1**. Toronto, ON, Canada, 2023. p. 12–18. Citado na página 16.

TAHA, H. A. **Operations research: an introduction**. 11. ed. Boston: Pearson Education, 2017. ISBN 978-0134444017. Citado 4 vezes nas páginas 7, 9, 10 e 21.

Tailwind Labs. **Tailwind CSS: A utility-first CSS framework**. 2023. <https://tailwindcss.com/>. Acessado em: 28/11/2025. Citado na página 14.

Vercel Inc. **Vercel: Develop. Preview. Ship.** 2023. <https://vercel.com/>. Acessado em: 28/11/2025. Citado na página 14.

ZWEIGMEDIA. **Linear Programming Grapher (two variables)**. 2024. <https://www.zweigmedia.com/utilities/lpg/index.html>. Acessado em: 28/11/2025. Citado na página 15.